IDAO

INTERNATIONAL DATA ANALYSIS OLYMPIAD

IDAO 2022 ML bootcamp: boosting algorithms

ELENA KANTONISTOVA

Gradient Boosting & Practice

Theoretical part:

- Classical Boosting
- Popular gradient boosting implementations

Practical part - Python notebook with:

- Feature engineering
- Feature selection
- Modeling

Gradient boosting

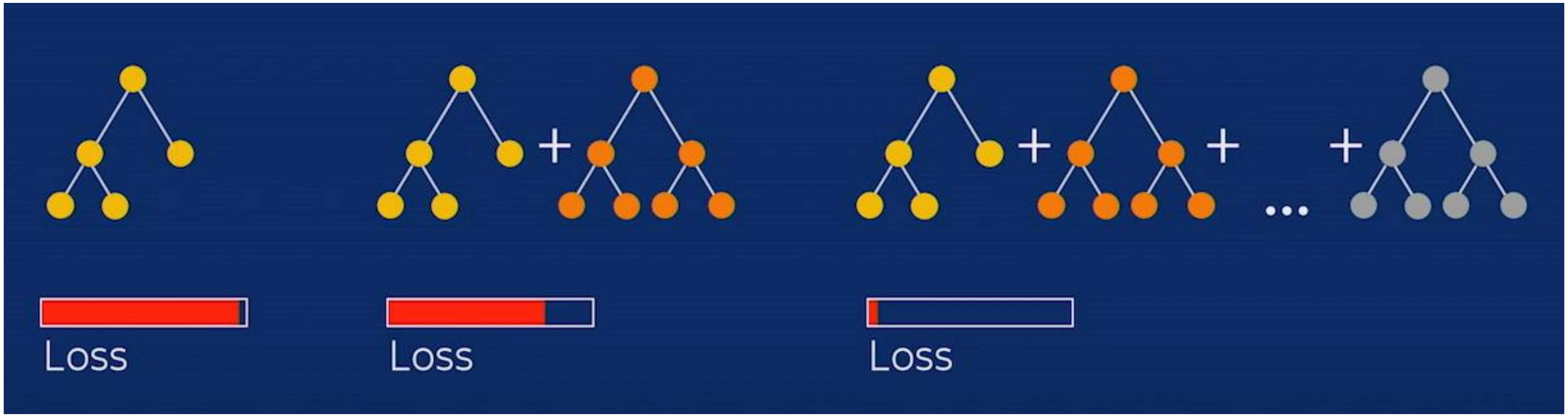
- Gradient boosting is a machine learning algorithm for classification and regression.
- It is an ensemble of multiple weak learners b_i:

$$a(x) = \sum_{i=1}^{N} b_i(x)$$

- In gradient boosting, we build trees step by step and add them to the composition.

Gradient boosting

<u>Key idea:</u> at each next step, we look for a basic algorithm that corrects the composition error in the previous step



Boosting algorithm

x is an object and y is a target vector.

- Boosting builds an ensemble of trees *one-by-one*, then the predictions of the individual trees *are summed*.

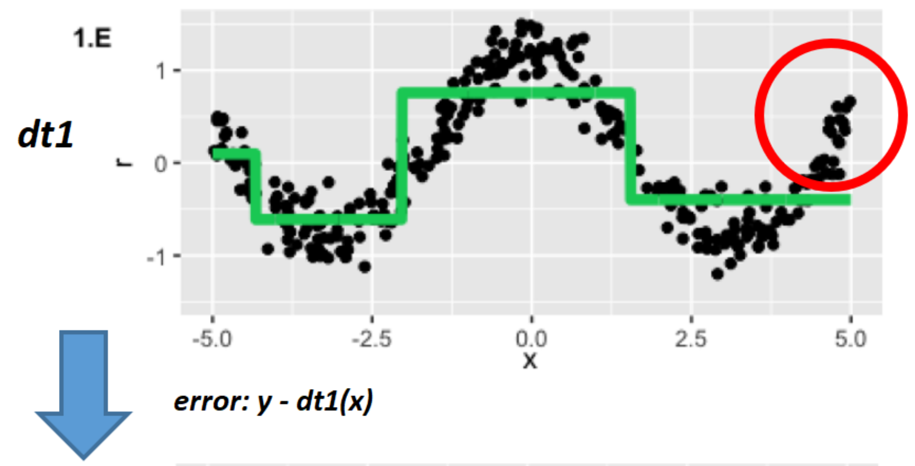
- If an ensemble has two trees at the current step, we have:

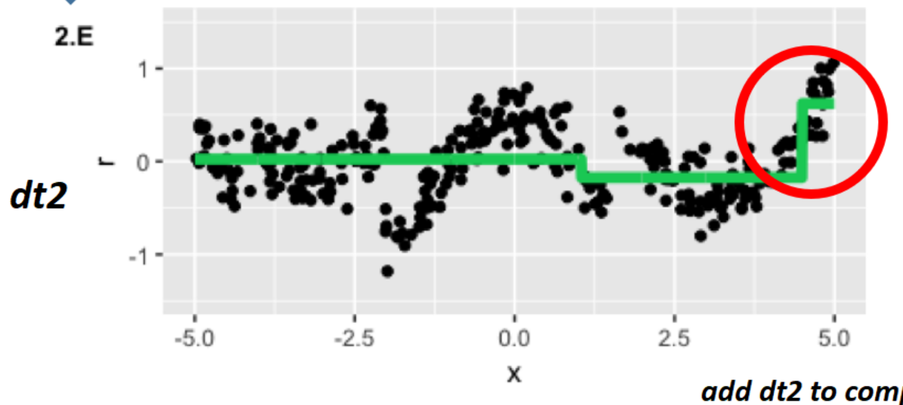
$$a(x) = b_1(x) + b_2(x)$$

- *The next decision tree $b_3(x)$ tries to predict the error of the composition*. That means that the target vector for the next algorithm is

