

# データサイエンティスト(仮)

元素粒子論博士。今はデータサイエンティスト（仮）。

## プロフィール



id:tekenuko

読者になる

66

## 検索

記事を検索

## リンク

はてなブログ

ブログをはじめる

週刊はてなブログ

はてなブログPro

## 注目記事

[Pythonでデータ分析：線形回帰モデル](#)[Pythonでデータ分析：ランダムフォレスト](#)[Pythonでデータ分析：XGboost](#)[Rでスパースモデリング：Elastic Net回帰についてまとめてみる](#)[Pythonでデータ分析：imbalanced-learnで不均衡データのサンプリングを行う](#)[KerasでDeep Learning：CNNを組んでみる](#)[KerasでDeep Learning：KerasでMNISTデータを扱ってみる](#)[KerasでDeep Learning：とりあえずネットワークを組んでみる](#)[KerasでDeep Learning：LSTMで日経平均株価を予測してみる](#)

## Pythonでデータ分析 : Catboost

Python Catboost

0

5

ツイート



送る

シェア

2017-10-13

## 導入

2017年7月に、ロシアのGoogleと言われている（らしい）Yandex社から、Catboostと呼ばれるGradient Boostingの機械学習ライブラリが公開されています。

### CatBoost - state-of-the-art open-source gradient boosting library with categorical featur...

CatBoost - state-of-the-art open-source gradient boosting library with categorical features support, <https://catboost.yandex/> #catboost

 catboost.yandex **2 users**[catboost.yandex](https://catboost.yandex/)

ここ何ヶ月か調整さんになっていて分析から遠ざかりがちになりやすくなっていたのですが、手を動かしたい欲求が我慢できなくなってきたので、いい機会だと思って触ってみることにしました。

## インストール

anacondaなどでPythonを導入していることを前提にします。その場合、おそらくpipで一瞬でインストールできます。

```
$ pip install catboost
```

## 使用データセット：Titanic

Titanic号の乗客の情報および生存・死亡のデータを用いて、どの程度の精度のモデルができるかを試してみます。

データセットは、Kaggleのページからダウンロードしました。

Titanic: Machine Learning from Disaster | Kaggle

train.csvとtest.csvの2つを利用します。train.csvはSurvived（生存なら1、死亡なら0）の列があるデータ、test.csvはSurvivedの列が無いデータになっています。train.csvを用いてモデルを作成し、test.csvからSurvivedの予測値を出し、Kaggleにsubmitすることで、モデルの精度(今回はAccuracy)を見ます。

train.csvのデータはdf\_raw、test.csvのデータはdf\_raw\_testという名前で読み込んでおきます。

### データ加工（笑）

Titanicのデータは、いくつか欠損がある列があるのですが、今回は雑に-999で埋めます。また、乗客の名前などの非構造データもいくつかあるのですが、それらは予めインデックスを保持しておき、モデリングの際によしなにカテゴリカル変数として処理してもらうことにします（Catboostはそういった機能があったので、挙動確認も込めてあえて特徴量加工を頑張らないことにしました）。

```
# pandasやnumpyはインポート済み
# 欠損値補完
df_raw.fillna(-999, inplace=True)

# 説明変数・目的変数
X = df_raw.drop('Survived', axis = 1)
y = df_raw['Survived']

# カテゴリカル変数として扱う列指定
categorical_features_indices = np.where(X.dtypes != np.float)[0]
```

訓練用と検証（Validation）用にデータを分割しておきます。

```
from sklearn.model_selection import train_test_split
X_train, X_validation, y_train, y_validation = train_test_split(X, y, test_size = 0
```



### モデリング

今回はクラス分類なので、catboost.CatBoostClassifierという関数を利用します。

```
import catboost
mod = catboost.CatBoostClassifier(iterations=5000,
                                  calc_feature_importance = True,
                                  use_best_model=True,
                                  eval_metric = 'Accuracy' )

mod.fit(X_train, y_train,
        cat_features=categorical_features_indices,
        eval_set=(X_validation, y_validation),
        plot=True)
```

