CatBoost is a recently open-sourced machine learning algorithm from Yandex. It can easily integrate with deep learning frameworks like Google’s TensorFlow and Apple’s Core ML. It can work with diverse data types to help solve a wide range of problems that businesses face today. To top it up, it provides best-in-class accuracy.

It is especially powerful in two ways:

* It yields state-of-the-art results without extensive data training typically required by other machine learning methods, and
* Provides powerful out-of-the-box support for the more descriptive data formats that accompany many business problems.

“CatBoost” name comes from two words “**Cat**egory” and “**Boost**ing”.

As discussed, the library works well with multiple **Cat**egories of data, such as audio, text, image including historical data.

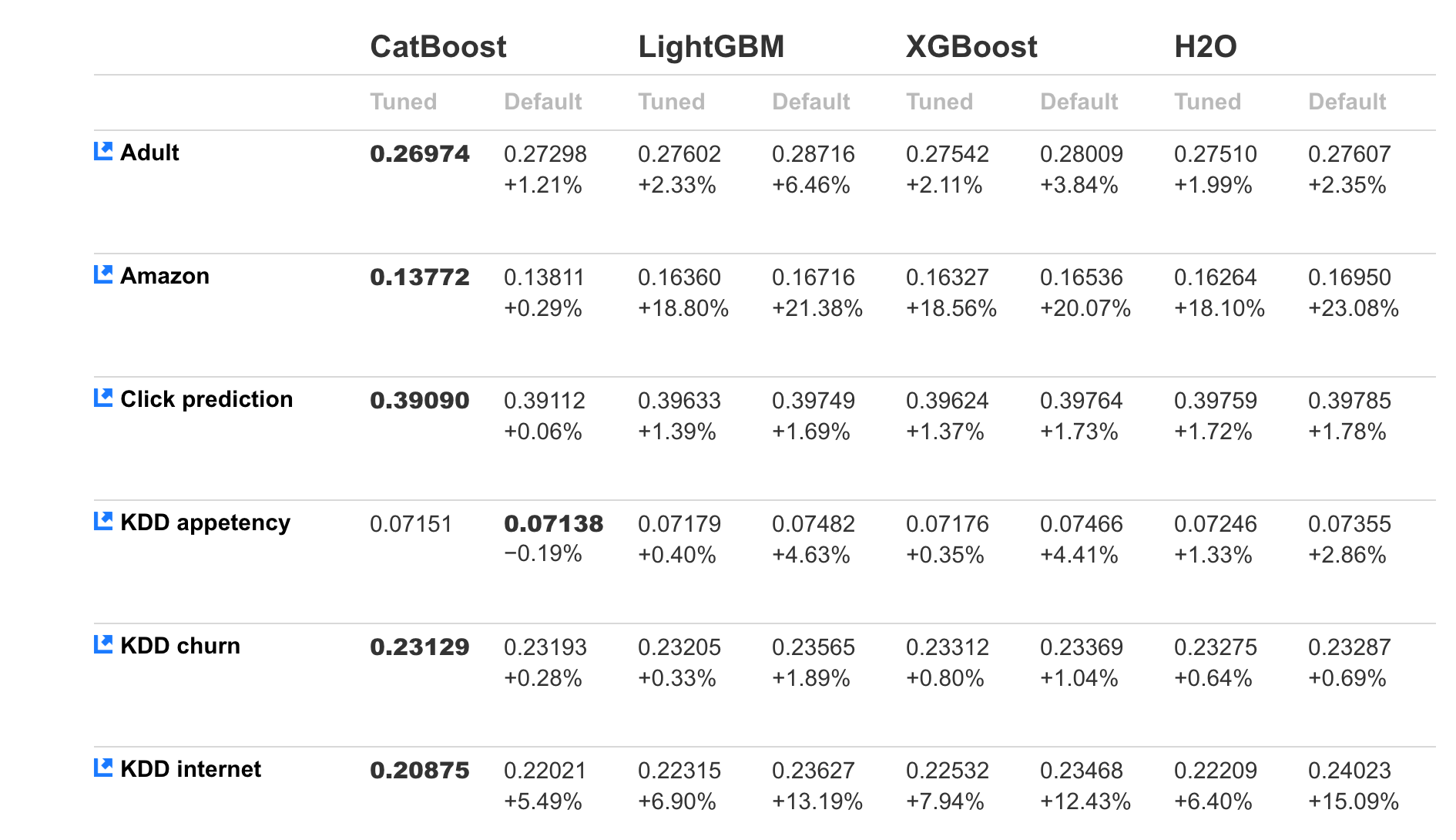
“**Boost**” comes from gradient boosting machine learning algorithm as this library is based on gradient boosting library. Gradient boosting is a powerful machine learning algorithm that is widely applied to multiple types of business challenges like fraud detection, recommendation items, forecasting and it performs well also. It can also return very good result with relatively less data, unlike DL models that need to learn from a massive amount of data.

