Reporte Técnico de Modelo de Clasificación

Universidad Tecnológica Nacional

Clase de Ciencia de Datos 2023 – Segundo Cuatrimestre

Acosta, Bruno

Bartolomeo, Agustin

Introducción y Objetivos

Un banco quiere predecir cuantos clientes se suscribirán a una campaña de marketing. Para ello, comparten un data set con una cartera de clientes de 45,211 personas con 17 variables que muestran algunas características de los clientes en el banco.

Se realizará un estudio profundo del data set, para entender con que datos estamos trabajando y cuales son las tendencias de sus datos; seguido de una transformación de las variables. Con el data set transformado, se entrenarán varios modelos de machine learning con el objetivo de encontrar el que predice si un cliente se va a subscribir o no de la mejor manera posible.

Descripción del data set

El data set provisto por el banco tiene las siguientes variables:

1. Age: Edad del cliente
2. Job: Tipo de empleo del cliente
3. Martial Status: Estado civil
4. Education: Educación máxima alcanzada por el cliente
5. Credit: Si tiene deuda de crédito o no
6. Balance (euros): Promedio de saldo en la cuenta en el año
7. Housing loan: Si tiene seguro de hogar o no
8. Persona loan: Si tiene prestamos o no
9. Contact: Tipo de contacto del cliente (celular o teléfono)
10. Last Contact Day: Ultimo día de contacto con el cliente en el mes
11. Last Contact Month: Ultimo mes de contacto con el cliente en el año
12. Last Contact Duration: Duración del último contacto con el cliente medido en segundos
13. Campaign: Cantidad de contactos al cliente durante esta campaña, incluye el último contacto.
14. Pdays: Cantidad de dias que pasaron del último contacto con el cliente de una campaña anterior. -1 significa que no hubo contacto previo
15. Previous: Cantidad de contactos previos a esta campaña para cada cliente
16. Poutcome: Performance de la campaña de marketing anterior para este cliente
17. Subscription: Si el cliente accede a la campaña (1) o no (0).

La última variable, **Subscription**, es la variable objetivo que se tiene que predecir.

Análisis Exploratorio de Datos (EDA) y Transformación de Datos

Tal como el banco dijo, el data set entregado tiene 45,211 filas con 17 columnas.

Entendamos primero que nada que tan completos están los datos, con un chequeo de filas nulas o NaN de todas las columnas

Aunque a primera vista parezca que hay una poca cantidad de filas nulas, los NaN están esparcidos por todo el data set, siendo en total 34,581 filas con algún valor NaN.

Sobre la variable objetivo, resultará muy importante a la hora de definir los modelos

entender la proporción de clientes que se subscriben en este data set.

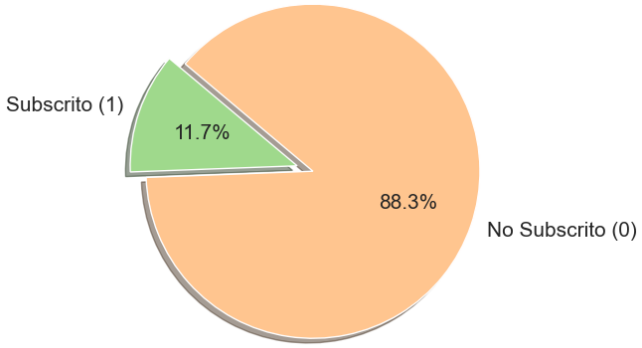

Ilustración 1. Proporción de clientes que acceden a la subscripción

Otro hecho a destacar en el análisis exploratorio de datos, es que no hay dos variables numéricas que se relacionen entre sí.

