UNIVERSIDAD DE ANTIOQUIA

Caso de estudio - Aprendizaje Supervisado
Analítica para la toma de decisiones

Alanis Álvarez, Juan E. Soto, Paola A. Arabia, Santiago Restrepo
Departamento de Ingeniería Industrial
Universidad de Antioquia, Colombia

Introducción

El presente informe presenta el desarrollo de un proyecto de Machine Learning el cual se centra en abordar una problemática específica en un e-commerce relacionada con la predicción del comportamiento de sus clientes. El objetivo de este proyecto es desarrollar y evaluar modelos de aprendizaje supervisado que puedan predecir si un cliente potencial realizará una compra o no en el sitio web de e-commerce.

A lo largo del desarrollo del proyecto, este se dividió en dos notebooks de jupyter: uno dedicado a la exploración y preprocesamiento de los datos, seguido por la implementación y evaluación de 4 tipos de modelos de aprendizaje supervisado, donde finalmente se plantearán recomendaciones y conclusiones derivadas del análisis realizado en el proyecto.

Diseño de solución propuesto

Para la empresa asociada al problema (E-corp) es fundamental tener un sistema de información que le permita centrar sus esfuerzos en conseguir nuevos clientes y diferenciar quienes realmente son sus clientes potenciales. Para ello, brindamos una solución centrada en el montaje de un modelo de aprendizaje supervisado, específicamente para clasificar a los visitantes del sitio web como posibles clientes o no, basado en el análisis de algunas características del lead que navega por la pagina web. De esta forma, la empresa podrá contar con información clara que le permita destinar recursos humanos y monetarios en diferentes estrategias como por ejemplo, el marketing.

Limpieza y transformación de los datos

Inicialmente, cabe mencionar que el dataframe original contiene 12.330 observaciones y 18 columnas (variables), de las cuales 7 son tipo *float*, 7 de tipo *int*, 2 de tipo *object* y 2 de *bool*, y que además, ninguna de ellas posee datos nulos.

Ahora, en lo que respecta a la limpieza de los datos es importante mencionar que el dataset posee 125 observaciones duplicadas, las cuales son eliminadas del conjunto de datos. Por otra parte, gracias al análisis realizado de los datos se encuentra que la mayoría de las variables numéricas poseen datos atípicos o muy alejados de la media, y a partir de esto se eliminan 5 columnas (*'Informational', 'Informational_Duration', 'Reviews_Duration', 'BounceRates' y 'ProductRelated_Duration'*) dado que poseen una gran cantidad de datos atípicos y no generan explicabilidad relevante al modelo. Así mismo, las variables numéricas que se mantienen son tratadas a través de la eliminación de los datos que se encuentren por encima del límite del tercer cuartil. Finalmente, en lo que respecta a la transformación de variables se realiza categorización a 5 variables de tipo *int* para que sean de tipo *object* dado que contarían

con pocas categorías y puede ser útil su análisis posterior, pero en el caso de la variable *'PageValues'* se realiza un cambio de su tipología *float* a *int* de modo que los valores que estén por debajo de 20 tomen el valor 0 y los valores mayores o iguales a 20 tomen el valor de 1, y que así sea más fácil de tratar, finalmente, su nombre se cambia a 'PageValues>=20'.

Una vez realizada la limpieza y transformación de los datos se obtiene un dataset con 10.570 observaciones (se eliminan alrededor del 15% de las observaciones) y 13 columnas, de las cuales 7 son de tipo *object*, 5 son de tipo *int* y 1 es de tipo *float*.

Análisis exploratorio de los datos

Inicialmente, para el análisis exploratorio de los datos se toma como insumo principal la matriz de correlación en la cual se evidencia a simple vista que existen algunas variables independientes que se correlacionan entre ellas, lo que generaría un problema de multicolinealidad (principalmente las variables de tipo 'Duration'), y para ello se realiza la eliminación de una de las variables que poseen correlación. Por otra parte, la variable ('PageValues') posee una correlación positiva considerable (0.49) pero con la variable objetivo ('Purchase') la cual se mantiene dentro del conjunto de datos con el tratamiento que se mencionaba anteriormente. Así mismo, la mayoría de estas variables que se eliminan poseen datos muy lejanos a la media, los cuales se pueden observar en los boxplots realizados tanto para el análisis univariado como bivariado, además, se decide realizar boxplots dado que se desea comparar una variable categórica ('Purchase') con variables numéricas.

Finalmente, se evidencia que el conjunto de datos posee una variable objetivo con clases desbalanceadas, ya que en ella (variable 'Purchase') la categoría 0 (no compradores) representa alrededor del 85% de las observaciones, mientras que el 15% restante corresponde a la categoría 1 (compradores).