オプションについては、例えば以下のページにあります。

Rでスパースモデリング：Adaptive Lasso

#### 最新記事

Ubuntu16.04LTSにSparkを入れる

Ubuntu 16.04LTSにTexを入れる

NN論文の読み会でまた発表した

PytorchでDeep Learning：CPU onlyでインストールする際のメモ


Pythonでデータ分析：Auto-sklearnについてのメモ

#### カテゴリー

- Python (22)
- Deep Learning (10)
- R (9)
- トポロジカルデータアナリス (8)
- scikit-learn (5)
- C++ (5)
- 不定期 (5)
- Keras (5)
- スパースモデリング (4)
- 強化学習 (2)
- XGboost (2)
- auto-sklearn (2)
- アソシエーション分析 (1)
- 時系列解析 (1)
- TDA (1)
- 状態空間モデル (1)
- Stan (1)
- Spark (1)
- RStudio (1)
- Boost (1)
- Catboost (1)
- FastBDT (1)
- Gluon (1)
- LaTeX (1)
- Pandas (1)
- Prophet (1)

## Training parameters

Several parameters have aliases. For example, the iterations parameter has the following synonyms: num\_boost\_round, n\_estimators, num\_trees. Simultaneous usage of different names of one parameter raises an error.

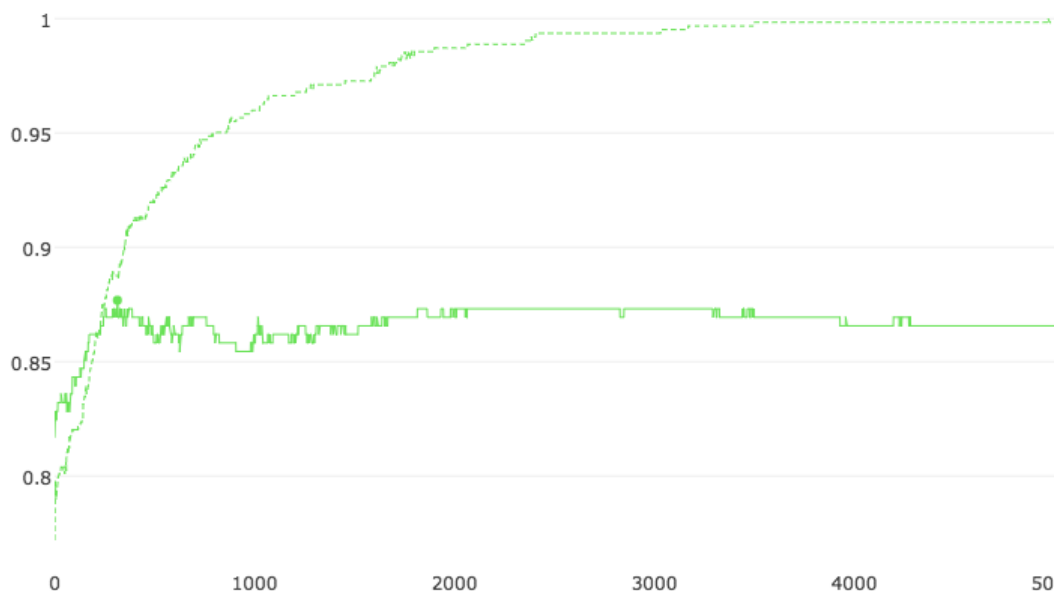
 tech.yandex.com

tech.yandex.com

今回は、CatBoostClassifierにはiterations：何回学習と検証を繰り返すか、calc\_feature\_importance：変数重要度を計算するか、use\_best\_model：iterationの中で一番検証用の指標が良いものを選ぶか、eval\_metric：評価指標をオプションとして選んでいます。

モデル作成は、sklearnのようにfitメソッドで行います。cat\_featuresでカテゴリカル変数として扱うカラムを指定し、eval\_setで最初のvalidationに使用するデータを指定します。

実行後は、以下のような図がプロットされます。



これは、訓練・検証用データに対してAUCをプロットしたものです。点線が訓練用、普通の線が検証用なので、結構過学習しちゃってます（汗）。検証用データで一番良い場合はaccuracyが0.88程度のです。

```
from sklearn.metrics import accuracy_score
y_val_pred = mod.predict(X_validation)
print('accuracy : %.2f' % accuracy_score(y_val_pred, y_validation))

# 出力
accuracy : 0.88
```

テストデータを用いて、予測してみます。これはsklearnの場合と同様に、predictメソッドで出力できます。ただし、floatとして出て来るので、出力後にintに変換します。こうしておかないと、Kaggleにsubmitしたときにscoreが0になります。

```
# テストデータはdf_raw_test
★ df_raw_test.fillna(-999, inplace = True)
y_test = mod.predict(df_raw_test)
```

[PyStan \(1\)](#)

[Pytorch \(1\)](#)

[R Notebook \(1\)](#)

[自己紹介 \(1\)](#)

