CATBOOST

CatBoost name comes from two words Category and Boosting. CatBoost is a decision tree based ensemble machine learning algorithm that uses a gradient boosting framework. CatBoost can be used for both regression and classification problems.

CatBoost is a good choice to train if most of the features in your dataset are categorical

HOW IS CATBOOST DIFFERENT

- Missing value support

Just leave NaN. It can treat missing values automatically.

- Feature and Data point importance

CatBoost gives not only important features of the model but it also tells us that for a given data point what are the important features.

- Handling Categorical Features

CatBoost relies on the ordering principle also called Target-Based with prior (TBS). While calculating encoded value, instead of considering all the data points, it will consider only data points that are past in time to a data point and calculates the mean to the target values of those data points having the same categorical feature

By aligning the categorical features with the relative values of the target variables, the least amount of information loss is maintained in the implicit conversion of categorical features into a vector. .

Categorical feature values are encoded using the following formula:

$$\frac{TargetSum+prior}{FeatureCount+1}$$

TargetCount - It is sum of the target value for that particular categorical feature (upto the current one)

Prior - Sum of target values in the whole dataset/ Total number of observations (i.e. rows) in the dataset

FeatureCount - Total number of categorical features observed upto the current one with the same value as the current one.

For Example, if we have categorical feature column with values

Color = [“red”, “blue”, “blue”, “green”, “red”, “red”, “black”, “black”, “blue”, “green”]

Target= [1, 2, 3, 2, 3, 1, 4, 4, 2, 3]

Prior = 25/10 = 2.5

For the “red” category

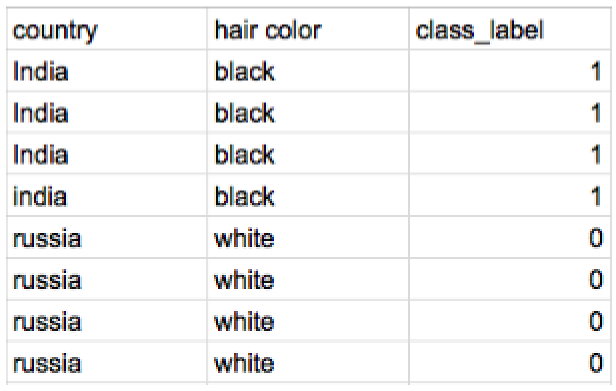
TargetCount will be 1+3+1 = 5

FeatureCount = 3

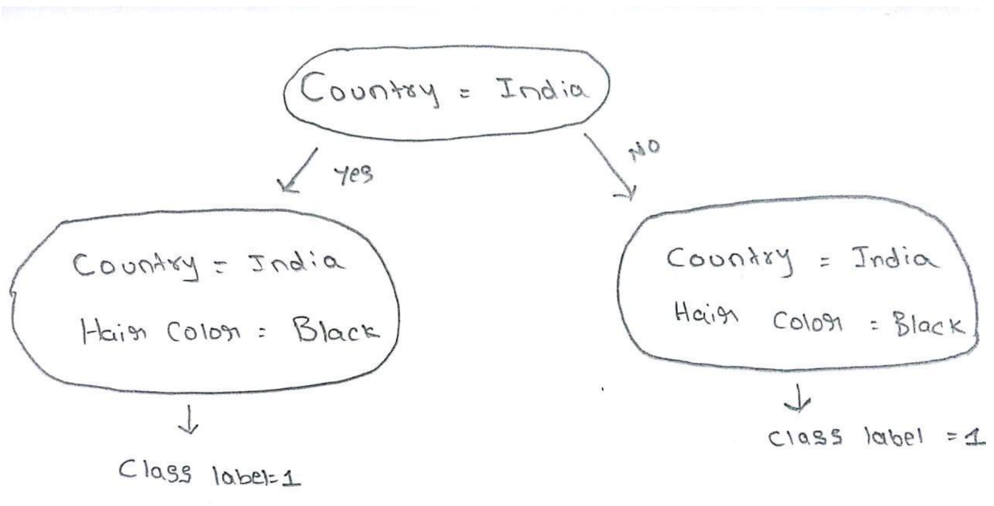
Encoded value for “red” = (5+2.5) / (3+1) = 1.875

- Categorical Feature Combinations

CatBoost combines multiple categorical features. For the most number of times combining two categorical features makes sense. CatBoost does this for you automatically.

