CatBoost

Michael Matějů

AI Squad

July 30, 2023

Outline

Motivation

- MRMR Feature Selection Algorithm implemented by Smazzanti
- For categorical encoding he used three target based algorithms
- And inspite of all common sense it works in Kaggle Challanges
- CatBoost is an algorithm for gradient boosting on decision trees.
- It is developed by Yandex since 2009, resp. 2014-2015.
- In 2016 it went open-source.
- JetBrains uses CatBoost for Code Completion
- Cloudflare uses CatBoost for bot detection
- Careem uses CatBoost to predict future destinations of the rides

Motivation

- MRMR Feature Selection Algorithm implemented by Smazzanti
- For categorical encoding he used three target based algorithms
- And inspite of all common sense it works in Kaggle Challanges
- CatBoost is an algorithm for gradient boosting on decision trees.
- It is developed by Yandex since 2009, resp. 2014-2015.
- In 2016 it went open-source.
- JetBrains uses CatBoost for Code Completion
- Cloudflare uses CatBoost for bot detection
- Careem uses CatBoost to predict future destinations of the rides

Motivation

- MRMR Feature Selection Algorithm implemented by Smazzanti
- For categorical encoding he used three target based algorithms
- And inspite of all common sense it works in Kaggle Challanges
- CatBoost is an algorithm for gradient boosting on decision trees.
- It is developed by Yandex since 2009, resp. 2014-2015.
- In 2016 it went open-source.
- JetBrains uses CatBoost for Code Completion
- Cloudflare uses CatBoost for bot detection
- Careem uses CatBoost to predict future destinations of the rides

Motivation

- MRMR Feature Selection Algorithm implemented by Smazzanti
- For categorical encoding he used three target based algorithms
- And inspite of all common sense it works in Kaggle Challanges
- CatBoost is an algorithm for gradient boosting on decision trees.
- It is developed by Yandex since 2009, resp. 2014-2015.
- In 2016 it went open-source.
- JetBrains uses CatBoost for Code Completion
- Cloudflare uses CatBoost for bot detection
- Careem uses CatBoost to predict future destinations of the rides

Motivation

- MRMR Feature Selection Algorithm implemented by Smazzanti
- For categorical encoding he used three target based algorithms
- And inspite of all common sense it works in Kaggle Challanges
- CatBoost is an algorithm for gradient boosting on decision trees.
- It is developed by Yandex since 2009, resp. 2014-2015.
- In 2016 it went open-source.
- JetBrains uses CatBoost for Code Completion
- Cloudflare uses CatBoost for bot detection
- Careem uses CatBoost to predict future destinations of the rides

Motivation

- MRMR Feature Selection Algorithm implemented by Smazzanti
- For categorical encoding he used three target based algorithms
- And inspite of all common sense it works in Kaggle Challanges
- CatBoost is an algorithm for gradient boosting on decision trees.
- It is developed by Yandex since 2009, resp. 2014-2015.
- In 2016 it went open-source.
- JetBrains uses CatBoost for Code Completion
- Cloudflare uses CatBoost for bot detection
- Careem uses CatBoost to predict future destinations of the rides

Motivation

- MRMR Feature Selection Algorithm implemented by Smazzanti
- For categorical encoding he used three target based algorithms
- And inspite of all common sense it works in Kaggle Challanges
- CatBoost is an algorithm for gradient boosting on decision trees.
- It is developed by Yandex since 2009, resp. 2014-2015.
- In 2016 it went open-source.
- JetBrains uses CatBoost for Code Completion
- Cloudflare uses CatBoost for bot detection
- Careem uses CatBoost to predict future destinations of the rides

Motivation

- MRMR Feature Selection Algorithm implemented by Smazzanti
- For categorical encoding he used three target based algorithms
- And inspite of all common sense it works in Kaggle Challanges
- CatBoost is an algorithm for gradient boosting on decision trees.
- It is developed by Yandex since 2009, resp. 2014-2015.
- In 2016 it went open-source.
- JetBrains uses CatBoost for Code Completion
- Cloudflare uses CatBoost for bot detection
- Careem uses CatBoost to predict future destinations of the rides

Motivation

- MRMR Feature Selection Algorithm implemented by Smazzanti
- For categorical encoding he used three target based algorithms
- And inspite of all common sense it works in Kaggle Challanges
- CatBoost is an algorithm for gradient boosting on decision trees.
- It is developed by Yandex since 2009, resp. 2014-2015.
- In 2016 it went open-source.
- JetBrains uses CatBoost for Code Completion
- Cloudflare uses CatBoost for bot detection
- Careem uses CatBoost to predict future destinations of the rides

Overview of Gradient Boosting

Decition trees are weak learners. However, it has been shown that "combination" of weak learners you can achieve good results. What the the most common combinatons?