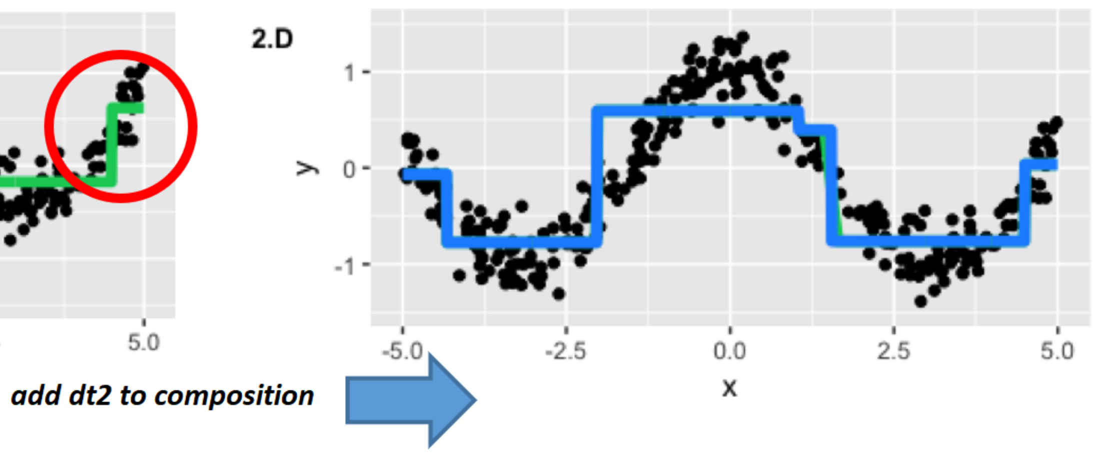
$$y_{new} = y - a(x)$$

Boosting algorithm



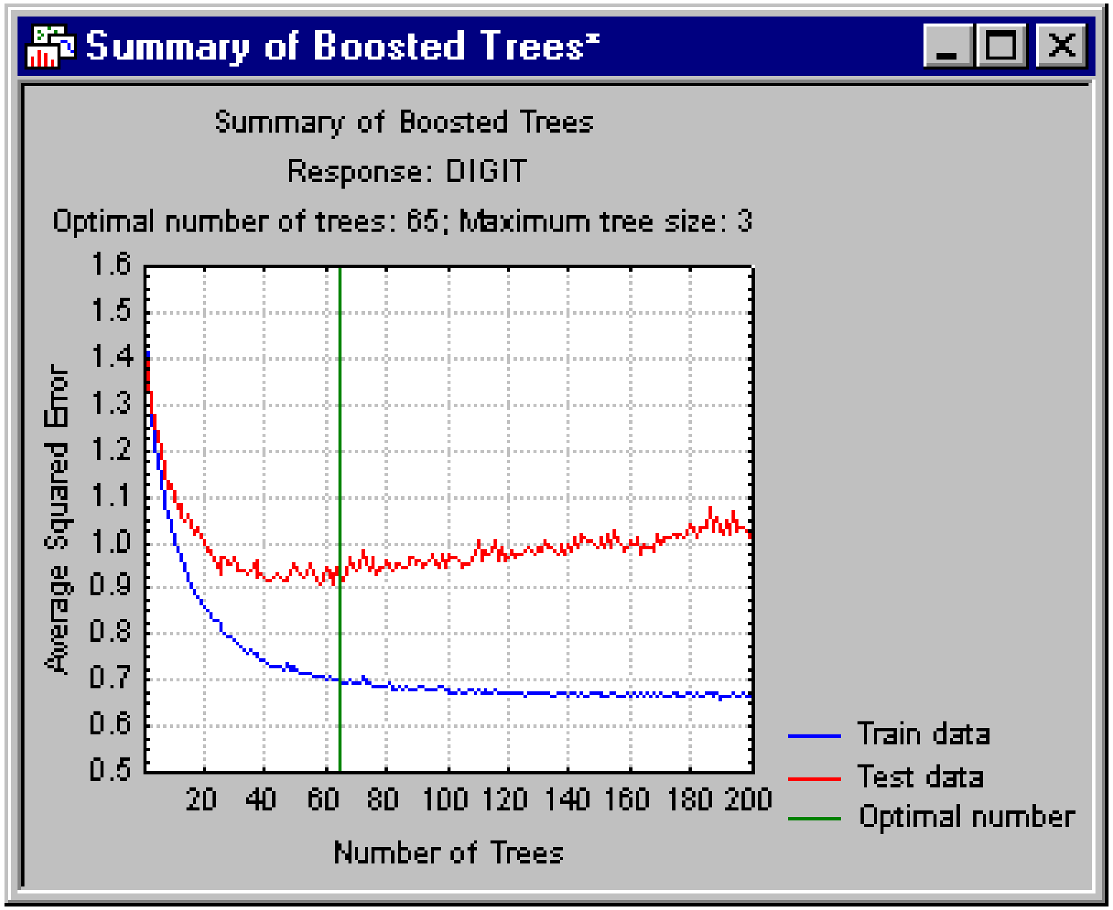


Boosting: a(x) = dt1(x) + dt2(x)



Gradient boosting overfits

- Since at each iteration we reduce the error on the training data, *at some point boosting will begin to overfit* - the error on the test data will begin to grow.
- *You need to find optimal number of trees* to minimize test error.



