XGBoost

XGBoost es una librería basada en Gradient Boosting creada en el 2014 por los desarrolladores XGBoost. Su objetivo es ser una librería eficiente, portable y flexible para su implementación en aplicaciones de Machine Learning. La base de esta librería es Gradient Boosting aplicada en árboles de decisión. Actualmente se puede utilizar en los lenguajes de programación de python, R, Java, C++, Julia, Perl y Scala (Wikipedia; 2022).

A diferencia de otros frameworks, XGBoost no soporta variables categóricas, sólo variables numéricas. Si se desean utilizar variables categóricas, deben ser transformadas previamente para que sean aceptadas en el modelo. Los métodos de transformación más comunes para variables categóricas son los codificadores como one hot encoder, mean encoding, etc (Swalin. A; 2018).

Para su funcionamiento, XGBoost tiene múltiples parámetros pero los más importantes son los siguientes: parámetros que controlan el overfitting y parámetros que controlan la velocidad. Los parámetros que controlan el overfitting son Learning_rate (valor óptimo 0.01 - 0.2), max_depth y min_child_weight (valor default = 1) (Swalin. A; 2018).

Los parámetros para controlar la velocidad son: colsample_tree que es una submuestra del ratio de columnas, subsample que es una sub muestra de los datos de entrenamiento, y n_estimators que es el número máximo de árboles pero si su valor es muy alto puede generar overfitting. Por lo general este modelo es más lento que el LightGBM y el CatBoost pero va a depender del dataset (Swalin. A; 2018).

En cuanto a su estructura, para hacer la división de datos utiliza un algoritmo de pre-sorting y un algoritmo basado en un histograma para realizar la mejor división. El pre-sorting se encarga de enumerar todas las features para cada nodo. Luego, para cada feature realiza un sort para cada valor de la variable. Luego realiza un scan lineal para decidir el mejor split en base a la solución obtenida de todas las features. Este proceso ocurre con el algoritmo basado en el histograma, se dividen todos los valores de la variable en bins del histograma y con se utilizan esos bins para obtener el mejor split. Con este método se obtiene una gran eficiencia y velocidad de entrenamiento de datos ya que enumera todas las posibilidades de la variable (Swalin. A; 2018).

LightGMB

LightGBM es un framework creado por Microsoft en el 2016 basado en Gradient Boosting y árboles de decisión utilizado principalmente en el área de Machine Learning. Este framework es bastante similar a XGBoost, pero la gran diferencia es que sus árboles de decisión no crecen en tamaño/altura sino en cantidad de hojas por rama del árbol. Se elige la hoja que tenga el menor valor de pérdida de datos. Como consecuencia, LightGBM es un algoritmo bastante eficiente en las predicciones y el uso de memoria (Wikipedia; 2022).

Para encontrar el mejor split value, LightGMB utiliza el método Gradient Based One Sided Sampling (GOSS). La base de esta técnica es la gradiente que se utiliza para asignar los valores de la función de pérdida de los datos. Mientras mayor sea la gradiente, los datos tendrán un mayor margen de error. GOSS mantendrá los datos que tengan los mayores valores de gradientes y eliminará de forma aleatoria los valores más pequeños. De esta

forma, se puede reducir los datos que no tengan un aporte significativo en el entrenamiento del modelo y mantener la accuracy de los árboles de decisión (Swalin. A; 2018).

Los parámetros más importantes de LightGBM son los que controlan el overfitting, variables categóricas, y velocidad del modelo. Los parámetros para controlar el overfitting son el Learning_Rate, max_depth cuyo valor default es 20 y el árbol sí puede seguir creciendo en número de hojas por lo que realizar el tuneo es vital, num_leaves es uno de los parámetros más importantes de LGMB ya que es el número de hojas en cada árbol y debería ser menor a 2^max_depth, y min_data_leaf cuyo valor default es 20 (Swalin. A; 2018).

Puede soportar variables categóricas por lo que sus parámetros para estas variables son categorical_feature donde se especifica las variables categóricas que se desean usar para el entrenamiento del modelo. Si bien el modelo soporta variables categóricas, antes de ser ingresadas a este es recomendable pasarlas a tipo entero antes de construir el dataset para LGBM. Si no se pasan a enteros antes, el modelo no reconocerá strings incluso aunque se ponga en el parámetros categorical_feature (Swalin. A; 2018).

Los parámetros para controlar la velocidad son feature_fraction que es una fracción de variables extraídas en cada iteración, bagging_fraction que es data utilizada en cada iteración y generalmente agiliza el entrenamiento permitiendo evitar overfitting, y num_iterations que es el número de iteraciones Boosting a ser realizadas (valor default es 100) (Swalin. A; 2018).

CatBoost

CatBoost es un algoritmo desarrollado por Yandex y es un algoritmo basado en Gradient Boosting con árboles de decisión. Este algoritmo tiene múltiples aplicaciones como investigaciones, sistemas de recomendación, predicción del clima, etc. Permite crear modelos bastante eficientes ya que permite utilizar variables categóricas y su método de Novel Gradient Boosting reduce el overfitting (Yandex; 2022).

CatBoost tiene múltiples parámetros pero los más importantes son los parámetros para controlar overfitting, parámetros para variables categóricas y los parámetros para controlar la velocidad. En cuanto a los parámetros para controlar overfitting, se destacan el Learning_Rate que controla el ritmo de aprendizaje, Depth que puede ser un integer hasta 16 pero sus valores recomendados son del 1 - 10, y l2-leaf-reg es un coeficiente de regularización L2 que se utiliza para calcular el valor de las hojas (cualquier integer positivo es permitido) (Swalin. A; 2018).

Los parámetros de variables categóricas son cat_features que denota el index de las variables categóricas, y one_hot_max_size que utiliza one hot encoding para todas las features con números de diferentes valores (menores o iguales) al valor del parámetro dado (valor máximo -255). Si no se pasa nada en estos parámetros, el modelo tratará todas las variables como numéricas (Swalin. A; 2018).

Los parámetros para controlar la velocidad son rsm que es un método de subespacio aleatorio de un porcentaje de features para controlar cada corte de selección, y iteraciones que es la cantidad máxima de árboles que se puede construir y un valor alto puede generar overfitting (Swalin. A; 2018).

Referencias:

Swalin. A (2018). *CatBoost vs. Light GBM vs. XGBoost*. Recuperado de https://towardsdatascience.com/catboost-vs-light-gbm-vs-xgboost-5f93620723db [Consultado el 27 de mayo del 2022]}

Microsoft Corporation (2022). *LightGBM*. Recuperado de https://lightgbm.readthedocs.io/en/latest/ [Consultado el 29 de mayo del 2022]

Pandiri. S (2019). *LightGBM + XGBoost + Catboost*. Recuperado de https://www.kaggle.com/code/samratp/lightgbm-xgboost-catboost/notebook#Catboost [Consultado el 27 de mayo del 2022]

Wikipedia (2022). *LightGBM*. Recuperado de https://en.wikipedia.org/wiki/LightGBM [Consultado el 29 de mayo del 2022]

Wikipedia (2022). *XGBoost*. Recuperado de https://en.wikipedia.org/wiki/XGBoost [Consultado el 29 de mayo del 2022].

xgboost developers (2021). *XGBoost Documentation*. Recuperado de https://xgboost.readthedocs.io/en/stable/ [Consultado el 29 de mayo del 2022].

Yandex (2022). *CatBoost*. Recuperado de https://catboost.ai/ [Consultado el 29 de mayo del 2022]