# PyCaretでみる機械学習のlow code化

みんなのPython勉強会＃73

9/8, 2021

辻真吾（@tsjshg）

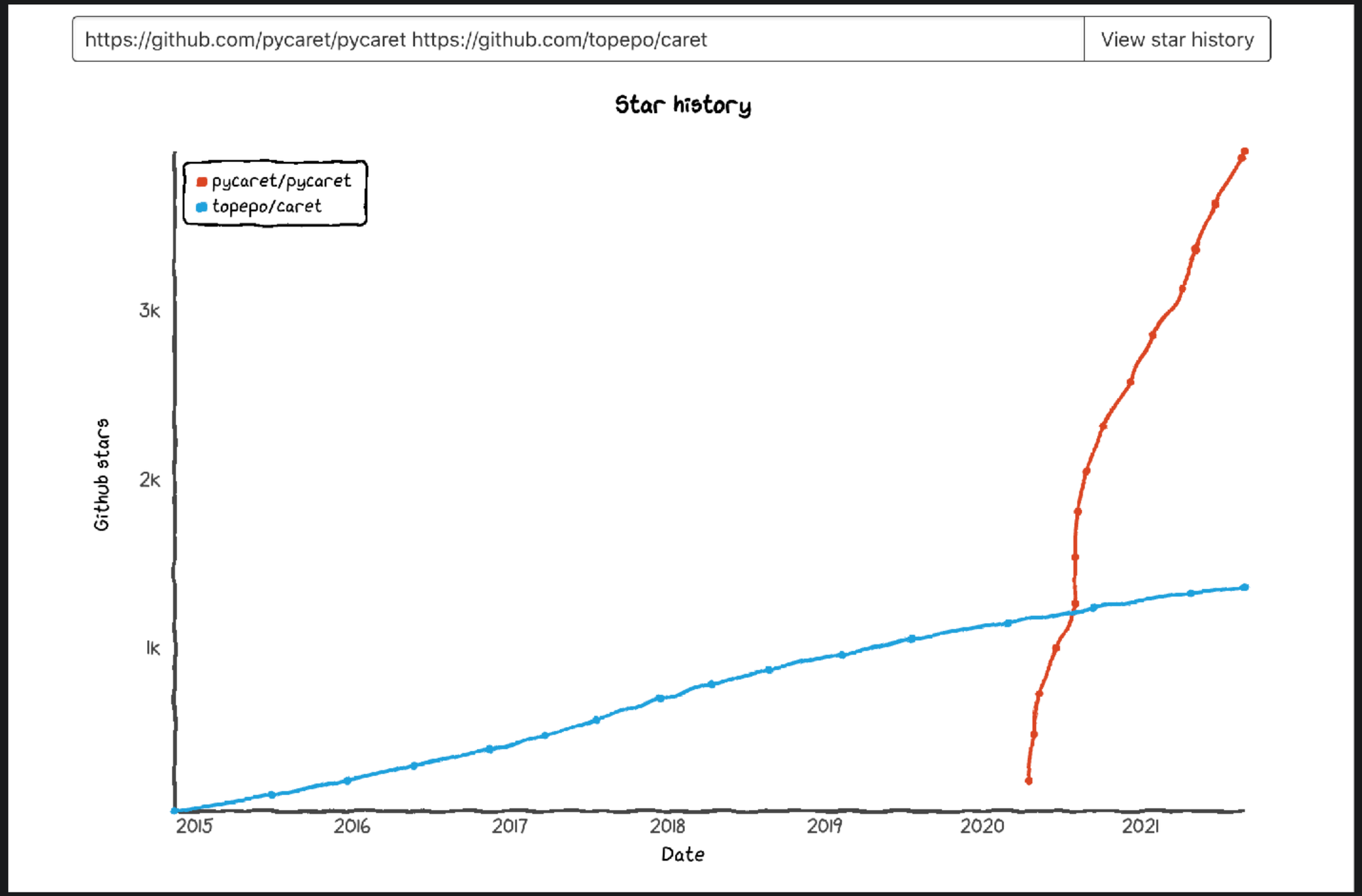
# おまえ、誰よ？

- 辻真吾（つじしんご）
- 大学の研究所に勤めています（バイオ→エネルギー）
  - 応用データサイエンスという分野を作りたい
- 小学生のころ（MSX2）からコンピュータが好き
  - BASIC, LOGO → C/C++ → Java → Python
- 執筆企画は残り1つ（今冬には出せる予定）

# PyCaretのはなし

- AutoML（Automated Machine Learning）のためのPythonライブラリ
- （話すこと）PyCaretの概要といまの時点では注意したほうがいいこと
- （話せないこと）機械学習の基本