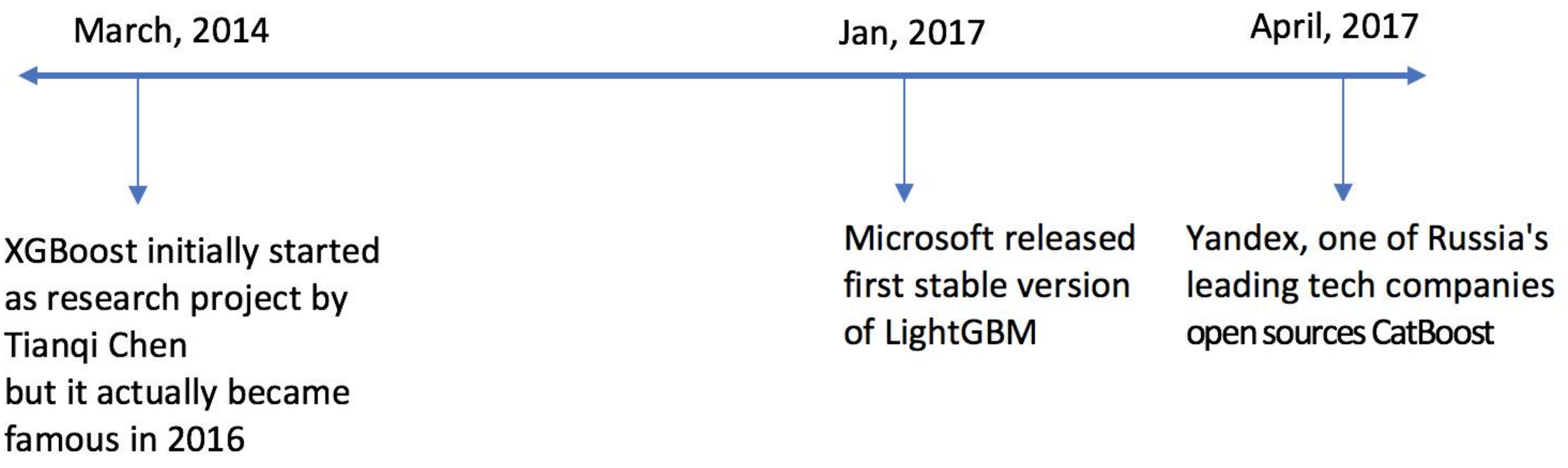
Gradient Boosting implementations

XGBoost

CatBoost

LightGBM

XGBoost, LightGBM, CatBoost



March, 2014

XGBoost initially started as research project by Tianqi Chen
but it actually became famous in 2016

Jan, 2017

Microsoft released first stable version of LightGBM

April, 2017

Yandex, one of Russia's leading tech companies open sources CatBoost

XGBoost loss (Extreme gradient boosting)

Main differences between XGBoost and classic gradient boosting:

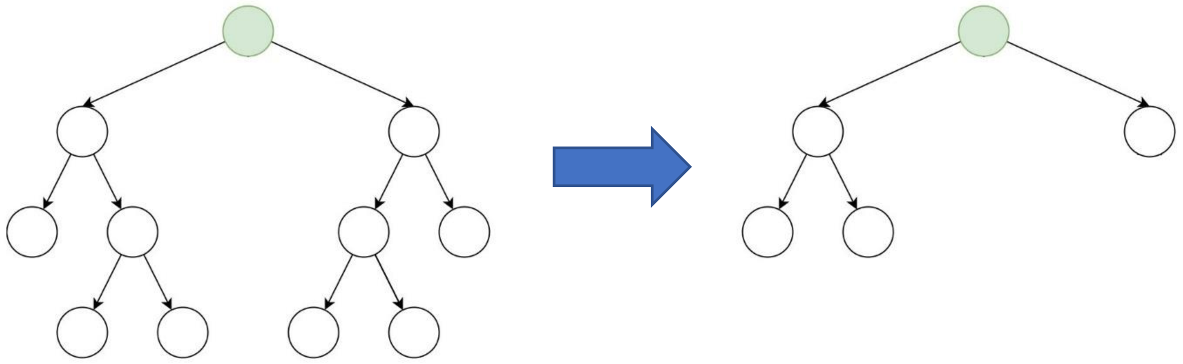
1. when constructing each weak learner, the *loss function is approximated to second-order* *(in gradient boosting – to first order)*

XGBoost loss (Extreme gradient boosting)

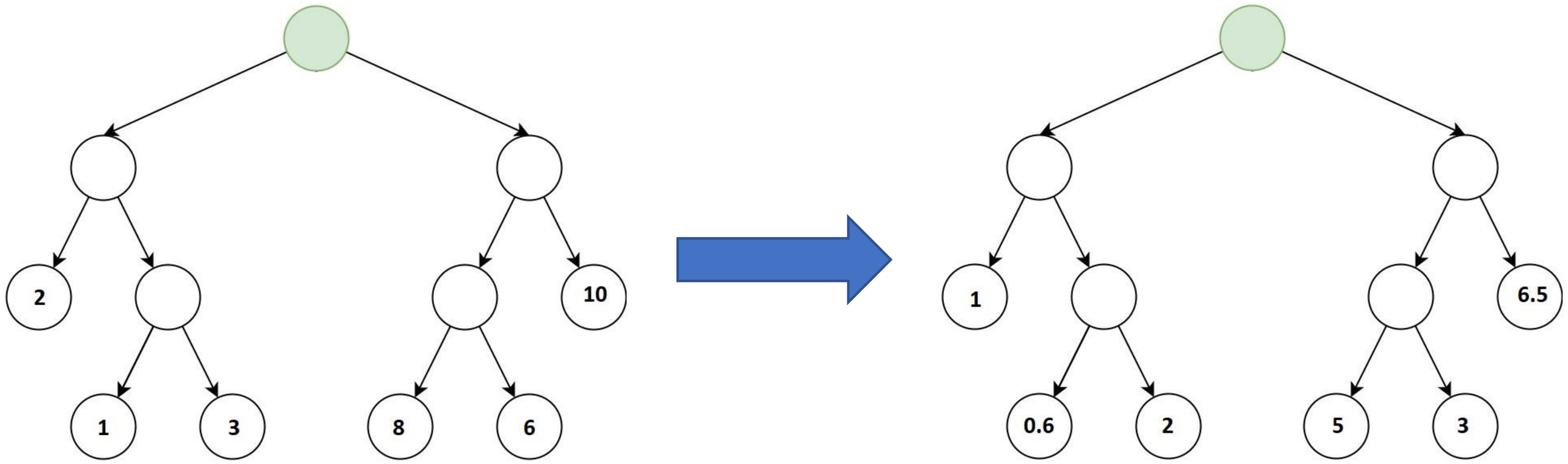
Main differences between XGBoost and classic gradient boosting:

2. we add a *regularization* to the loss: we penalize the tree

- for the number of leaves



- for large forecasts in the leaves



XGBoost

- Xgboost documentation
- https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/
- https://www.datacamp.com/community/tutorials/xgboost-in-python

CATBOOST

CatBoost is the first Russian machine learning algorithm developed to be open source. The algorithm was developed in the year 2017 by machine learning researchers and engineers at Yandex (a technology company).