**2. Advantages of CatBoost Library**

* **Performance:**CatBoost provides state of the art results and it is competitive with any leading machine learning algorithm on the performance front.
* **Handling Categorical features automatically:**We can use CatBoost without any explicit pre-processing to convert categories into numbers. CatBoost converts categorical values into numbers using various statistics on combinations of categorical features and combinations of categorical and numerical features. You can read more about it [here](https://tech.yandex.com/catboost/doc/dg/concepts/algorithm-main-stages_cat-to-numberic-docpage/).
* **Robust:**It reduces the need for extensive hyper-parameter tuning and lower the chances of overfitting also which leads to more generalized models. Although, CatBoost has multiple parameters to tune and it contains parameters like the number of trees, learning rate, regularization, tree depth, fold size, bagging temperature and others. You can read about all these parameters [here](https://tech.yandex.com/catboost/doc/dg/concepts/parameter-tuning-docpage/).
* **Easy-to-use:**You can use CatBoost from the command line, using an user-friendly API for both Python and R.

**3. CatBoost – Comparison to other boosting libraries**

We have multiple boosting libraries like XGBoost, H2O and LightGBM and all of these perform well on variety of problems. CatBoost developer have compared the performance with competitors on standard ML datasets:

[](https://cdn.analyticsvidhya.com/wp-content/uploads/2017/08/13153401/Screen-Shot-2017-08-13-at-3.33.33-PM.png)

The comparison above shows the log-loss value for test data and it is lowest in the case of **CatBoost**in most cases. It clearly signifies that CatBoost mostly performs better for both tuned and default models.

In addition to this, CatBoost does not require conversion of data set to any specific format like XGBoost and LightGBM.

## 5. Solving ML challenge using CatBoost

The CatBoost library can be used to solve both classification and regression challenge. For classification, you can use “**CatBoostClassifier**” and for regression, “***CatBoostRegressor***“.

we saw a recently open sourced boosting library “CatBoost” by Yandex which can provide state of the art solution for the variety of business problems.

One of the key features which excites me about this library is handling categorical values automatically using various statistical methods.

ExAMPLES

## Regression

[CatBoostRegressor](https://catboost.ai/docs/concepts/python-reference_catboostregressor.html#python-reference_catboostregressor) class with array-like data

from catboost import CatBoostRegressor

# Initialize data

train\_data = [[1, 4, 5, 6],

[4, 5, 6, 7],

[30, 40, 50, 60]]

eval\_data = [[2, 4, 6, 8],

[1, 4, 50, 60]]

train\_labels = [10, 20, 30]

# Initialize CatBoostRegressor

model = CatBoostRegressor(iterations=2,

learning\_rate=1,

depth=2)

# Fit model

model.fit(train\_data, train\_labels)

# Get predictions

preds = model.predict(eval\_data)

## Train on GPU

Train a classification model on GPU:

from catboost import CatBoostClassifier

train\_data = [[0, 3],

[4, 1],

[8, 1],

[9, 1]]

train\_labels = [0, 0, 1, 1]

model = CatBoostClassifier(iterations=1000,

task\_type="GPU",

devices='0:1')

model.fit(train\_data,

train\_labels,

verbose=False)

## Binary classification

[CatBoostClassifier](https://catboost.ai/docs/concepts/python-reference_catboostclassifier.html#python-reference_catboostclassifier) class with array-like data

from catboost import CatBoostClassifier

# Initialize data

cat\_features = [0, 1]

train\_data = [["a", "b", 1, 4, 5, 6],

["a", "b", 4, 5, 6, 7],

["c", "d", 30, 40, 50, 60]]

train\_labels = [1, 1, -1]

eval\_data = [["a", "b", 2, 4, 6, 8],

["a", "d", 1, 4, 50, 60]]

# Initialize CatBoostClassifier

model = CatBoostClassifier(iterations=2,

learning\_rate=1,

depth=2)