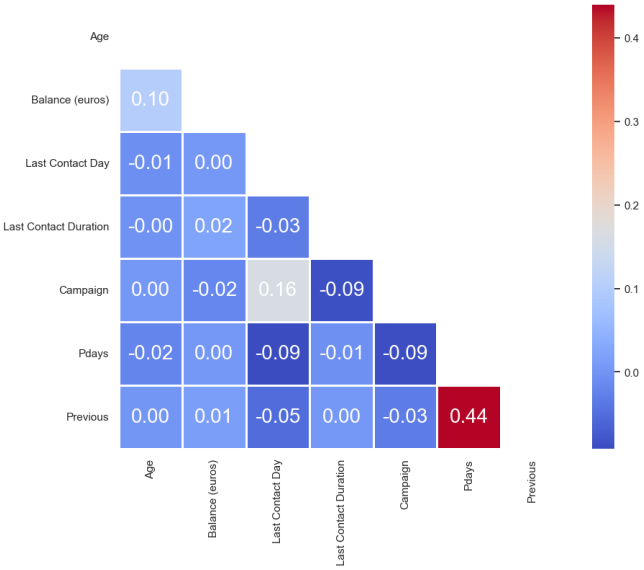

Ilustración 2. Correlación entre variables

Pdays y Previous tienen la correlación mas alta debido a que si al cliente no se lo contactó en una campaña anterior, valen -1 y 0 respectivamente, pero una vez que valen algo distinto, no tienen mucha correlación.

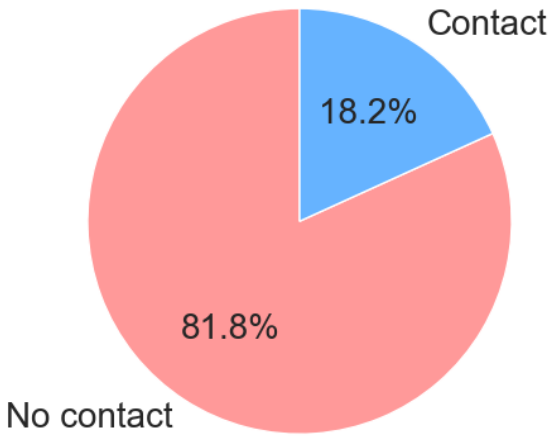

Ilustración 3. Porcentaje de clientes contactados en la campaña anterior

Como a más del 80% de los clientes no se los contactó, termina generando una falsa correlación entre ambos. Esto se corrige en la transformación de datos, para no entregar al modelo el mismo dato dos veces.

Para no perder todas las filas, se decidió reemplazar los valores NaN por la media en las columnas numéricas continuas Age, Balance y Last Contact Duration.

Luego de quitar los NaN, se quitaron outliers extremos en las columnas de Balance y Last Contact Duration, con el objetivo de alimentar al modelo con datos normales y mejorar el resultado. Esto significa que estamos trabajando con **15521 filas**.

Posterior a esto, se transformaron los datos con métodos de agrupación y One-Hot Encoding. Estos métodos terminan generando columnas numéricas adicionales, llegando a **50 columnas en total**.

Materiales y Métodos

Dado que la variable Y que tenemos que predecir es una cualitativa y no una cuantitativa, se debe usar el método de clasificación.[1]

La variable objetivo (Subscription) es una de dos clases: Clase 0 (No se subscribió) y Clase 1 (Se subscribió).

Usando el mapa de ayuda de Scikit Learn para elegir el modelo optimo de machine learning, sumando un modelo extra visto en la cursada, se eligieron los siguientes modelos:

1. Linear SVC
2. KNeighbors
3. SVC
4. Logistic Regression

1. Linear SVC

Empezamos con la primera recomendación de Scikit, que implementa SVC con el kernel 'linear', pero con LibLINEAR en vez de LibSVM. Esto significa que funciona mejor para un data set mayor a 10 mil registros, como es nuestro caso. Consideramos que es un buen punto de inicio para luego comparar los otros modelos.

El modelo de Support Vector Machines[2] básicamente determina la clase de un array de features según su función lineal $W^T x + b$. Si la función es positiva, es una clase, y si es negativa, es la otra.

2. KNeighbors

El modelo de k – Nearest Neighbors[3] emplea un acercamiento fundamentalmente distinto al de Support Vector Machines.

En KNN, el modelo asigna una clase a los datos según los datos 'vecinos'. Esto significa que, para cualquier x sin etiqueta, reúne a sus vecinos con etiqueta conocida más cercanos según la coincidencia o similitud en atributos, y le asigna a x la clase según la que predomina en estos vecinos.