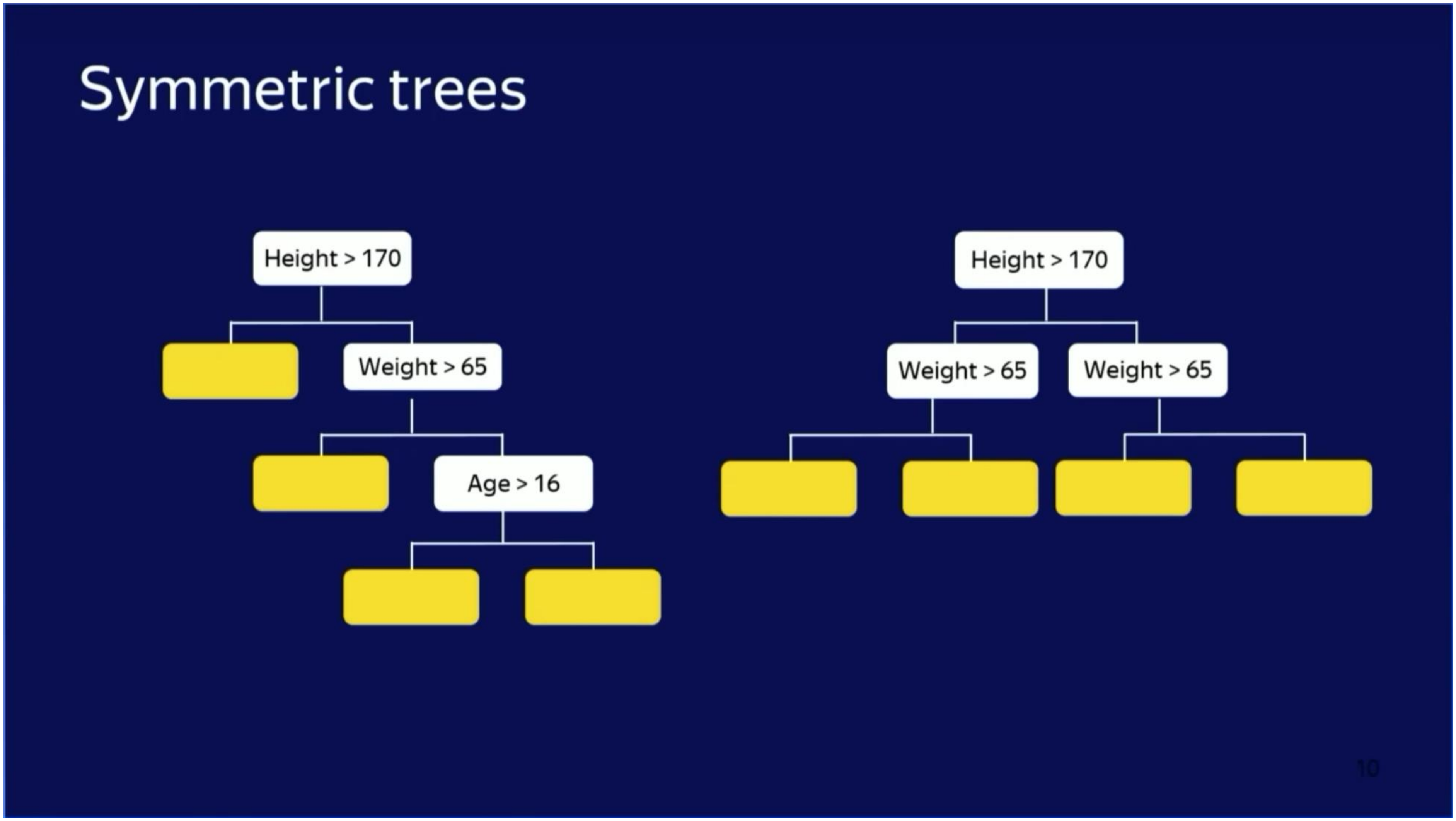
CATBOOST

Main advantages:

- Very good quality on many datasets
- High prediction speed
- Support for both numerical and categorical features
- Fast GPU support

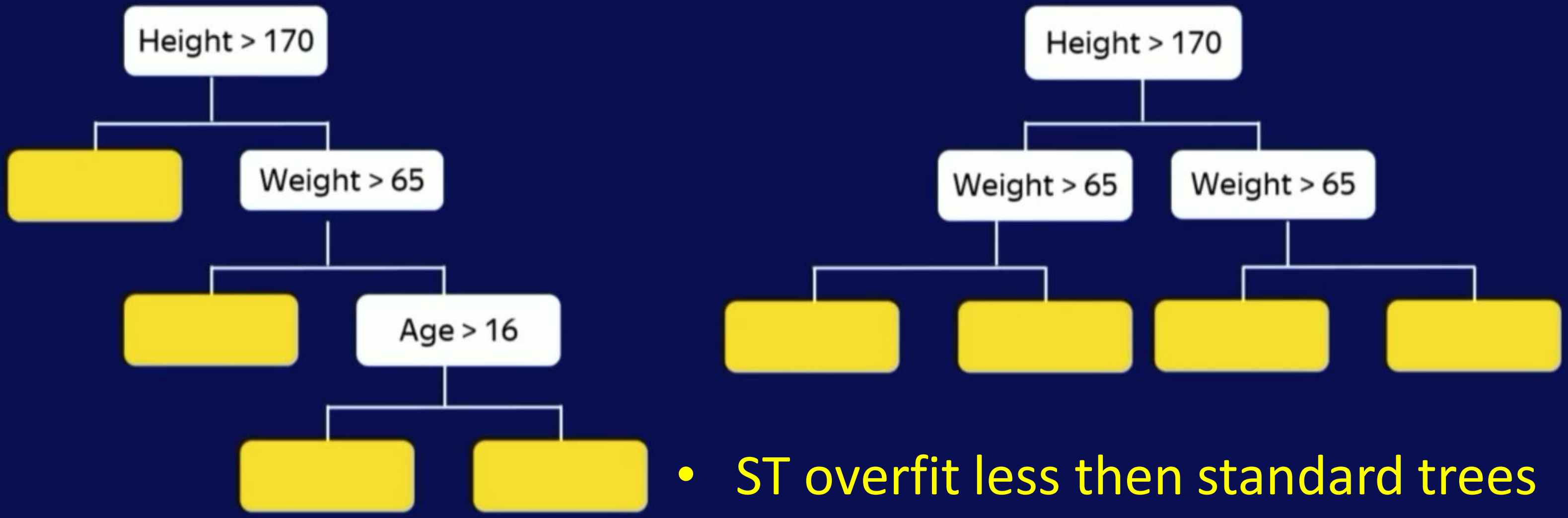
CATBOOST

- CatBoost builds *symmetric trees*:



CATBOOST

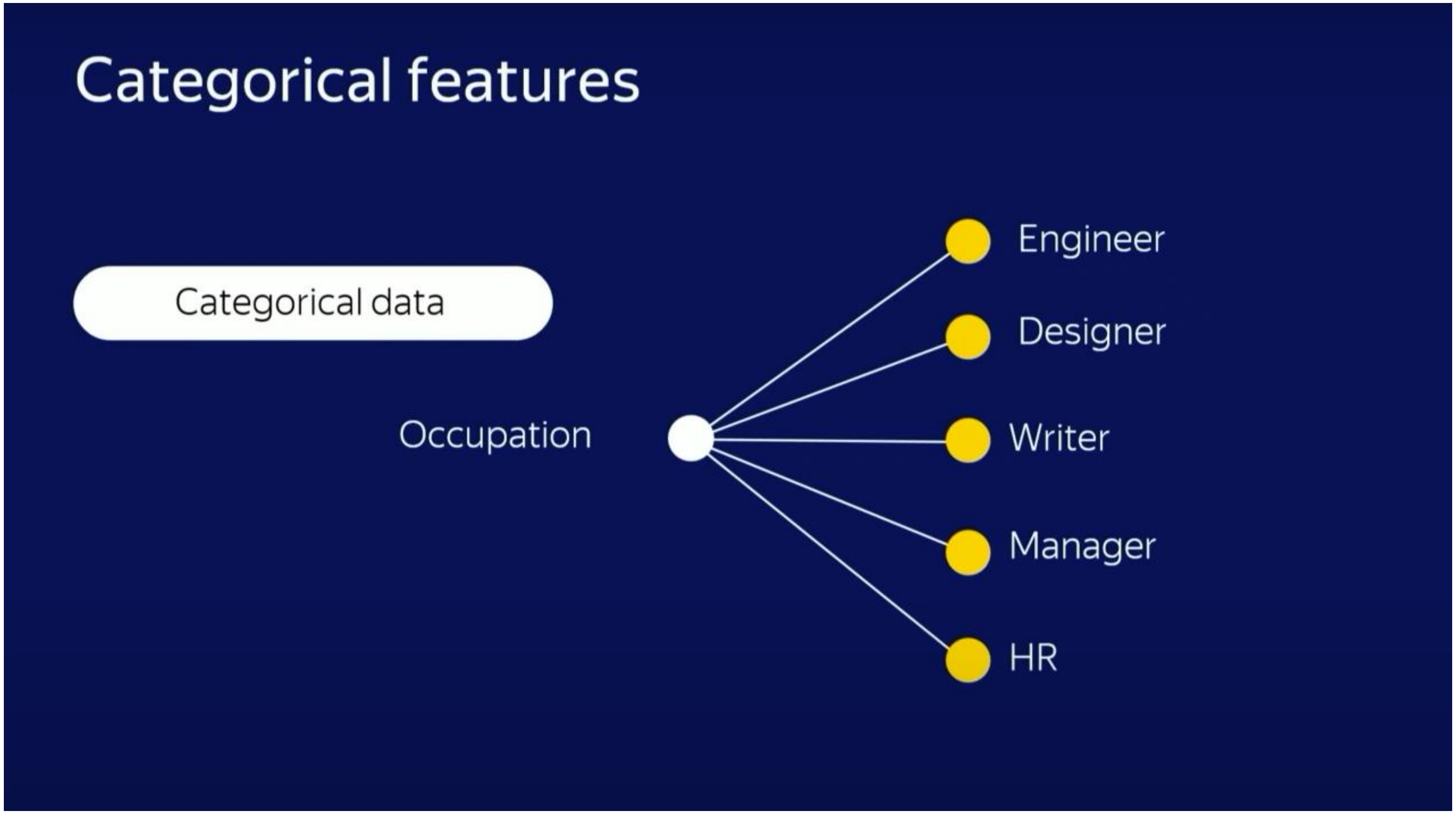
Symmetric trees (ST)



- ST overfit less then standard trees because of the fixed structure
- Thanks to ST, CatBoost shows good quality even with default parameters

