Instituto Tecnológico y de Estudios Superiores de Monterrey

Inteligencia Artificial Avanzada para la Ciencia de Datos (Gpo 101)

Selección, configuración y entrenamiento del modelo

Equipo 5

Jorge Eduardo De León Reyna - A00829759
David Esquer Ramos - A01114940
Francisco Mestizo Hernández - A01731549
Adrián Emmanuel Faz Mercado - A01570770

Agosto 28, 2023

1. Introducción

Una vez que se finalizó con la limpieza de los datos, es posible comenzar con la decisión del modelo que se utilizará. El problema que se presenta es generar un modelo que permita predecir si un tripulante del Titanic sobrevivió o no en base a diferentes características, tales como el sexo, edad, la tarifa que pagó, la clase a la que pertenece, si cuenta con familiares dentro del barco, entre otros. Se trata de un problema de **clasificación binaria,** pues se busca generar un modelo que permita clasificar entre 2 posibles salidas, si el tripulante sobrevivió o si no lo hizo.

En la entrega anterior, se realizó un análisis de las variables presentadas en el set de datos, y las características que permanecieron después de realizar la limpieza de los datos están relacionadas con su persona, el lugar donde embarcó, la clase de su cabina, y cuánto pagó por su boleto.

En esta entrega, se explorarán diferentes modelos que sean compatibles con el problema que se presenta, además de probar diferentes configuraciones para cada uno. El objetivo final es identificar y seleccionar un modelo que ofrezca las predicciones más adecuadas y precisas, tomando en cuenta las métricas seleccionadas.

1.1 Set de datos

El set de datos utilizado es el del titanic, obtenido de la plataforma de Kaggle. Para consultarlo a más detalle se puede consultar en el siguiente link. El set tiene 891 registros y cuenta con las siguientes variables: Passenger ID, Survival, Ticket class (PClass), Sex, Age, Number of siblings / spouses aboard the Titanic (sibsp), Number of parents / children aboard the Titanic (parch), Ticket number, Passenger fare, Cabin number, Port of Embarkation. Estos datos presentan únicamente dos posibles resultados: 1, indicando que la persona sobrevivió, y 0, señalando que no lo logró.

Debido a que el set de datos no está listo para utilizarse, se realizó una limpieza de datos. En esta limpieza se utilizó la técnica de “one hot encoding” para las variables que eran categóricas. Además, existían algunos registros que les faltaban datos como la edad. Basados en su título y su clase se determinó un aproximado de la edad que podrían haber tenido. Para ver con más detalle la limpieza de datos se sugiere visitar esa entrega.

El archivo de datos que se utilizó fue “train_clean.csv”, el cual ya tiene los datos limpios, y se pueden encontrar en la siguiente liga.

1.1.1 Separación de los datos

Para la evaluación y refinamiento de nuestro modelo, se realizó la separación de los datos de entrenamiento en 2 grupos más, uno para entrenar el modelo y el otro para validarlo de manera iterativa y con diferentes parámetros. Cuando el desempeño del subset de datos para validación sea adecuado, se procederá a usar el de prueba original, y así obtener una estimación final de cómo el modelo se desempeñará en datos completamente nuevos.

Para la división, se designó el 80% de los datos de entrenamiento para entrenar el modelo, y el 20% para realizar la validación del mismo.

Datos de entrenamiento *train_clean.csv*	Datos de entrenamiento (80%)
	Datos para validación (20%)
Datos de prueba *test_clean.csv*	Datos de prueba

Los algoritmos de clasificación están especializados en aprender patrones en sets de datos que se encuentran previamente categorizados. Una vez que el modelo está entrenado y detecta estos patrones, se le pueden dar nuevos sets de datos que no estás categorizados y el modelo predecirá a qué categoría pertenecen.

Por esto, es posible utilizar un algoritmo de clasificación para predecir si un pasajero del Titanic sobrevivió o no basado en sus características. Algunos de los modelos de clasificación que existen son la regresión logística, Support Vector Machines, árboles de decisión, Random Forests, Extreme-Gradient Boosting (Wolff, 2022).