3. Support Vector Classifier

El modelo de Support Vector Machines ya fue introducido en el primer modelo de Linear SVM. Sin embargo, el linear es una variación de SVC y sin la obligación de usar el kernel lineal, puede clasificar los datos de otra manera que no sea con un híper plano lineal. Admite entonces clasificar los datos con una función de base radial o polinómica, por ejemplo.

4. Logistic Regression

Por último, tenemos el modelo de regresión logística. A diferencia de los modelos anteriores, Logistic Regression[4] no clasifica directamente con un *Yes* o un *No*, sino que asigna una probabilidad de que Y pertenezca a una clase.

De manera predeterminada, si la probabilidad supera 0.5, se clasificará como una clase, y sino se clasificará como otra. Sin embargo, este modelo permite desplazar el límite, cambiando la clasificación final dependiendo de si se quiere ser conservativo o más permisivo. El umbral de decisión se ajustará de acuerdo con las necesidades y objetivos específicos de cada problema.

Principal Component Analysis (PCA)

El análisis de componentes principales es una técnica de reducción de dimensionalidad que transforma un conjunto de datos en un nuevo sistema de coordenadas, buscando maximizar la varianza y expresar la información original en términos de componentes principales no correlacionados.

Se utiliza para simplificar la representación de datos complejos manteniendo la mayor cantidad posible de su variabilidad.

Luego de definir el mejor modelo, utilizaremos PCA para ver si se mejora el rendimiento.

Métricas a utilizar

Para el scoring de los modelos se utilizará el score y la curva AUC ROC, junto con la especificidad y la sensibilidad.

Experimentos y Resultados

Punto de partida

Los datos a partir de este momento fueron escalados con el método de StandardScaler para mejorar los resultados de todos los modelos.

Antes de comenzar con los 4 modelos previamente descriptos, aplicamos al data set un simple SVC:

```
#Defino el modelo
basic_svc = svm.SVC(probability=True)
# Ajusto mi modelo a las muestras de training
basic_svc.fit(xtrain_scal, ytrain)
```

Los resultados son visiblemente malos:

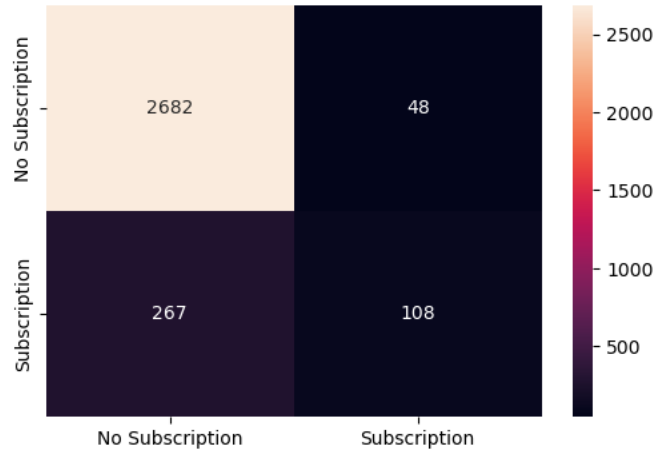

Ilustración 4. Matriz de confusión de un modelo básico de SVC

Tiene una especificidad de 0.98, pero tiene un puntaje AUC ROC de 0.64 y una sensibilidad de 0.288. Esto se debe a la naturaleza del data set: Su desbalance de la variable objetivo Y, tal como se descubrió en el EDA. Al haber mayoría de negativos, el modelo es recompensado por clasificar la mayoría de los datos como negativos, sean verdaderos o falso negativos.

Ajustes al modelo según los datos

Para que el modelo entienda este hecho sobre los datos, resulta clave especificar un peso de clases 'balanced' (balanceado). Este es un parámetro que automáticamente ajusta el peso de las clases de manera inversamente proporcional a la frecuencia de cada clase en el data set.

No es la única medida que se toma para el resto de los modelos: Se buscan los mejores parámetros de cada modelo con Grid Search Cross Validation, y para asegurarse que cada fold tenga una proporción de variable objetivo similar a la del data set original, se introduce la estratificación.

