Proyecto Final

Rodrigo García Gorostizaga
Andrés Moguel López Jensen

Inteligencia de Datos

Docente: Luis Zúñiga

8 de mayo

Licenciatura en Actuaría
Primavera 2023
Universidad Iberoamericana

Planteamiento

Pegarle a una pelota de béisbol es una de las cosas más difíciles de hacer en cualquier deporte profesional. El bateador tiene alrededor de 125 milisegundos para decidir si intentará hacer contacto o no con la pelota, es decir producir un *swing.* Se tiene una base de datos de *Statcast* de *Major League Baseball* (MLB) con información sobre lanzamientos de las últimas 3 temporadas de la liga de béisbol profesional de los Estados Unidos. La base de datos se encuentra en Kaggle [1] y es parte de la competencia 'Predict if a pitch will result in a swing'. Se busca crear un modelo que pueda clasificar o identificar si un bateador busca hacer contacto (swing) o no con dicho lanzamiento. Es un problema de clasificación binario; se utiliza Python para el desarrollo de una máquina de vectores de soporte y un bosque aleatorio.

Análisis Exploratorio de Datos

Carga de Datos e Identificación de Variables

Para el desarrollo del modelo se utiliza el archivo de 'train' que se encuentra disponible en la página de Kaggle. Los datos se cargan en un data frame y se procede a hacer un análisis exploratorio de los mismos. La base de datos contiene información de 1,443,276 lanzamientos. La base de datos original consta de 91 columnas (90 variables predictoras y 1 variable de respuesta), la variable respuesta se identifica como "swing". De las 91 variables hay 65 de tipo flotante, 7 de tipo entero y 19 de tipo objeto (cualitativo). Tras consultar el diccionario de datos de Statcast [2], en su página oficial se decide que se mantendrán las siguientes variables: "pitch_type", "release_speed", "release_pos_x", "release_pos_z", "zone", "pfx_x", "pfx_z", "plate_x", "plate_z", "vx0", "vy0", "vz0", "ax", "ay", "az", "effective_speed", "release_spin_rate", "release_extension", "release_pos_y" ,"spin_axis", y por último, la variable respuesta, "swing". Estas variables son las que se consideran como más importantes o relevantes en cuanto a determinar si un bateador intentará hacer contacto con un lanzamiento. Ahora se tienen 20 variables predictoras y la variable respuesta. Todas son de tipo flotante menos la variable "pitch_type" que es de tipo objeto (cualitativa). Estas variables se pasan a un nuevo data frame y el análisis de datos se continua solamente para dichas variables.

Identificación de Valores Nulos y Duplicados

Para desarrollo correcto del modelo se revisa el conjunto de datos para identificar datos nulos, faltantes y duplicados. No se encontraron datos nulos o faltantes y por lo tanto no es necesario ni rellenarlos ni eliminarlos. No se encontraron datos duplicados y por lo tanto no es necesario eliminarlos.

Tratamiento Variable Cualitativa

Como se ha mencionado anteriormente, la variable "pitch_type" es cualitativa. Cuando un bateador identifica o cree identificar que tipo de lanzamiento es, (por ejemplo: recta, curva, slider, cambio de velocidad, entre otras) esto es una parte crucial de su decisión en cuanto a intentar hacer contacto o no a la pelota. Es por esto por lo que se decide mantener la variable y cuantificarla de alguna manera. La base de datos identifica 16 tipos de lanzamientos diferentes. Ya que el tipo de lanzamiento no es una variable jerárquica se opta por hacer un one hot encoding ya que es más eficiente para este caso.

Análisis de la Variable Respuesta

Se busca predecir la variable “swing” la cuál indica si el bateador intento o no hacer contacto con la pelota. Esta variable es binaria; el bateador solamente puede o no intentar hacer contacto o intentar hacer contacto con la pelota. En la base de datos el 0 indica que el bateador no intento hacer contacto; no hubo evento de un *swing.* El 1 indica que el bateador intentó hacer contacto con la pelota; si hubo evento de *swing*. La distribución se ve de la siguiente manera:

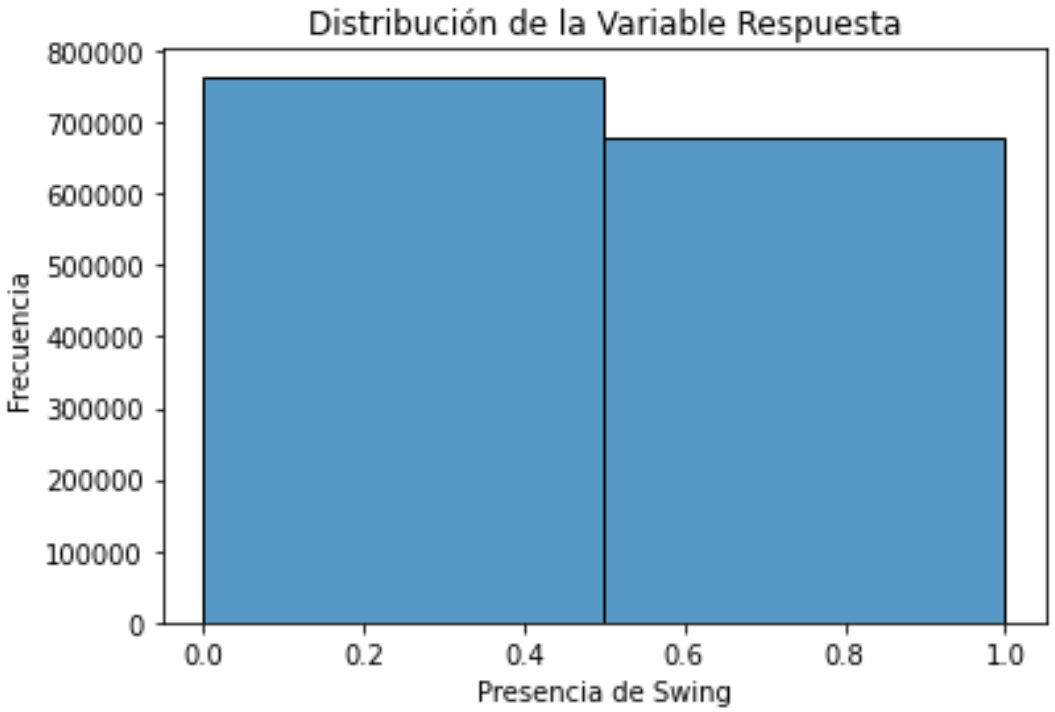

Figura 1: histograma de la variable respuesta.

En la figura 1 se puede observar que hay más ocasiones en las que un bateador decide no hacer swing ante el lanzamiento que en las que si decide hacerlo. En si no existe mucho desbalanceo entre las clases. El 52.99% de los datos pertenece a la clase 0 y el 47.01 a la clase 1. Este pequeño desbalanceo será considerado en el desarrollo del modelo, añadiendo una penalización. A continuación se desglosa el histograma de forma tabular:

¿Intento de Swing?	Frecuencia	Porcentaje
No (0)	764,736	52.99%
Si (1)	678540	47.01%
TOTAL	**1,443,276**	**100%**

Tabla 1: Frecuencia del intento de swing

Desarrollo del Modelo

Selección de Datos

La base de datos original tiene 1,443,276 observaciones. Para el desarrollo del modelo se estará utilizando scikit-learn, el cual no soporta tal cantidad de datos. Para poder utilizar dicha librería se decide tomar una muestra aleatoria de 10,000 observaciones de la base de datos original y a esta se le ajustarán los modelos seleccionados.

Split y Scaling

El data frame se divide en dos partes: X y Y. X se compone de las 20 variables predictoras y Y solamente contiene la variable respuesta. Las partes de X y Y se dividen en conjuntos de entrenamiento y de prueba utilizando el *train_test_split* con un tamaño de entrenamiento del 25%. Los datos se escalan y se transforman con scaler.

Modelo a Desarrollar

Se tiene información tabular por lo cual se considera que los modelos más apropiados para trabajar con los datos y poder hacer predicciones son: Máquina de Vectores de Soporte (SVM) y Bosques Aleatorios (RF). Los modelos se entrenan y se ajustan simultáneamente mediante validación cruzada con 10 pliegues. Se desean encontrar los mejores parámetros de cada modelo así que se utiliza una malla de búsqueda para encontrarlos y posteriormente ajustar y evaluar los modelos con los mismos. Se itera sobre los distintos parámetros de la malla y se establece una condición de paro con una tolerancia de 0.0001. A continuación se detalla cada modelo por separado.

SVM

Para la SVM la malla de búsqueda busca encontrar los mejores valores para los parámetros 'C' y 'gamma'. Se utiliza un delta de búsqueda de 0.25. Se utiliza un kernel RBF, base de función radial. En primera instancia la malla de búsqueda se enfoca en iterar valores de 'C' con valores de 2^k con k en un rango de [-5, 16]. En segunda instancia se itera en un rango de [-8, 8] actualizando con la delta de búsqueda. En primera instancia la malla de búsqueda se enfoca en iterar valores de 'gamma' con valores de 2^k con k en un rango de [-15, 4]. En segunda instancia se itera en un rango de [-8, 8] actualizando con la delta de búsqueda.

Random Forest

Para el bosque aleatoria la malla de búsqueda busca encontrar los valores óptimos o los mejores valores para los parámetros de 'min_samples_leaf', número de estimadores y el criterio para medir la calidad de una partición. Se utiliza un delta de búsqueda de 25. Para el parámetro de 'min_samples_leaf' la malla de búsqueda itera sobre valores de 1, 2, 4, 8 y 16. En primera instancia, para el número de estimadores, se itera sobre los valores de 100, 200, 400, 600, 800 y 1000. En segunda instancia se itera sobre estos mismos valores pero considerando un rango de [-25, 25]. En cuanto al criterio, se prueban los siguientes: gini, entropía y log_loss.