2. Métricas

Para la selección del mejor modelo, se utilizará el criterio de "**accuracy**". Esta métrica mide la proporción de predicciones correctas que hizo un modelo en relación con el total de predicciones.

Para calcularla, se utilizan los valores obtenidos de la matriz de confusión, estos valores son los siguientes:

Verdaderos positivos (VP)	Número de casos positivos (1) que el modelo predijo correctamente como positivos.
Verdaderos negativos (VN)	Número de casos negativos (0) que el modelo predijo correctamente como negativos.
Falsos positivos (FP)	Número de casos negativos (0) que el modelo predijo incorrectamente como positivos.
Falsos negativos (FN)	Número de casos positivos (0) que el modelo predijo incorrectamente como negativos.

La fórmula matemática para obtener la "accuracy" o exactitud del modelo es la siguiente:

$$Accuracy = \frac{VP+VN}{VP+VN+FN+FP}$$

Básicamente, consiste en dividir la cantidad total de predicciones correctas entre el número total de predicciones. La elección de esta métrica se debe a que se busca tratar de predecir la mayor cantidad de registros correctamente. Esta métrica se utilizará tanto para el subconjunto de entrenamiento como para el subconjunto de prueba.

3. Definición de los modelos

Para decidir el modelo de clasificación que se utilizará, es importante conocer los datos que utiliza y los resultados que se pueden obtener de ellos. Por esto se listan a continuación las características principales de cada modelo..

- **Regresión logística:** Este modelo puede recibir parámetros categóricos o numéricos, pero el resultado que dará siempre será categórico. El resultado será la probabilidad de que el elemento pertenezca a una categoría, en una escala del 0 al 1.

- **Support Vector Machines:** Este modelo genera hiperplanos que dividen a los datos en categorías. Lo que esté de un lado del plano es de una categoría y lo que esté del otro lado será de la segunda categoría. Es un modelo que se puede adaptar bien a sets de datos grandes ya que es multidimensional.
- **Árboles de decisión:** Este es un modelo que permite hacer una clasificación sencilla, permitiendo tener clases, sin la necesidad de mucha supervisión humana. Este algoritmo funciona tomando decisiones en cada capa, dependiendo del valor que una variable tiene, hasta llegar al final del árbol, donde obtendremos predicción de la clasificación.
- **Random Forest:** Es una expansión de los árboles de decisión. Los Random Forest generan promedios para los datos y así generan las conexiones. Resuelven el problema de los árboles de decisión a forzar que las variables sean categóricas.
- **Extreme-Gradient Boosting (XGBoost):** Modelo que se basa en la idea de construir árboles de decisión de manera secuencial, en donde cada árbol intenta corregir los errores de árboles anteriores. Además, cuenta con herramientas para la regularización, lo que ayuda a prevenir el sobreajuste.

Como se puede observar, cada algoritmo muestra eficiencia variada dependiendo del escenario en el que se aplique. Además, cada uno tiene sus propias restricciones en cuanto a las entradas que acepta o el número de clasificaciones de salida que maneja. Para determinar el modelo que mejor se ajusta a los datos, se realizarán pruebas con 3 de estos algoritmos y se utilizará la métrica de accuracy en los datos de entrenamiento y validación para elegir el más adecuado.

Los modelos seleccionados para realizar pruebas y configuraciones fueron:

- Regresión logística
- Random Forest
- Extreme Gradient Boosting (XGBoost)

4. Pruebas de modelos

Se comienza con la prueba de los tres modelos seleccionados para resolver el problema. Para seleccionar el modelo que se utilizará para resolver el problema, se harán pruebas con diferentes hiper parámetros de cada uno, para que se pueda asegurar que el modelo elegido sea verdaderamente el más efectivo y no uno que por azar dio buenos resultados pero que no modele correctamente los datos.

