PROYECTO FINAL
SEGUNDA ENTREGA

Julián Y. Torres Torres, Carlos J. Barreto Mora, Juan A. Angulo Rincón
Sistema de recomendación de productos BAC
Universidad de los Andes, Bogotá, Colombia
{jy.torres, c.barretom, ja.angulor1}@uniandes.edu.co
https://github.com/jaangulor/Ciencia-de-datos-aplicada---Proyecto-final.git
Fecha de presentación: noviembre 03 de 2022

Tabla de contenido

1. Preparación de los datos

Para la ejecución del proyecto, contamos con una base de datos con 500000 prospectos y 49 características medidas y facilitadas por una entidad bancaria.

En primer lugar, realizamos la validación de la calidad de los datos, evidenciando que no se presentan registros duplicados en toda la base de datos, teniendo un registro único por cliente. También evidenciamos que hay características que podemos omitir para el interés del desarrollo del ejercicio, es decir; consideramos que las variables FechaCorte, SBAN ,FechaVinculacionBAC y EstadoClienteDesc son variables que podemos precindir a lo largo de nuestro ejercicio, puesto que la fecha de corte es un registro que mide únicamente un dato en el cual tenemos la fecha de extracción de la base de datos, el SBAN es el número de oficina en la cual se encuentra nuestro cliente, FechaVinculacionBAC que es la fecha de apertura del primer producto, también irrelevante en nuestro análisis y por último EstadoClienteDesc, el cual contiene una única observación siendo esta el estado vigente del cliente, el cual se puede obviar puesto que el ejercicio únicamente se hará con clientes vigentes.

Luego del análisis mencionado anteriormente se realizó el tratamiento de valores nulos, en el cual prescindimos de los valores nulos de las variables categóricas debido a que

con respecto a la muestra total no se evidenciaba una reducción significativa de la muestra y también nos encontramos con información personal que el cliente no estuvo dispuesto a diligenciar. Respecto a las variables numéricas, se decidió imputar estos valores con 0, puesto que es usual encontrar en las instituciones financieras que, al preguntar valores financieros, estos clientes no conocen la suma o no la quieren reportar pensando en que esto puede afectar la solicitud de la adquisición de productos.

Luego del análisis anterior, se realizaron gráficas que permitieran conocer la distribución de nuestros clientes y así obtener un conocimiento sobre los valores atípicos que encontramos en cada una de las variables. Sin embargo, a través de este análisis pudimos evidenciar que la base de datos obtenida no era homogénea y contenía demasiados valores atípicos los cuales fueron identificados a través del método del rango intercuartílico y posterior a ello, se transformaron en valores nulos, para su posterior eliminación, lo cual repercutió en una distribución de la base de datos de 500000 registros a 253590 y 25 variables, debido a que el gráfico boxplot nos permitió identificar variables que únicamente contenían valores de 0.

Se realiza por último un cambio de las variables categóricas, para poder entrenar los modelos adecuadamente. Por ejemplo, a los datos de masculino y femenino, se le asigna un valor numérico 1 o 0.

```
#Se realiza un cambio de las variables categóricas para poder aplicarlo en el entrenamiento del modelo.
df3['SexoDesc']=df3['SexoDesc'].map({'FEMENINO':1, 'MASCULINO':0})
```

Figura 1. Cambio de datos categóricos

2. Estrategias de validación y selección del modelo

Se realiza una selección de variables con el objetivo de determinar el número de productos de captación que debería tener un cliente, basados en sus datos demográficos y financieros, ejemplo, edad, estado civil, ingresos, egresos, activos, pasivos, entre otros.