Resultados

Tras correr la malla de búsqueda y ajustar y evaluar los modelos con los parámetros encontrados se obtienen los siguientes resultados.

SVM

Parámetro/Métrica	Resultado
C	256.0
Gamma	0.004645340292979379
Kernel	RBF
Media	0.715263
Desviación Estándar	0.020887
Balanced Accuracy	0.7264251207729469
F1 score	0.7251278018088871

Tabla 2: Resultados implementación SVM.

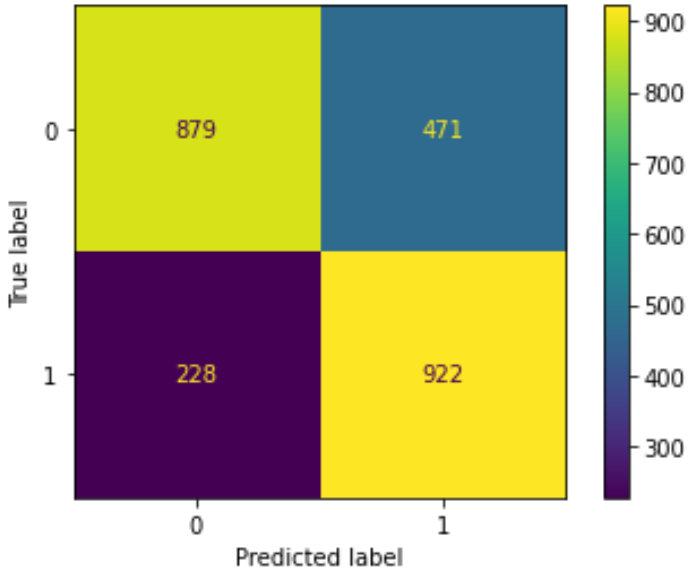

Figura 2: Matriz de confusión para SVM

Random Forest

Parámetro/Métrica	Resultado
Criterio	Entropy
Max_depth	1
n_estimators	1018
min_samples_leaf	1
Media	0.689542
Desviación Estándar	0.022430
Balanced Accuracy	0.7126570048309179
F1 score	0.6972245584524811

Tabla 3: Resultados implementación Random Forest

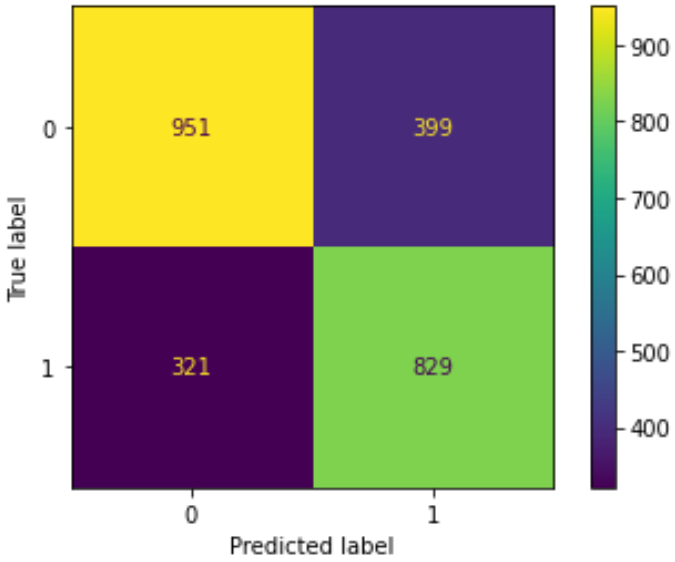

Figura 3: Matriz de confusión para Random Forest.

Conclusiones

De los dos modelos de predicción implementados el que mejor clasifica fue el SVM. Obtiene un resultado de balanced accuracy de 0.7264251207729469 (vs. 0.7126570048309179 del Random Forest) y un f1 score de 0.7251278018088871 (vs. 0.6972245584524811 del Random Forest). El modelo de SVM implementado tiene una media de 0.715263 y una desviación estándar de 0.020887. El Random Forest implementado tiene una media de 0.689542 y una desviación estándar de 0.022430.

En las matrices de confusión se puede observar que para el conjunto de prueba había 1350 ejemplos de la clase 0 y 1150 ejemplos de la clase 1. El modelo de SVM predice mejor la clase de 1 que el modelo de Random Forest mientras que el modelo de Random Forest predice mejor la clase 0 que el modelo de SVM.

Es importante entender que no existe un mejor modelo. Todos los modelos tienen sus ventajas y sus desventajas. Siempre hay algún tipo de costo de oportunidad. Hay veces que se sacrifica accuracy por precisión o viceversa.

Referencias

1. *Predict if a pitch will result in a swing*. Kaggle. (n.d.). Retrieved May 6, 2023, from https://www.kaggle.com/competitions/nwds-xswing/data
2. *Statcast search CSV documentation*. baseballsavant.com. (n.d.). Retrieved May 6, 2023, from https://baseballsavant.mlb.com/csv-docs