# Fit model

model.fit(train\_data, train\_labels, cat\_features)

# Get predicted classes

preds\_class = model.predict(eval\_data)

# Get predicted probabilities for each class

preds\_proba = model.predict\_proba(eval\_data)

# Get predicted RawFormulaVal

preds\_raw = model.predict(eval\_data, prediction\_type='RawFormulaVal')

Load the dataset using [Pool](https://catboost.ai/docs/concepts/python-reference_pool.html), train it with [CatBoostClassifier](https://catboost.ai/docs/concepts/python-reference_catboostclassifier.html) and make a prediction

from catboost import CatBoostClassifier, Pool

train\_data = Pool(data=[[1, 4, 5, 6],

[4, 5, 6, 7],

[30, 40, 50, 60]],

label=[1, 1, -1],

weight=[0.1, 0.2, 0.3])

model = CatBoostClassifier(iterations=10)

model.fit(train\_data)

preds\_class = model.predict(train\_data)

## Multiclassification

from catboost import Pool, CatBoostClassifier

train\_data = [["summer", 1924, 44],

["summer", 1932, 37],

["winter", 1980, 37],

["summer", 2012, 204]]

eval\_data = [["winter", 1996, 197],

["winter", 1968, 37],

["summer", 2002, 77],

["summer", 1948, 59]]

cat\_features = [0]

train\_label = ["France", "USA", "USA", "UK"]

eval\_label = ["USA", "France", "USA", "UK"]

train\_dataset = Pool(data=train\_data,

label=train\_label,

cat\_features=cat\_features)

eval\_dataset = Pool(data=eval\_data,

label=eval\_label,

cat\_features=cat\_features)

# Initialize CatBoostClassifier

model = CatBoostClassifier(iterations=10,

learning\_rate=1,

depth=2,

loss\_function='MultiClass')

# Fit model

model.fit(train\_dataset)

# Get predicted classes

preds\_class = model.predict(eval\_dataset)

# Get predicted probabilities for each class

preds\_proba = model.predict\_proba(eval\_dataset)

# Get predicted RawFormulaVal

preds\_raw = model.predict(eval\_dataset,

prediction\_type='RawFormulaVal')

## Get the best result for each metric

Return the best results for each metric calculated on the eval dataset:

from catboost import CatBoostClassifier, Pool

train\_data = [[0, 3],

[4, 1],

[8, 1],

[9, 1]]

train\_labels = [0, 0, 1, 1]

eval\_data = [[2, 1],

[3, 1],

[9, 0],

[5, 3]]

eval\_labels = [0, 1, 1, 0]

eval\_dataset = Pool(eval\_data,

eval\_labels)

model = CatBoostClassifier(learning\_rate=0.03,

custom\_metric=['Logloss',

'AUC:hints=skip\_train~false'])

model.fit(train\_data,

train\_labels,

eval\_set=eval\_dataset,

verbose=False)

print(model.get\_best\_score())

Note. This example illustrates the usage of the method with the [CatBoostClassifier](https://catboost.ai/docs/concepts/python-reference_catboostclassifier.html) class. The usage with other classes is identical.

## Get the identifier of the iteration with the best result

Return the iteration with the best value of the evaluation metric on the eval dataset:

from catboost import CatBoostClassifier, Pool

train\_data = [[0, 3],

[4, 1],

[8, 1],

[9, 1]]

train\_labels = [0, 0, 1, 1]

eval\_data = [[2, 1],

[3, 1],

[9, 0],

[5, 3]]

eval\_labels = [0, 1, 1, 0]

eval\_dataset = Pool(eval\_data,

eval\_labels)

model = CatBoostClassifier(learning\_rate=0.03,

eval\_metric='AUC')

model.fit(train\_data,

train\_labels,

eval\_set=eval\_dataset,

verbose=False)

print(model.get\_best\_iteration())

Note. This example illustrates the usage of the method with the [CatBoostClassifier](https://catboost.ai/docs/concepts/python-reference_catboostclassifier.html) class. The usage with other classes is identical.