A continuación se describen los hiper parámetros que se modificarán de cada modelo. Para que el proceso sea más efectivo se utilizará un grid search que seleccionará la mejor combinación de hiper parámetros.

Hiper parámetros para el random forest	
n_estimators	Indica la cantidad de árboles de decisión que se utilizarán.
max_depth	Marca la profundidad máxima que puede tener cada árbol de decisión.
min_sample_split	Determina cuántas muestras se necesitan como mínimo para partir un nodo.
min_sample_leaf	Especifica la cantidad mínima de muestras que debe haber en un nodo final o nodo hoja.
bootstrap	Indica si se usa el *bootstraping* para crear los árboles de decisión individuales.
criterion	Se usa para escoger el método que tendrán los árboles de decisión para hacer las divisiones en cada nodo (gini o por entropía).

Hiper parámetros para la regresión logística	
penalty	Indica la regularización que se aplica al modelo, puede ser Lasso (L1), Ridge (L2) o las dos.

C	También es un hiper parámetro de regularización que indica qué tan fuerte es la regularización que se aplica al modelo.
l1_ratio	Es un parámetro que se usa cuando se tiene un *penalty* de los dos métodos. Controla la proporción para la mezcla del L1 y L2.
max_iter	Debido a que el modelo se resuelve por un método iterativo, este parámetro define cuántas iteraciones se realizan para que el modelo se ajuste a los datos.

Hiper parámetros para la XGBoost	
alpha	Marca la regularización Lasso (L1) para el modelo, dependiendo de la penalización que se de por la función de pérdida.
lambda	Marca la regularización Ridge (L2) para el modelo, dependiendo de la penalización que se de por la función de pérdida.
gamma	Parámetro de regularización que controla la mínima reducción en la pérdida para que se sigan haciendo divisiones en el nodo.
max_depth	Indica la profundidad máxima que puede tener cada árbol de decisión.
min_child_weight	Especifica el peso mínimo que debe tener cada hijo.

Al utilizar el grid search con estos hiper parámetros, se obtendrán tres modelos diferentes con los hiper parámetros que mejor se ajusten a los datos. Estos modelos se compararán más adelante.

5. Análisis y discusión de resultados

Para cada uno de los modelos obtenidos, se entrenaron con los datos de entrenamiento, se realizaron pruebas con los datos reservados para pruebas y finalmente se obtuvo la matriz de confusión y el valor de la exactitud (“accuracy”) para analizar el desempeño de cada uno.

Los resultados obtenidos de cada modelo fueron:

1. Logistic Regression (0.865)
2. Gradient Boosting Classifier (0.8426)
3. Extreme Gradient Boosting Classifier (0.831)

Sin embargo, se decidió que para tener un mejor desempeño, en donde se busque tener una mayor exactitud y precisión con los resultados, sería una buena idea usar una combinación en donde se tomen en cuenta todos los modelos.

Para lograr esta combinación, se utilizó un Voting Classifier, el cual es un método de ensamblaje que combina las predicciones de varios modelos de aprendizaje automático para tomar una decisión final. Esto permite mejorar la precisión y la robustez del modelo.

Después de entrenar al voting classifier se puede ver que tiene más overfitting que el modelo de logistic regression. Esto se observa porque tiene una diferencia más alta entre la precisión de testing y la de training. A pesar de hacer esta prueba, se obtuvieron resultados que no eran mejores que los modelos por separado, por lo tanto se decidió quedarse con el modelo que mejor desempeño individual tuvo.

Por lo tanto, el mejor modelo que se puede utilizar para predecir si una persona sobrevive o no en el Titanic, se obtendrá con la regresión logística, ya que no presenta overfitting sobre los datos y tiene un accuracy por arriba del 85%.

Referencias

Wolff, Rachel. (2022). *5 Types of Classification Algorithms in Machine Learning*. Estados Unidos: Monkey Learn. https://monkeylearn.com/blog/classification-algorithms/