Machine Learning

Con estas premisas, se entrenaron los 4 modelos descriptos previamente, obteniendo los siguientes resultados:

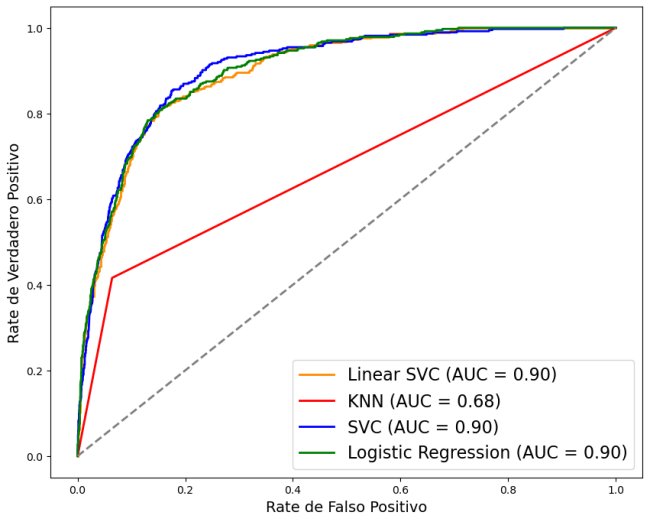

Ilustración 5. Curva AUC ROC de los 4 modelos

Empezando por KNN, es el único modelo que no soporta el parámetro de *class_weight*, y aun con la estratificación y un amplio grid search, se obtuvieron resultados muy similares a los de el SVC básico.

Los otros tres modelos tuvieron un desempeño muy similar, con un score AUC ROC de 0.90 para los 3. Es entonces muy importante ver su desempeño en las otras dos métricas:

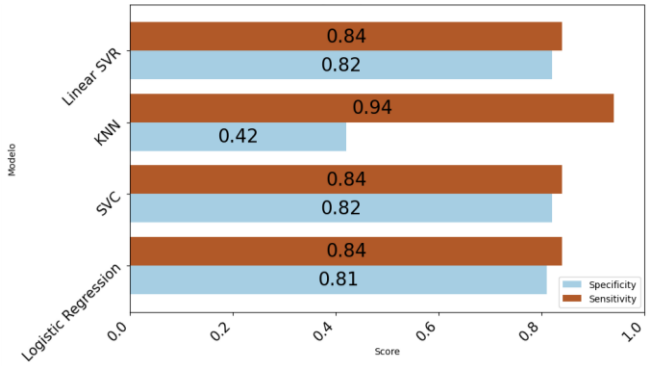

Ilustración 6. Sensibilidad y Especificidad de los 4 modelos

Los 3 modelos tienen un desempeño extremadamente similar, por lo cual se podrían usar cualquiera de los 3. Ante esta realidad, se elige SVC sobre el resto, al mostrar una leve mejora en la curva AUC ROC y al tener los mejores valores de sensibilidad y especificidad. Vale remarcar que se podrían utilizar los otros dos modelos para tener más validación y certeza acerca de las predicciones.

Aplicación de PCA previo a Machine Learning

Luego de definir un modelo y tener valores de referencia, se probó la implementación de PCA antes de entrenar el modelo de SVC.

Se redujo de 49 a 30 features. Estas features nuevas explican la varianza de la siguiente manera:

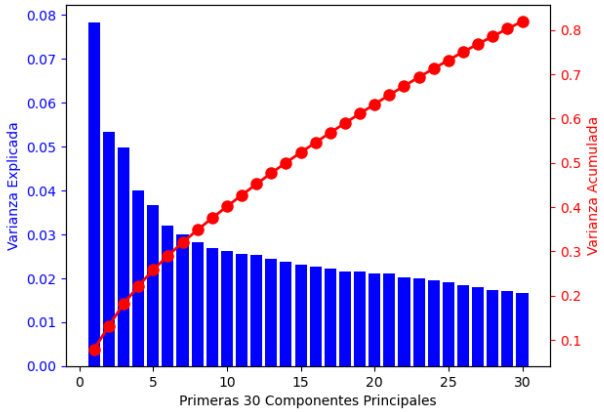

Ilustración 7. Varianza explicada y acumulada de los 30 componentes

Podemos ver que no se pudieron obtener 2 o 3 features principales que expliquen la gran mayoría de la variabilidad de los datos, recién llegando a un 80% de varianza en la columna 30.