## Load the dataset from list, ndarray, pandas.DataFrame, pandas.Series

Dataset with categorical features

from catboost import Pool

cat\_features = [0, 1, 2]

data = [["a","b", 1, 4, 5, 6],

["a","b", 4, 5, 6, 7],

["c","d", 30, 40, 50, 60]]

label = [1, 1, -1]

dataset = Pool(data, label, cat\_features)

Dataset without categorical features

from catboost import Pool

data = [[1, 4, 5, 6],

[4, 5, 6, 7],

[30, 40, 50, 60]]

label = [1, 1, -1]

dataset = Pool(data, label)

Dataset without labels (for prediction)

from catboost import Pool

data = [[1, 4, 5, 6],

[4, 5, 6, 7],

[30, 40, 50, 60]]

dataset = Pool(data)

## Load the dataset from a file

Dataset without specified columns description (without categorical features)

from catboost import Pool

dataset = Pool("data.tsv")

pool\_no\_categ is the file following file with the [object descriptions](https://catboost.ai/docs/concepts/input-data_values-file.html#input-data_values-file):

1 1935 01

1 1958 08

0 1969 09

Since the columns description file is not specified, it is assumed that the first column of the file (indexed 0) defines the label value, and all other columns are the values of numerical features.

Dataset with specified columns description (with categorical features)

from catboost import Pool

dataset = Pool("data\_with\_cat\_features.tsv",

column\_description="data\_with\_cat\_features.cd")

* pool is the following file with the [object descriptions](https://catboost.ai/docs/concepts/input-data_values-file.html#input-data_values-file):
* 1935 born 1
* 1958 deceased 1

1969 born 0

* pool.cd is the following file with the [columns description](https://catboost.ai/docs/concepts/input-data_column-descfile.html#input-data_column-descfile):
* 1 Categ

2 Label

Dataset in libsvm format

1. Create a file (data.libsvm in this example) with the dataset in the [extended libsvm format](https://catboost.ai/docs/concepts/input-data_libsvm.html):
2. 1 1:0.1 3:2.2 4:3
3. 0 2:0.22 3:0.82
4. 0 1:0.02 4:0.61

1 3:0.72 4:0.5

1. Load the dataset:
2. from catboost import Pool

dataset = Pool("libsvm://data.libsvm")

Dataset in extended libsvm format with categorical features

1. Create a file (data\_with\_cat\_features.libsvm in this example) with the dataset in the [extended libsvm format](https://catboost.ai/docs/concepts/input-data_libsvm.html):
2. 1 1:0.1 3:small 4:3 5:Male
3. 0 2:0.22 3:small 5:Female
4. 0 1:0.02 4:0.61 5:Female

1 3:large 4:0.5 5:Male

1. Create the corresponding [Columns description](https://catboost.ai/docs/concepts/input-data_column-descfile.html) file (data\_with\_cat\_features\_for\_libsvm.cd in this example):
2. 0 Label
3. 1 Num
4. 2 Num
5. 3 Categ
6. 4 Num

5 Categ

1. Load the dataset:
2. from catboost import Pool
3. dataset = Pool("libsvm://data\_with\_cat\_features.libsvm",

column\_description="data\_with\_cat\_features\_for\_libsvm.cd")

## Load the dataset from sparse python data

Dataset as scipy.sparse.csr\_matrix

import numpy as np

import scipy.sparse

import catboost as cb

row = np.array([0, 0, 1, 2, 2, 2, 3, 3, 4])

col = np.array([0, 2, 2, 0, 1, 2, 0, 2, 2])

data = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9])

X = scipy.sparse.csr\_matrix((data, (row, col)), shape=(5, 3))

y = np.array([0, 1, 0, 1, 0])

dataset = cb.Pool(X, y)

Dataset as pandas.SparseDataFrame

import numpy as np

import pandas as pd

import catboost as cb

X = pd.SparseDataFrame(

{'a': [ 1, 4, 0, 0, 1],

'b': [ 0, 0, 1, 7, 8],

'c': [30, 0, 0, 50, 0]

}

)

y = np.array([0, 1, 0, 1, 1])

dataset = cb.Pool(X, y)

Dataset as pandas.DataFrame with sparse columns with categorical features

import numpy as np

import pandas as pd

import catboost as cb

X = pd.DataFrame(

{'a': pd.SparseArray([ 1, 4, 0, 0, 1]),

'b': pd.SparseArray([ 0.0, 0.0, 1.0, 7.0, 8.0]),

'c': pd.SparseArray([ 30, 0, 0, 50, 0]),

'd': pd.SparseArray([ 'a', 'b', '', 'c', ''], fill\_value=''),

}

)

y = np.array([0, 1, 0, 1, 1])

dataset = cb.Pool(X, y, cat\_features=['d'])