- もともとRにCaretというAutoMLライブラリがあった
  - 便利でいいなーと思っていたら、いつのまにかPython版が登場




https://star-history.t9t.io/#pycaret/pycaret&topepo/caret

https://pycaret.org/


ドキュメントにExample Notebooksがあるのでそのまま実行も可能

# 今日のデータ：クレジットカードの退会者を予想するデータセット

| CLIENTNUM | Attrition_Flag | Customer_Age | Gender | Dependent_count | Education_Level | Marital_Status | Income_Category | Card_Category | Months_on_book | Total_R |
|---|---|---|---|---|---|---|---|---|---|---|
| 768805383 | Existing Customer | 45 | M | 3 | High School | Married | $60K-80K | Blue | 39 | |
| 818770008 | Existing Customer | 49 | F | 5 | Graduate | Single | Less than $40K | Blue | 44 | |
| 713982108 | Existing Customer | 51 | M | 3 | Graduate | Married | $80K-120K | Blue | 36 | |
| 769911858 | Existing Customer | 40 | F | 4 | High School | Unknown | Less than $40K | Blue | 34 | |
| 709106358 | Existing Customer | 40 | M | 3 | Uneducated | Married | $60K-80K | Blue | 21 | |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| 772366833 | Existing Customer | 50 | M | 2 | Graduate | Single | $40K-60K | Blue | 40 | |
| 710638233 | Attrited Customer | 41 | M | 2 | Unknown | Divorced | $40K-60K | Blue | 25 | |
| 716506083 | Attrited Customer | 44 | F | 1 | High School | Married | Less than $40K | Blue | 36 | |
| 717406983 | Attrited Customer | 30 | M | 2 | Graduate | Unknown | $40K-60K | Blue | 36 | |
| 714337233 | Attrited Customer | 43 | F | 2 | Graduate | Married | Less than $40K | Silver | 25 | |

10127 rows × 22 columns

https://www.kaggle.com/sakshigoyal7/credit-card-customers

# さっそく使ってみる

## データのちょっとした前処理

```
import pandas as pd

data = pd.read_csv('BankChurners.csv', index_col=0)
# 最後の2列にNaive Bayesのスコアが入っていて答えになってしまうので、削除する。
drop_cols = data.columns[-2]
data.drop(drop_cols, axis=1, inplace=True)
data
```

## セットアップ（目的変数の設定）

```
from pycaret.classification import *

clf1 = setup(data, target='Attrition_Flag')
```

## モデル探し

```
best_model = compare_models()
```

PyCaretのコードはこの3行

# いきなりこれだけの結果がでる！

```
best_model
```

| | Model | Accuracy | AUC | Recall | Prec. | F1 | Kappa | MCC | TT (Sec) |
|---|---|---|---|---|---|---|---|---|---|
| **lightgbm** | Light Gradient Boosting Machine | 0.9698 | 0.9936 | 0.9856 | 0.9784 | 0.9820 | 0.8891 | 0.8895 | 0.0680 |
| **gbc** | Gradient Boosting Classifier | 0.9629 | 0.9885 | 0.9871 | 0.9690 | 0.9780 | 0.8605 | 0.8622 | 0.4230 |
| **ada** | Ada Boost Classifier | 0.9523 | 0.9844 | 0.9770 | 0.9662 | 0.9716 | 0.8236 | 0.8243 | 0.1260 |
| **rf** | Random Forest Classifier | 0.9458 | 0.9836 | 0.9870 | 0.9500 | 0.9682 | 0.7874 | 0.7940 | 0.2190 |
| **dt** | Decision Tree Classifier | 0.9335 | 0.8786 | 0.9608 | 0.9597 | 0.9602 | 0.7590 | 0.7592 | 0.0300 |
| **et** | Extra Trees Classifier | 0.9190 | 0.9634 | 0.9897 | 0.9195 | 0.9533 | 0.6535 | 0.6800 | 0.1900 |
| **lda** | Linear Discriminant Analysis | 0.9121 | 0.9312 | 0.9647 | 0.9324 | 0.9482 | 0.6580 | 0.6636 | 0.0380 |
| **ridge** | Ridge Classifier | 0.9041 | 0.0000 | 0.9834 | 0.9091 | 0.9448 | 0.5839 | 0.6124 | 0.0190 |
| **lr** | Logistic Regression | 0.8997 | 0.9177 | 0.9665 | 0.9177 | 0.9414 | 0.5935 | 0.6044 | 0.1430 |
| **nb** | Naive Bayes | 0.8952 | 0.8886 | 0.9450 | 0.9305 | 0.9376 | 0.6086 | 0.6105 | 0.0230 |
| **knn** | K Neighbors Classifier | 0.8912 | 0.8814 | 0.9530 | 0.9196 | 0.9360 | 0.5756 | 0.5802 | 0.0450 |
| **svm** | SVM - Linear Kernel | 0.7665 | 0.0000 | 0.8487 | 0.8785 | 0.8409 | 0.1876 | 0.2317 | 0.0540 |
| **qda** | Quadratic Discriminant Analysis | 0.4402 | 0.5141 | 0.4035 | 0.8423 | 0.5302 | 0.0163 | 0.0216 | 0.0260 |

# なにをやってくれているか？（抜粋）

- 全体（10127）を訓練データ（7088）とテストデータ（3039）に分割
- 変数変換と欠損値の処理
  - 文字列で入っているカテゴリー変数をone-hot encoding
  - ちなみにこのデータには欠損値はないです
- 各種機械学習アルゴリズムの訓練と、10-fold cross validationによる性能評価
  - ここではテストデータは使われない