Preparación de los datos

Para la preparación de los datos, una vez realizado el preprocesamiento y análisis exploratorio de ellos, se consolida un dataset mucho más limpio y adecuado para proceder con el montaje de los modelos de aprendizaje supervisado, donde dicho dataset es llamado '*df_sin_atipicos*', el cual es el insumo principal para la selección de variables y para la aplicación de los modelos. Ahora, el dataset se lleva inicialmente a formato dummy ('*df_dummies*') pasando de 13 a 75 columnas, luego se realiza la separación de los datos para train (80%) y para test (20%), y finalmente se escalan los datos con el método *MinMaxScaler()* para que se encuentren en el formato adecuado para la ejecución de modelos.

Selección de variables

Ahora, para la selección de variables se decide utilizar el método *SelectFromModel*. Dicho método integrado es seleccionado dado que puede llegar a ser más preciso que un método de filtrado y además, puede ser más riguroso en la selección de las características utilizando un modelo de Machine Learning que evita el overfitting. Para su implementación se decide evaluar 6 alternativas, de las cuales 3 implementen el método Lasso variando su valor alpha

en cada una de ellas y las otras 3 implementando un modelo de RandomForest variando el threshold en cada uno de ellos, lo que da como resultado 6 diferentes conjuntos de datos.

Finalmente, para validar cuál será el conjunto de datos a escoger se implementa un modelo de regresión logística tradicional con los 6 datasets y se define (por recomendación de un experto) que el conjunto de datos que obtenga como resultado un f1 score y un recall mayor será el dataset a utilizar en el despliegue de los modelos.

Selección y aplicación de algoritmos/ técnicas de modelado

Los algoritmos o modelos de aprendizaje supervisado utilizados para ejecutar el proyecto fueron:

- **Regresión logística:** Se decide utilizar regresión logística dado que es el modelo básico y además suele tener una buena precisión cuando se intenta predecir variables dicotómicas (sí o no) como es en este caso.
- **Random forest:** El modelo de random forest fue considerado por su capacidad para manejar conjuntos grandes de datos y su resistencia al sobreajuste, ya que es un modelo más robusto que se compone de varios árboles de decisión lo que hace que no sea tan sensible a valores atípicos.
- **Gradient Boosting Machine:** Este modelo parece ser flexible y adecuado para manejar cualquier tipo de datos, además también es un buen modelo para evitar el sobreajuste.
- **Support Vector Machine:** Se decide utilizar este tipo de modelo dado que al implementar un kernel de transformación de características (o hiperplano) podría obtener resultados interesantes, además es un modelo que evita el sobreajuste.

Comparación y selección de técnicas

Para la ejecución de los diferentes técnicas de aprendizaje supervisado se desarrollaron 3 modelos con cada uno de ellos, donde el primero se ejecuta con los datos resultantes del preprocesamiento, el segundo con el conjunto de datos resultante de la selección de características y el último de ellos con los parámetros del modelo optimizados, es decir, que se emplearon un total de 12 modelos.

Ahora, para realizar la comparación entre los modelos se imprime el classification report (el cual arroja accuracy, precision, recall y f1 score) tanto para train como para test, la matriz de confusión y la curva ROC con su respectivo valor AUC. La información arrojada por las herramientas mencionadas permite cuestionar el comportamiento de cada uno de los modelos empleados, donde evidentemente se busca que las métricas de desempeño, el valor AUC, la cantidad de verdaderos positivos y verdaderos negativos se maximicen, mientras que se reduce la cantidad de falsos positivos.

Afinamiento de hiperparámetros

Para realizar la optimización de los hiperparámetros de los diferentes modelos se utiliza el método de búsqueda aleatoria ya que es la menos pesada computacionalmente hablando. Para ejecutar la optimización, inicialmente se hace uso de la función '*get_params()*' de sklearn

para conocer los parámetros que emplea el modelo por defecto. Posteriormente, se acude a la documentación de Machine Learning disponible en la página de sklearn para conocer los parámetros que se pueden optimizar en cada tipo de modelo, y finalmente, se definen una serie de opciones para los parámetros a través de un método de tanteo buscando que el modelo arroje las mejores métricas de desempeño posibles.

Evaluación y análisis del mejor modelo

Una vez se ejecutaron los modelos se obtienen las siguientes métricas de desempeño:

Modelo		Entrenamiento (train)				Prueba (test)			
		Accuracy	*Precision	*Recall	*F1 score	Accuracy	*Precision	*Recall	*F1 score
Regresión logística	Base	0.8512	0.73	0.58	0.60	0.8623	0.75	0.58	0.61
	Select From Model	0.8512	0.73	0.58	0.60	0.8623	0.75	0.58	0.61
	Optimizado	0.8507	0.73	0.58	0.59	0.8623	0.75	0.58	0.61
Random Forest	Base	1	1	1	1	0.9058	0.84	0.76	0.79
	Select From Model	1	1	1	1	0.9011	0.83	0.75	0.78
	Optimizado	0.9320	0.92	0.81	0.85	0.9077	0.86	0.74	0.78
Gradient Boosting Machine	Base	0.9184	0.86	0.82	0.84	0.9077	0.83	0.79	0.81
	Select From Model	0.9184	0.86	0.82	0.84	0.9077	0.83	0.79	0.81
	Optimizado	0.9187	0.87	0.81	0.84	0.9124	0.84	0.80	0.82
Support Vector Machine	Base	0.8559	0.79	0.57	0.58	0.8599	0.75	0.56	0.57
	Select From Model	0.8559	0.79	0.57	0.58	0.8604	0.76	0.56	0.57
	Optimizado	0.8450	0.70	0.57	0.59	0.8571	0.71	0.59	0.61

Tabla 1. Métricas de desempeño de los modelos de aprendizaje supervisado empleados

(*): cabe mencionar que para las métricas precision, recall y f1 score se reporta el valor 'macro avg' que arroja el classification report para no poner los valores tanto de la clase positiva como negativa.