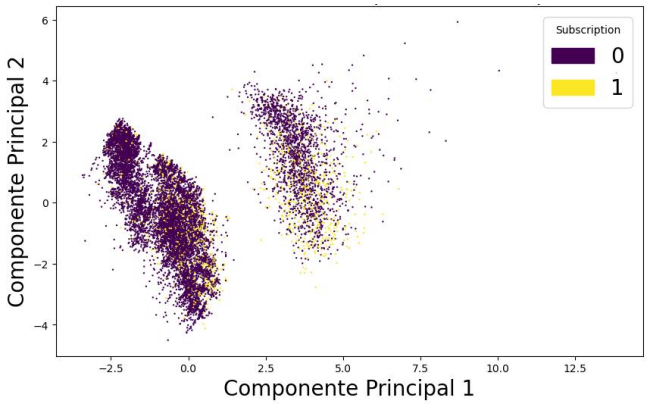

Ilustración 8. Scatter Plot de los 2 componentes principales

Un scatter plot de las dos principales variables confirma que PCA en este caso no ayuda a la visualización de los datos. A continuación vemos los resultados de la aplicación del modelo de SVC utilizando de 1 a 30 features post PCA.

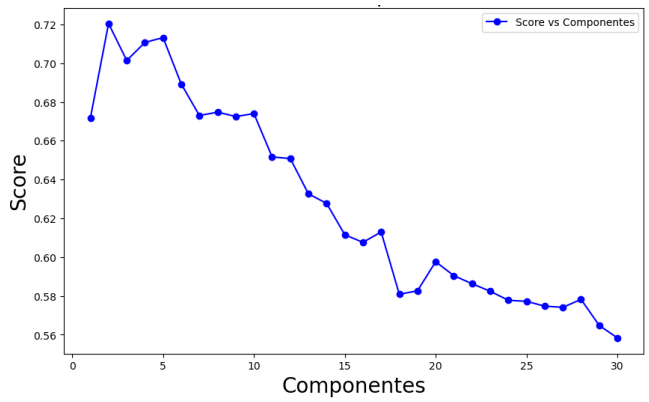

Ilustración 9. Score AUC ROC del modelo con relación a cuantos componentes se utilizaron

Discusión y Resultados

A partir de los datos proporcionados, podemos concluir que es difícil predecir con una alta seguridad si un cliente accederá o no a subscripción.

Se consiguieron buenos resultados con varios modelos, especialmente con SVC, pero sigue teniendo una sensibilidad y especificidad apenas superior a 80%.

Dependerá del usuario de este modelo donde colocará su umbral de decisión, dependiendo de si quiere obtener una lista de potenciales clientes que accederán a la subscripción aunque una gran parte sean falsos positivos; o estar seguro de cuales son los clientes que con máxima certeza accederán a la subscripción, aunque no signifiquen la totalidad de los mismos.

Nos pareció sumamente interesante el trato diferencial que se debió tener con este set de datos debido al desbalance que se presenta en la variable objetivo, y nos dirige a poner un enfoque más grande en los parámetros de futuros modelos.

Concluimos también que en este caso no es conveniente aplicar PCA. Esto se puede deber a que PCA asume linealidad en los datos, y el data frame utilizado tiene muchas variables binarias.

Referencias

[1] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani and Jonathan Taylor (2023). 4. Classification. An introduction to Statistical Learning with Applications in Python (pp 135).

[2] Ian Goodfellow, Yoshua Bengio y Aaron Courville (2016). 5.7.2 Support Vector Machines. Deep Learning (pp 139).

[3]Natasha Latysheva (2016). k-nearest neighbor algorithm using Python. Data Science Central [En Linea]. Disponible en: https://www.datasciencecentral.com/k-nearest-neighbor-algorithm-using-python/

[4]Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani and Jonathan Taylor (2023). 4.3. Logistic Regression. An introduction to Statistical Learning with Applications in Python (pp 138).