- Bagging: This technique builds different models in parallel using random subsets of data and deterministically aggregates the predictions of all predictors.
- Boosting: This technique is iterative, sequential, and adaptive as each predictor fixes its predecessor's error.
- Stacking: It is a meta-learning technique that involves combining predictions from multiple machine learning algorithms, like bagging and boosting.

CatBoost Features

1. **Symmetric trees**: CatBoost builds symmetric (balanced) trees, unlike XGBoost and LightGBM.
2. **Ordered boosting**: CatBoost 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 over-fitting.
3. **Native feature support**: CatBoost supports all kinds of features be it numeric, categorical, or text and saves time and effort of preprocessing.

CatBoost Features

1. **Symmetric trees**: CatBoost builds symmetric (balanced) trees, unlike XGBoost and LightGBM.
2. **Ordered boosting**: CatBoost 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 over-fitting.
3. **Native feature support**: CatBoost supports all kinds of features be it numeric, categorical, or text and saves time and effort of preprocessing.

CatBoost Features

1. **Symmetric trees**: CatBoost builds symmetric (balanced) trees, unlike XGBoost and LightGBM.
2. **Ordered boosting**: CatBoost 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 over-fitting.
3. **Native feature support**: CatBoost supports all kinds of features be it numeric, categorical, or text and saves time and effort of preprocessing.

Catboost Encoder

- Catboost overcomes the target leakage by introducing time into dataset - the order of the observations.
- $\hat{x}_i^k = \frac{\sum_{j=0}^{j \leq i}(y_j \cdot (x_j == k)) - y_i + prior}{\sum_{j=0}^{j \leq i}(x_j == k) + 1}$
- To prevent the over-fitting, the process is repeated several times on shuffled dataset and results are averaged.
- Catboost "on-the-fly" encoding is one of the core advantages of CatBoost.

```
CBE_encoder = CatBoostEncoder()
train_cbe =
    CBE_encoder.fit_transform(train[feature_list],
    target)
test_cbe = CBE_encoder.transform(test[feature_list])
```

Catboost Encoder

- Catboost overcomes the target leakage by introducing time into dataset - the order of the observations.
- $\hat{x}_i^k = \frac{\sum_{j=0}^{j \leq i} (y_j \cdot (x_j == k)) - y_i + prior}{\sum_{j=0}^{j \leq i} (x_j == k) + 1}$
- To prevent the over-fitting, the process is repeated several times on shuffled dataset and results are averaged.
- Catboost "on-the-fly" encoding is one of the core advantages of CatBoost.

```
CBE_encoder = CatBoostEncoder()
train_cbe =
    CBE_encoder.fit_transform(train[feature_list],
    target)
test_cbe = CBE_encoder.transform(test[feature_list])
```

Catboost Encoder

- Catboost overcomes the target leakage by introducing time into dataset - the order of the observations.
- $\hat{x}_i^k = \frac{\sum_{j=0}^{j \leq i}(y_j \cdot (x_j == k)) - y_i + prior}{\sum_{j=0}^{j \leq i}(x_j == k) + 1}$
- To prevent the over-fitting, the process is repeated several times on shuffled dataset and results are averaged.
- Catboost "on-the-fly" encoding is one of the core advantages of CatBoost.

```
CBE_encoder = CatBoostEncoder()
train_cbe =
    CBE_encoder.fit_transform(train[feature_list],
    target)
test_cbe = CBE_encoder.transform(test[feature_list])
```

Catboost Encoder

- Catboost overcomes the target leakage by introducing time into dataset - the order of the observations.
- $\hat{x}_i^k = \frac{\sum_{j=0}^{j \leq i}(y_j \cdot (x_j == k)) - y_i + prior}{\sum_{j=0}^{j \leq i}(x_j == k) + 1}$
- To prevent the over-fitting, the process is repeated several times on shuffled dataset and results are averaged.
- Catboost "on-the-fly" encoding is one of the core advantages of CatBoost.

```
CBE_encoder = CatBoostEncoder()
train_cbe =
    CBE_encoder.fit_transform(train[feature_list],
    target)
test_cbe = CBE_encoder.transform(test[feature_list])
```

Catboost Encoder

- Catboost overcomes the target leakage by introducing time into dataset - the order of the observations.
- $\hat{x}_i^k = \frac{\sum_{j=0}^{j\leq i}(y_j \cdot (x_j == k)) - y_i + prior}{\sum_{j=0}^{j\leq i}(x_j == k) + 1}$
- To prevent the over-fitting, the process is repeated several times on shuffled dataset and results are averaged.
- Catboost "on-the-fly" encoding is one of the core advantages of CatBoost.

```
CBE_encoder = CatBoostEncoder()
train_cbe =
    CBE_encoder.fit_transform(train[feature_list],
    target)
test_cbe = CBE_encoder.transform(test[feature_list])
```

Feature Processing

- **Numerical Featuers**: The same way like the others - i.e. XGBoost.
- **Categorical Features**: Supported One-Hot Encoding and so-called CatBoost encoder. Also, greedy search for combinations. CatBoost automatically combines categorical features, most times two or three.
- **Text features**: CatBoost also handles text features (containing regular text) by providing inherent text preprocessing using Bag-of-Words (BoW), Naive -Bayes, and BM-25 (for multiclass) to extract words from text data, create dictionaries (letter, words, grams), and transform them into numeric features.