## 月別アーカイブ

▼ [2018 \(4\)](#)  
[2018 / 6 \(2\)](#)  
[2018 / 2 \(1\)](#)  
[2018 / 1 \(1\)](#)

▶ [2017 \(23\)](#)

▶ [2016 \(25\)](#)

▶ [2015 \(1\)](#)

## 関連記事

[PytorchでDeep Learning : CPU onlyでインストールする際のメモ](#)

[Pythonでデータ分析 : Auto-sklearnについてのメモ](#)

[Pythonでデータ分析 : imbalanced-learnで不均衡データのサンプリングを行う](#)

[Pythonでデータ分析 : Prophetを使ってビットコインの予測（笑）をやってみる](#)

[Pythonでデータ分析 : 主成分分析 \(PCA\) による異常検知](#)

```
est = DataFrame()
test['PassengerId'] = df_raw_test['PassengerId']
test['Survived'] = y_test
# 予測値をintに変換
test['Survived'] = test['Survived'].astype('int')
# 保存
test.to_csv('out/benchmark_catboost.csv', index = False)
```

この結果をKaggleにsubmitすると....

benchmark_catboost.csv 38 minutes ago by tekenuko benchmark(catboost)	0.80382	<input type="checkbox"/>
---	---------	--------------------------

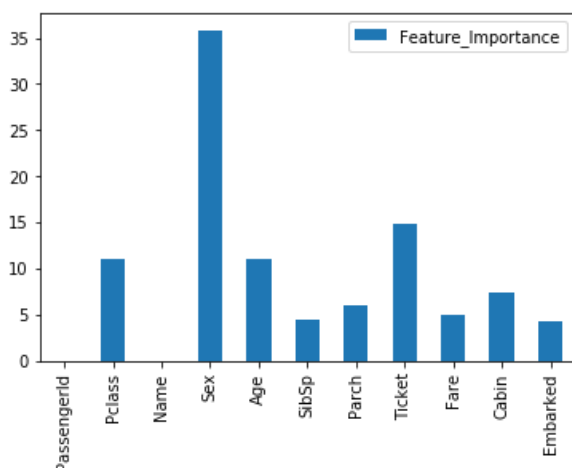
ほぼほぼ何も頑張っていないのにaccuracyが0.8を超えました。ただ、過学習しているモデルなので、より安定して精度を出したかったらパラメータ調整などをおこなって汎化性能を上げる必要があるかと思います。そういった努力は今後の課題にします。

## 変数重要度

Catboostはfeature\_importances\_で変数の重要度も出力することができます（ソートはされていない）。ざっくり可視化しておきます。

```
FI = DataFrame()
FI = FI.append(mod.feature_importances_)
FI.index = X.columns
FI.columns = ['Feature_Importance']

# 可視化
import matplotlib.pyplot as plt
%matplotlib inline
FI.plot(kind='bar')
```



Pclassは乗客の社会的地位、Sexは性別、Ageは年代で、地位の高い女性や子供が優先的に救出されたという歴史的経緯が概ね反映されています。Ticketが効いているのはちょっと難しいですが。

## Next Step

大雑把には使い方が分かったので、今後はGrid Searchなどを詰めていって、より使いこなせるようにしていこうと思います。

tekenuko 1年前



0

5  
シェア

ツイート



送る

関連記事

2017-12-11  
Pythonでデータ分析 : imbalanced-learnで不均衡データのサンプリングを行う  
導入 クラス分類、例えば0 : 負例と1 : 正例の二値分類を行う際に...

2017-10-18  
Pythonでデータ分析 : Prophetを使ってビットコインの予測（笑）をやってみる  
導入 直近、これといって緊急の業務がなく、「自分の時間だ何勉...

2017-10-14  
Pythonでデータ分析 : PyStanで線形回帰モデル  
導入 ベイズ推定を行うための道具として、マルコフ連鎖モンテカ...

2017-07-23  
KerasでDeep Learning : CNNを組んでみる  
導入 前はMNISTデータに対してネットワークを構築して、精度...

2017-07-05  
KerasでDeep Learning : KerasでMNISTデータを扱ってみる  
導入 前は人工データを用いたネットワーク構築について紹介し...

コメントを書く

« Pythonでデータ分析 : PyStan  
で線形回帰モ...

Memo : MacOS Sierraで  
XGboostをpipで入れる »

はてなブログをはじめよう！

tekenukoさんは、はてなブログを使っています。あなたもはてなブログをはじめてみませんか？

[はてなブログをはじめる（無料）](#)

はてなブログとは

 データサイエンティスト(仮)

Powered by Hatena Blog | [ブログを報告する](#)