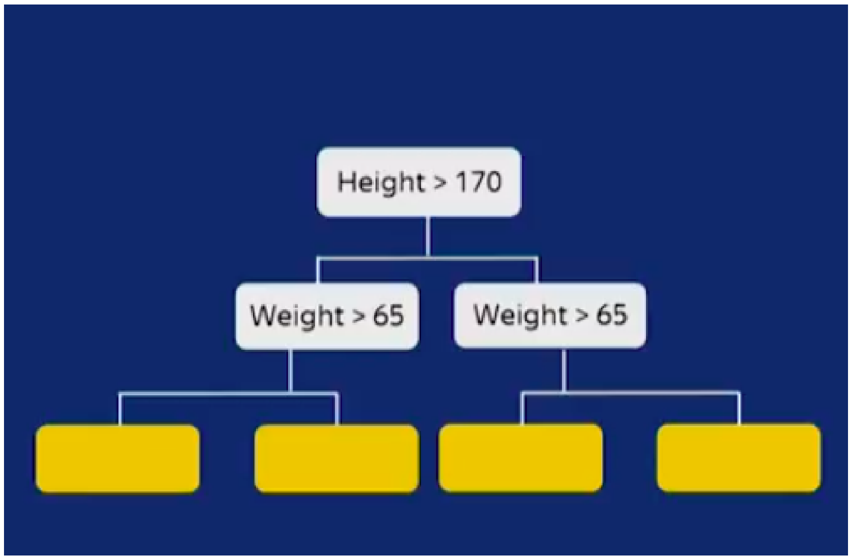


country	hair color	class_label
India	black	1
India	black	1
India	black	1
india	black	1
russia	white	0
russia	white	0
russia	white	0
russia	white	0



- Base Tree Structure

CatBoost implements symmetric trees. This may sound crazy but helps in decreasing prediction time which is extremely important for low latency environments.



- Overfitting detector

By default, CatBoost has an overfitting detector that it stops training when CV error starts increasing.

- GPU support

Catboost can be efficiently trained on a GPU.

HOW CATBOOST WORKS

CatBoost has two modes for choosing the tree structure

- Plain

Plain mode corresponds to a combination of the standard gradient boosting decision tree algorithm with an ordered Target Statistic

- Ordered

Classic boosting algorithms are prone to overfitting on small/noisy datasets due to a problem known as prediction shift. When calculating the gradient estimate of a data instance, these algorithms use the same data instances that the model was built with, thus having no chances of experiencing unseen data. CatBoost, on the other hand, uses the concept of ordered boosting, a permutation-driven approach to train model on a subset of data while calculating residuals on another subset, thus preventing target leakage and overfitting.

In ordered boosting, a tree is trained on a subset of the data set and used to calculated residuals for another subset that it hasn't seen. Here we calculate residuals for each data point using a model that has been trained on all the other data points at that time (To calculate residual for x5 datapoint, we train one model using x1, x2, x3, and x4). Hence we train different models to calculate residuals for different data points. In the end, we are calculating residuals for each data point that the corresponding model has never seen before. We then train the model by using the residuals of each data point as class labels. This process is repeated for specified number of iterations

In ordered boosting method, we should train n different models to get residuals for n data points. This is computationally expensive when we have many data points. Hence there exists a modification and by default instead of training one model per data point, it trains only log (num_of_datapoints) models. Now if a model has been trained on n data points then that model is used to calculate residuals for the next n data points. This procedure is called ordered boosting.