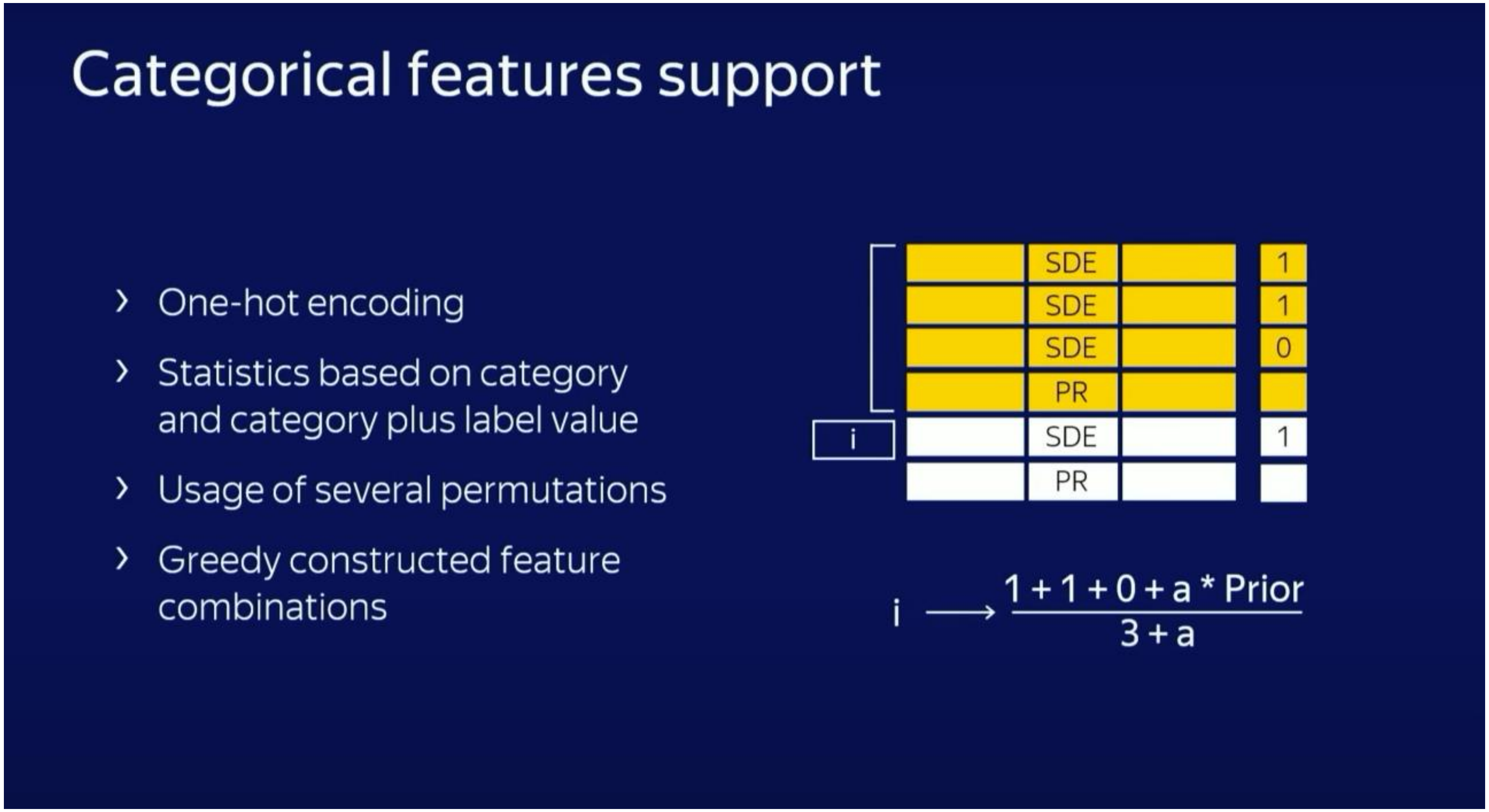
CATBOOST

- CatBoost *encodes categorical features*.



CATBOOST

- CatBoost encodes categorical features.



Categorical features support

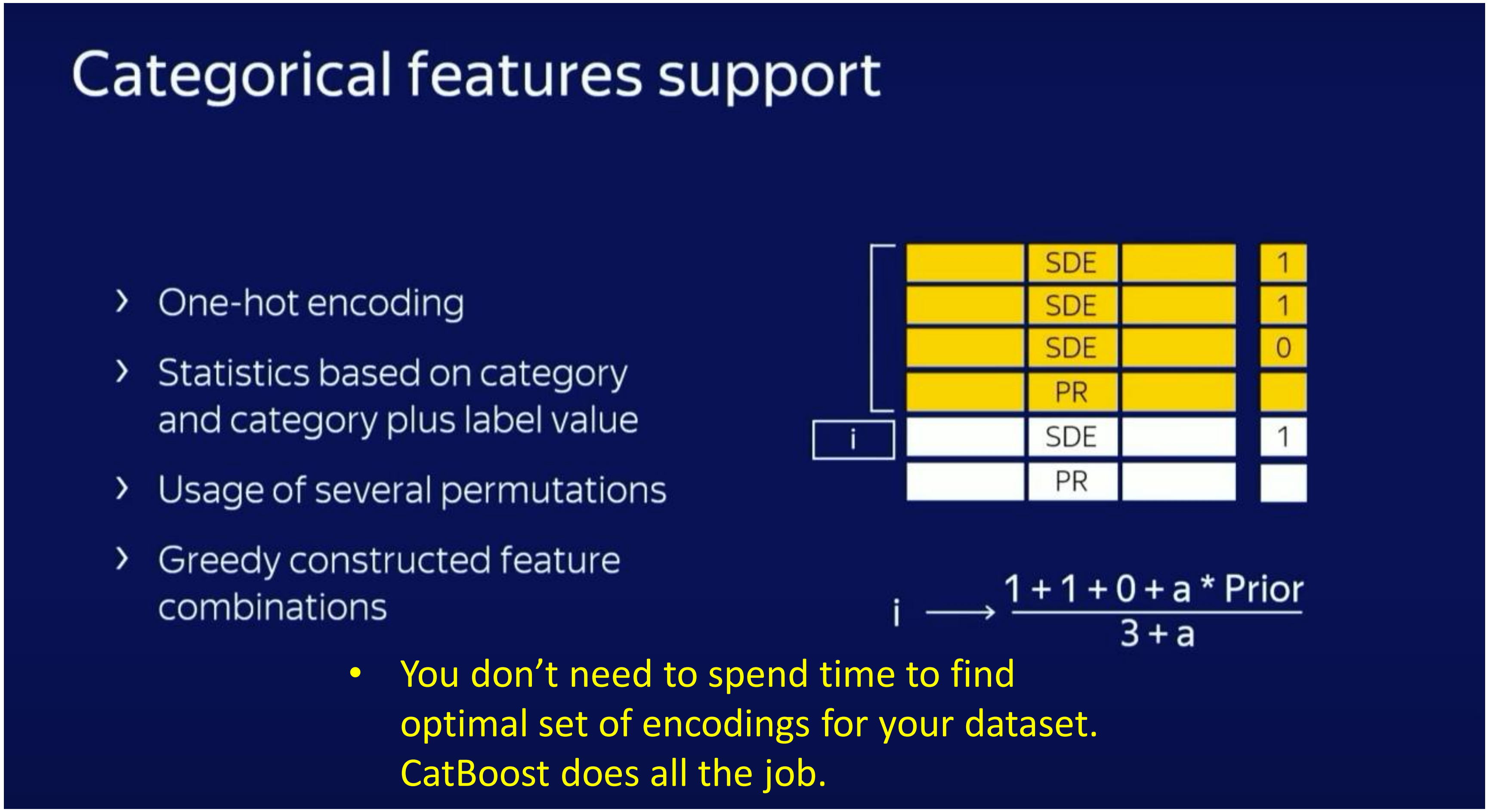
- One-hot encoding
- Statistics based on category and category plus label value
- Usage of several permutations
- Greedy constructed feature combinations

		SDE		1
		SDE		1
		SDE		0
		PR		
i		SDE		1
		PR		

$$i \longrightarrow \frac{1+1+0+a * \text{Prior}}{3+a}$$

CATBOOST

- CatBoost encodes categorical features.