Feature Processing

- **Numerical Featuers**: The same way like the others - i.e. XGBoost.
- **Categorical Features**: Supported One-Hot Encoding and so-called CatBoost encoder. Also, greedy search for combinations. CatBoost automatically combines categorical features, most times two or three.
- **Text features**: CatBoost also handles text features (containing regular text) by providing inherent text preprocessing using Bag-of-Words (BoW), Naive -Bayes, and BM-25 (for multiclass) to extract words from text data, create dictionaries (letter, words, grams), and transform them into numeric features.

Feature Processing

- **Numerical Featuers**: The same way like the others - i.e. XGBoost.
- **Categorical Features**: Supported One-Hot Encoding and so-called CatBoost encoder. Also, greedy search for combinations. CatBoost automatically combines categorical features, most times two or three.
- **Text features**: CatBoost also handles text features (containing regular text) by providing inherent text preprocessing using Bag-of-Words (BoW), Naive -Bayes, and BM-25 (for multiclass) to extract words from text data, create dictionaries (letter, words, grams), and transform them into numeric features.

CatBoost Features 2

5. **Ranking**: Ranking can be broadly done under three objective functions: Pointwise, Pairwise, and Listwise.
6. CatBoost's ranking mode variations:
   - Ranking (YetiRank, YetiRankPairwise)
   - Pairwise (PairLogit, PairLogitPairwise)
   - Ranking + Classification (QueryCrossEntropy)
   - Ranking + Regression (QueryRMSE)
   - Select top 1 candidate (QuerySoftMax)
7. CatBoost also provides ranking benchmarks comparing CatBoost, XGBoost and LightGBM with different ranking variations.
8. **The Usual Ones**: Speed, Feature Importance, Model Analysis (SHAP)

CatBoost Features 2

5. **Ranking**: Ranking can be broadly done under three objective functions: Pointwise, Pairwise, and Listwise.
6. CatBoost's ranking mode variations:
   - Ranking (YetiRank, YetiRankPairwise)
   - Pairwise (PairLogit, PairLogitPairwise)
   - Ranking + Classification (QueryCrossEntropy)
   - Ranking + Regression (QueryRMSE)
   - Select top 1 candidate (QuerySoftMax)
7. CatBoost also provides ranking benchmarks comparing CatBoost, XGBoost and LightGBM with different ranking variations.
8. **The Usual Ones**: Speed, Feature Importance, Model Analysis (SHAP)

CatBoost Features 2

5. **Ranking**: Ranking can be broadly done under three objective functions: Pointwise, Pairwise, and Listwise.
6. CatBoost's ranking mode variations:
   - Ranking (YetiRank, YetiRankPairwise)
   - Pairwise (PairLogit, PairLogitPairwise)
   - Ranking + Classification (QueryCrossEntropy)
   - Ranking + Regression (QueryRMSE)
   - Select top 1 candidate (QuerySoftMax)
7. CatBoost also provides ranking benchmarks comparing CatBoost, XGBoost and LightGBM with different ranking variations.
8. **The Usual Ones**: Speed, Feature Importance, Model Analysis (SHAP)

CatBoost Features 2

5. **Ranking**: Ranking can be broadly done under three objective functions: Pointwise, Pairwise, and Listwise.
6. CatBoost's ranking mode variations:
   - Ranking (YetiRank, YetiRankPairwise)
   - Pairwise (PairLogit, PairLogitPairwise)
   - Ranking + Classification (QueryCrossEntropy)
   - Ranking + Regression (QueryRMSE)
   - Select top 1 candidate (QuerySoftMax)
7. CatBoost also provides ranking benchmarks comparing CatBoost, XGBoost and LightGBM with different ranking variations.
8. **The Usual Ones**: Speed, Feature Importance, Model Analysis (SHAP)

Conclusion

Used Dataset "Cat-in-Dat".

Description	**Training Time**	**Prediction Time**	**ROC AUC Score**
Default Random Forest	5.173537	0.264041	0.600149
Default LightGBM without categorical support	1.026472	0.072304	0.635199
Default LightGBM with categorical support	2.491392	0.073969	0.644861
Default XGBoost	6.096581	0.017769	0.649817
Default Catboost without categorical support	18.193569	0.022442	0.655684
Default Catboost with categorical support	170.324903	0.296049	0.673017

The End

Thank you for your attention and patience.