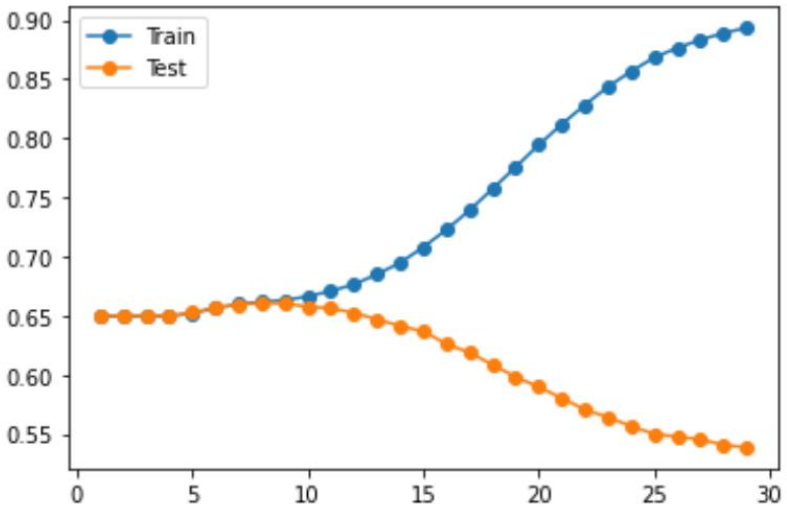
Se evalúan tres modelos de clasificación, en los que se encuentran árboles de decisión, *Random Forest* y una regresión logística. Se entrena cada uno de los modelos con el 70% de los datos. Los hiper parámetros inicialmente se dejan por defecto para cada modelo, pero se evaluarán más adelante.

En la siguiente figura se observan los resultados obtenidos para cada uno de los modelos, evaluados sobre sus propios datos de prueba. Esto nos indica que los dos modelos se ajustan muy bien, pero esto podría ser un sobreajuste a los datos de entrenamiento, por ende, se debe validar en los datos de prueba.

```
Accuracy of the Decision Tree Classifier:  0.999735230659163
Accuracy of the Random Forest Classifier:  0.9997239638787019
Accuracy of the Logistic Regression Classifier:  0.6491355562691183
```

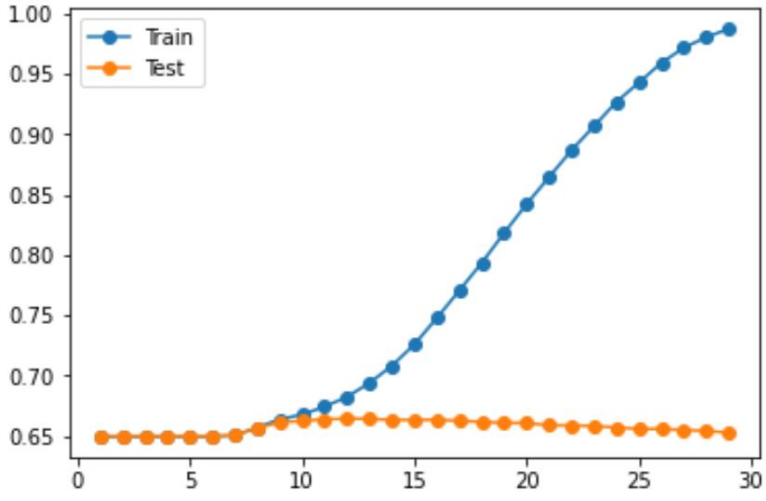
Figura 2. Resultados obtenidos modelos entrenados

Se realiza ahora una validación de los hiper parámetros, en esta ocasión se valida el parámetro *max_depth y* criterio (Gini o Entropía). Este procedimiento se realiza para los modelos de árbol de decisión y *Random Forest.* Se obtienen los resultados que se muestran a continuación.




```
>1, train: 0.649, test: 0.650
>2, train: 0.649, test: 0.650
>3, train: 0.649, test: 0.650
>4, train: 0.649, test: 0.650
>5, train: 0.652, test: 0.652
>6, train: 0.656, test: 0.657
>7, train: 0.660, test: 0.659
>8, train: 0.662, test: 0.661
>9, train: 0.663, test: 0.660
>10, train: 0.666, test: 0.658
>11, train: 0.671, test: 0.656
>12, train: 0.676, test: 0.652
>13, train: 0.685, test: 0.647
>14, train: 0.695, test: 0.641
>15, train: 0.708, test: 0.637
>16, train: 0.723, test: 0.626
>17, train: 0.739, test: 0.619
>18, train: 0.757, test: 0.609
>19, train: 0.776, test: 0.598
>20, train: 0.794, test: 0.590
>21, train: 0.812, test: 0.580
>22, train: 0.828, test: 0.571
>23, train: 0.844, test: 0.564
>24, train: 0.857, test: 0.557
>25, train: 0.868, test: 0.550
>26, train: 0.876, test: 0.547
>27, train: 0.883, test: 0.545
>28, train: 0.889, test: 0.541
>29, train: 0.893, test: 0.539
```