Categorical features support

› One-hot encoding

› Statistics based on category and category plus label value

› Usage of several permutations

› Greedy constructed feature combinations

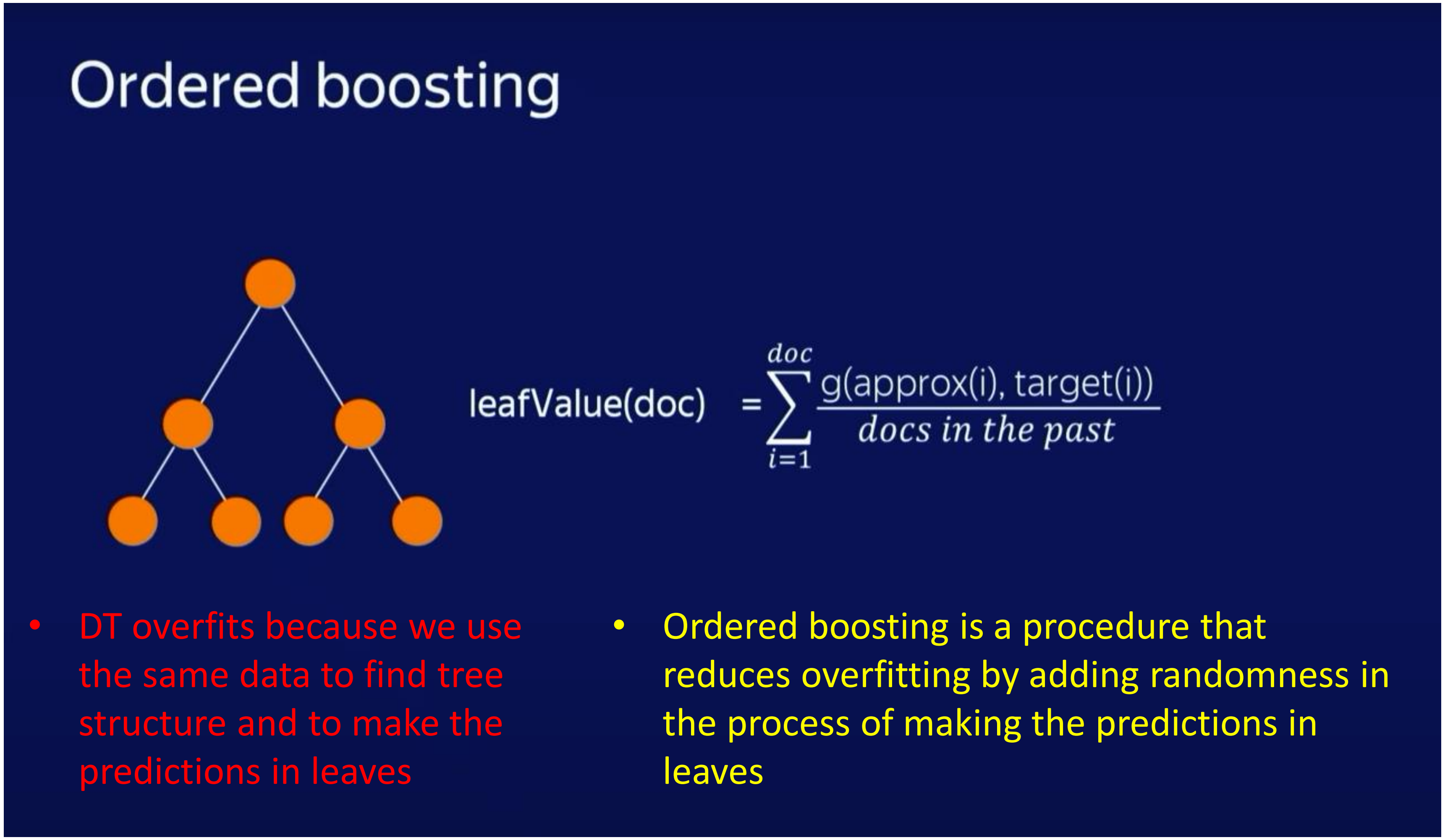
		SDE		1
		SDE		1
		SDE		0
		PR		
i		SDE		1
		PR		

$$i \longrightarrow \frac{1 + 1 + 0 + a * \text{Prior}}{3 + a}$$

- You don’t need to spend time to find optimal set of encodings for your dataset. CatBoost does all the job.

CATBOOST

- CatBoost *uses ordered boosting*



Ordered boosting

$$\text{leafValue(doc)} = \sum_{i=1}^{doc} \frac{\text{g(approx(i), target(i))}}{docs\ in\ the\ past}$$

- DT overfits because we use the same data to find tree structure and to make the predictions in leaves
- Ordered boosting is a procedure that reduces overfitting by adding randomness in the process of making the predictions in leaves

CatBoost

- CatBoost documentation
- video presentation (very good!)
- https://coderzcolumn.com/tutorials/machine-learning/catboost-an-in-depth-guide-python

LightGBM

LightGBM was developed by researchers in Microsoft Research in 2017. It is also an open source library.