Ahora, gracias a los resultados obtenidos que se aprecian en la Tabla 1 se concluye que el modelo que obtuvo las mejores métricas fue el Random Forest optimizado (no se toma el Random Forest con SelectFromModel dado que consideramos que tiene sobreajuste). Sin embargo, decidimos escoger como mejor modelo el **Gradient Boosting Machine optimizado** dado que sus métricas tanto en train como en test son muy buenas, similares y

consistentes, además, es el modelo que mejores métricas arroja en test maximizando tanto el recall como el f1-score que son métricas relevantes, ya que se centran en la identificación de falsos negativos y los falsos positivos, por ende, dicho modelo nos permite obtener predicciones mucho más correctas.

Ahora, de su matriz de confusión y su curva ROC (ver Ilustración 1 y 2) se logra apreciar que el modelo logra realizar una diferenciación adecuada de las clases tanto positivas como negativas, ya que logra predecir en una gran proporción los verdaderos negativos y los verdaderos positivos, y así mismo, se logra disminuir notablemente la presencia de falsos positivos y falsos negativos. En cuanto al valor AUC (0.80) de la curva ROC, al no ser realmente una curva si no un vector pues el valor del área bajo la curva será menor pero como vimos, este modelo posee la capacidad de diferenciar eficientemente las clases de la variable objetivo.

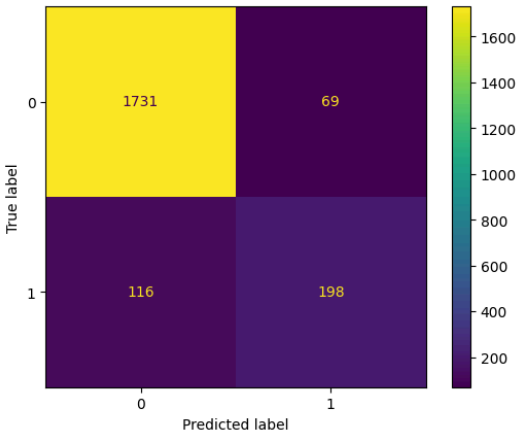

Ilustración 1. Matriz de confusión para modelo GBM optimizado

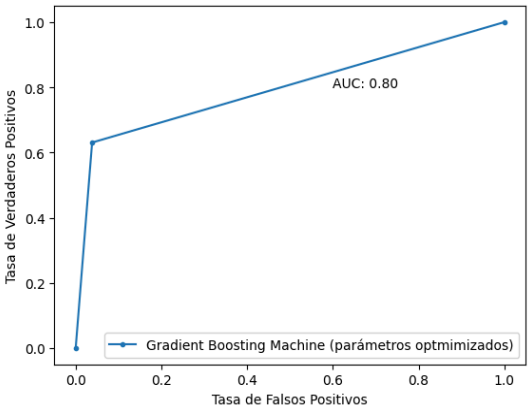

Ilustración 2. Curva ROC para modelo GBM optimizado

Finalmente, el correcto rendimiento del modelo Gradient Boosting Machine se atribuye a su composición, dado que es un modelo basado en árboles de decisión el cual se enfoca en corregir los errores del árbol anterior, lo que lo convierte en un modelo realmente preciso y conveniente para cualquier tipo de problema que se trate.

Conclusiones finales y recomendaciones

Existen diferentes tipos de modelos que se pueden emplear dependiendo de la situación, así mismo, algunos obtienen rendimientos superiores a los otros. La escogencia del modelo se debe a la realización de diferentes pruebas para la elección del modelo más óptimo. Si bien el **Gradient Boosting Machine optimizado** se destacó como el modelo más óptimo en este caso, se sugiere a la empresa considerar también el **Random Forest** debido a que también arroja buenos resultados. Se recomienda también, dedicar recursos a la optimización de los parámetros utilizando máquinas más potentes, ya que esto mejoraría el rendimiento de los modelos y conduciría a predicciones más precisas de la clase positiva. Finalmente, los modelos seleccionados demuestran capacidad efectiva para la diferenciación entre clases, lo que mitiga el impacto de la distribución de los datos y genera resultados realmente confiables.

Bibliografía

scikit-learn: machine learning in Python — scikit-learn 1.4.1 documentation. (s. f.). https://scikit-learn.org/stable/