Figura 3. Resultados obtenidos de variación de hiper parámetros para el modelo de árbol de decisión.




```
>1, train: 0.649, test: 0.650
>2, train: 0.649, test: 0.650
>3, train: 0.649, test: 0.650
>4, train: 0.649, test: 0.650
>5, train: 0.649, test: 0.650
>6, train: 0.649, test: 0.650
>7, train: 0.650, test: 0.651
>8, train: 0.657, test: 0.656
>9, train: 0.664, test: 0.661
>10, train: 0.668, test: 0.662
>11, train: 0.674, test: 0.664
>12, train: 0.682, test: 0.664
>13, train: 0.694, test: 0.664
>14, train: 0.708, test: 0.663
>15, train: 0.726, test: 0.663
>16, train: 0.748, test: 0.663
>17, train: 0.770, test: 0.662
>18, train: 0.793, test: 0.661
>19, train: 0.818, test: 0.661
>20, train: 0.842, test: 0.661
>21, train: 0.864, test: 0.659
>22, train: 0.887, test: 0.658
>23, train: 0.907, test: 0.658
>24, train: 0.927, test: 0.657
>25, train: 0.943, test: 0.656
>26, train: 0.959, test: 0.656
>27, train: 0.972, test: 0.655
>28, train: 0.980, test: 0.654
>29, train: 0.987, test: 0.652
```

Figura 4. Resultados obtenidos de variación de hiper parámetros para el modelo de Random Forest

Se obtiene que el hiper parámetro de Gini y entropía no tienen una gran influencia sobre el modelo, pero el de max_depth si es necesario controlarlo para evitar un sobreajuste del modelo, como se observa en las figuras, con valores mayores a 10 de max_depth, el

clasificador empieza a sobreajustar. Es necesario realizar esta validación con una mayor cantidad de hiper parámetros y así validar hasta que valor máximo de exactitud se puede llegar.

Se selecciona el modelo de RandomForest para la construcción con un max_depth de 10 que es dónde el modelo tiene la máxima exactitud antes de empezar a sobre ajustar.

3. Construcción y evaluación del modelo seleccionado

Se realiza la construcción del modelo seleccionado, con los hiper parámetros encontrados anteriormente, el criterio de selección del modelo para este caso fue la exactitud.

```
RFpredfinal = RandomForestclassifier.predict(X_test)
cmRFF = confusion_matrix(Y_train,RFpred)
print('Matriz de confusión')
print(cmRFF)
print()
print('Métricas del modelo')
print(classification_report(Y_test, RFpredfinal))
```

```
Matriz de confusión
[[ 26895     17      2      0]
 [     7 115246      6      1]
 [     2     12  33947      0]
 [     0      1      1   1376]]

Métricas del modelo
              precision    recall  f1-score   support

         0.0       0.41      0.11      0.17     11596
         1.0       0.68      0.90      0.78     49434
         2.0       0.47      0.27      0.35     14489
         3.0       0.00      0.00      0.00       558

    accuracy                           0.65     76077
   macro avg       0.39      0.32      0.32     76077
weighted avg       0.60      0.65      0.60     76077
```

Figura 5. Resultados obtenidos del modelo de Random Forest entrenado con los hiper parámetros obtenidos en el numeral anterior.

Como se observa en la figura se obtiene una exactitud de 65% y el máximo valor de F1 Score para el valor de 1 es de 78%. Es posible evaluar mayor cantidad de hiper parámetros y modelos alternativos para validar si es posible obtener un mayor ajuste.

4. Conclusiones

- Es necesario realizar una evaluación de los modelos con mayor cantidad de hiper parámetros, y determinar si es posible mejorar los resultados obtenidos.
- Se observa que basados en los datos demográficos y financieros, se podría tener una indicación del número de productos, y así realizar proyecciones de crecimiento y perfilación de clientes.

5. Referencias

1. Lapinlampi, G. (s. f.). *AI recommender systems for banking*. Creating purposeful technology |Tietoevry. https://www.tietoevry.com/en/blog/2022/05/unlock-new-revenue-by-personalizing-your-customer-approach-with-ai-recommender-systems/
2. Oyebode, O. y Orji, R. (2020). A hybrid recommender system for product sales in a banking environment. *Journal of Banking and Financial Technology*, *4*(1), 15–25. https://doi.org/10.1007/s42786-019-00014-w