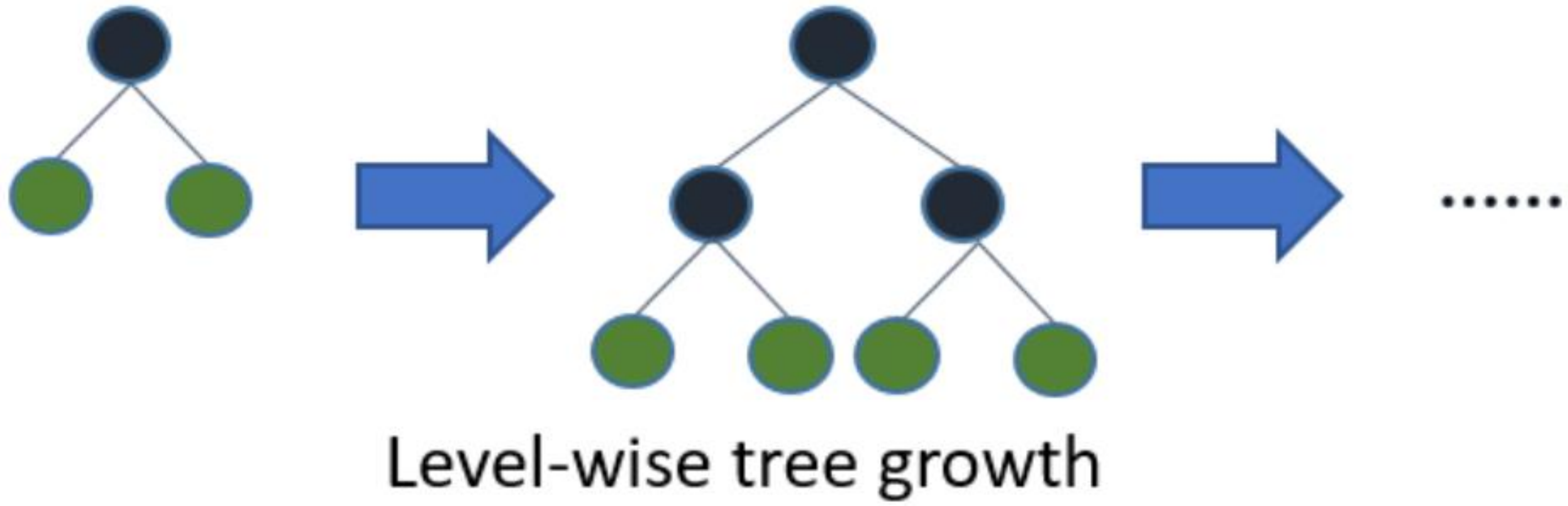
LightGBM

Main advantages of LightGBM:

- “Light” = high speed
- can handle large size of data
- focuses on the accuracy of results
- supports GPU learning

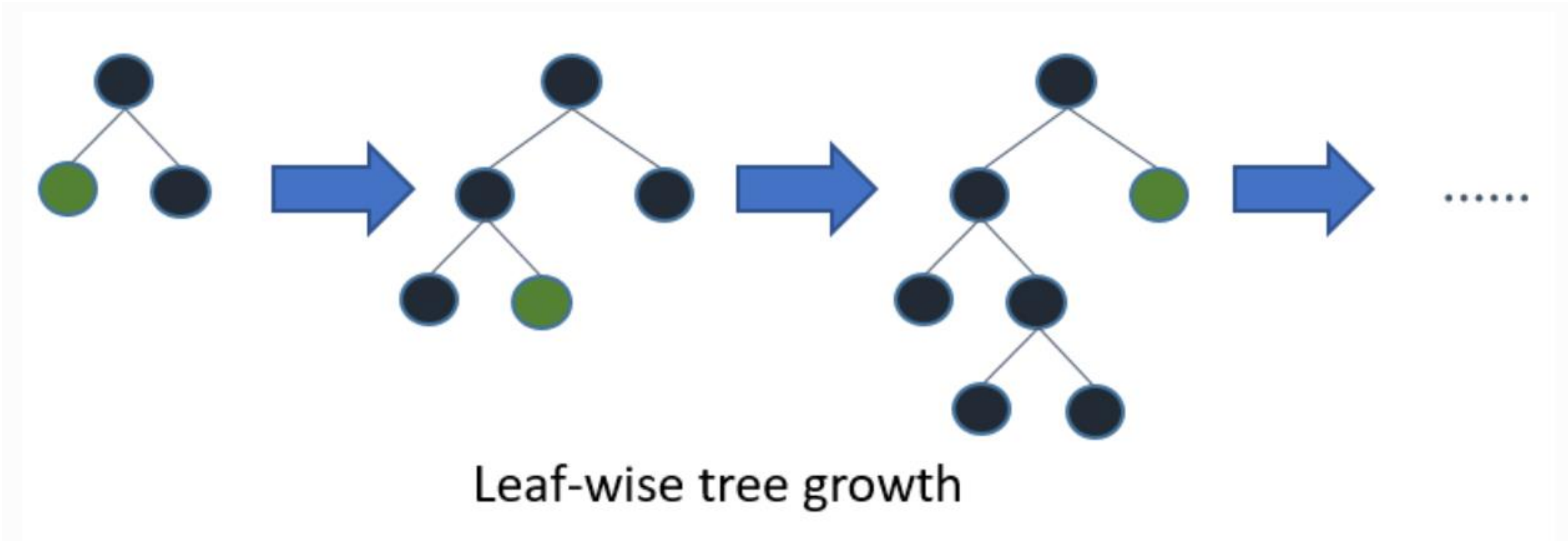
LightGBM

Usually boosting grows tree *level-wise*:



LightGBM

LightGBM grows tree *leaf-wise*:

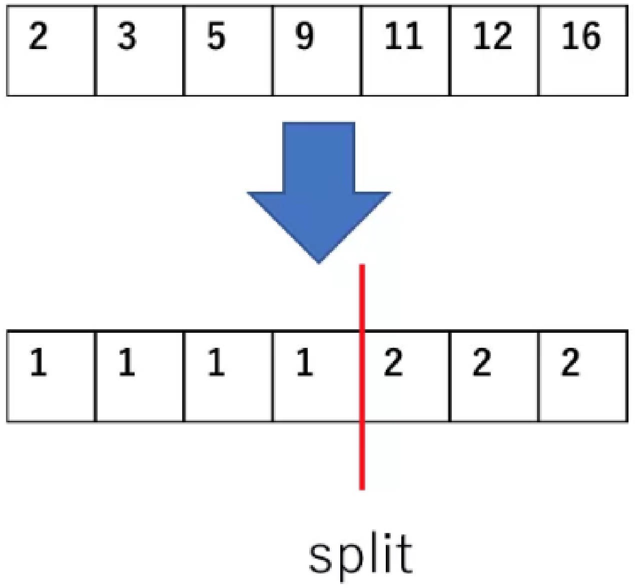


- it will choose the leaf with max delta loss to grow
- leaf-wise algorithm can reduce more loss than a level-wise algorithm

LightGBM

LightGBM works fast because of *feature bucketing*.

- it uses a histogram based algorithm to find the optimal split point while creating a weak learner
- therefore, each continuous numeric feature should be split into discrete bins



An example of how binning can reduce the number of splits to explore. The features must be sorted in advance for this method to be effective.

LightGBM

- LightGBM documentation
- https://towardsdatascience.com/what-makes-lightgbm-lightning-fast-a27cf0d9785e
- https://medium.com/@pushkarmandot/https-medium-com-pushkarmandot-what-is-lightgbm-how-to-implement-it-how-to-fine-tune-the-parameters-60347819b7fc

Thank you for your attention!

Now it's time for practice!