テストデータで予測

```
predict_model(best_model)
```

最初のセットアップで分割しておいてくれたデータを自動的に使ってくれる

予測の結果

| | Model | Accuracy | AUC | Recall | Prec. | F1 | Kappa | MCC |
|---|---|---|---|---|---|---|---|---|
| 0 | Light Gradient Boosting Machine | 0.9750 | 0.9937 | 0.9869 | 0.9838 | 0.9853 | 0.9005 | 0.9006 |

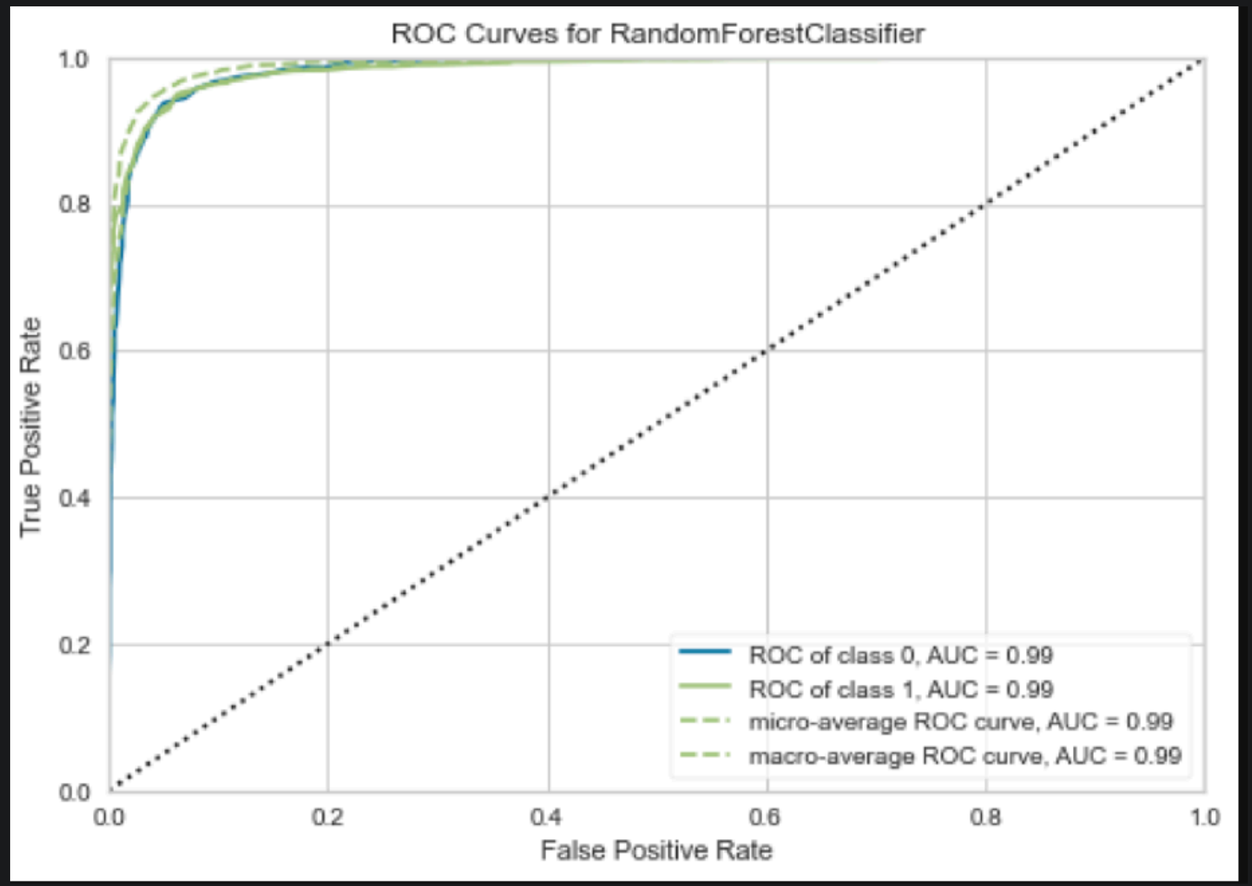
各サンプルの結果

| :ts_Count_12_mon_4 | Contacts_Count_12_mon_5 | Contacts_Count_12_mon_6 | Attrition_Flag | Label | Score |
|---|---|---|---|---|---|
| 1.0 | 0.0 | 0.0 | Existing Customer | Existing Customer | 0.9995 |
| 1.0 | 0.0 | 0.0 | Attrited Customer | Attrited Customer | 0.9904 |
| 0.0 | 0.0 | 0.0 | Existing Customer | Existing Customer | 0.9994 |
| 1.0 | 0.0 | 0.0 | Existing Customer | Existing Customer | 0.9996 |
| 0.0 | 0.0 | 0.0 | Existing Customer | Existing Customer | 0.9538 |

# plot_modelが便利

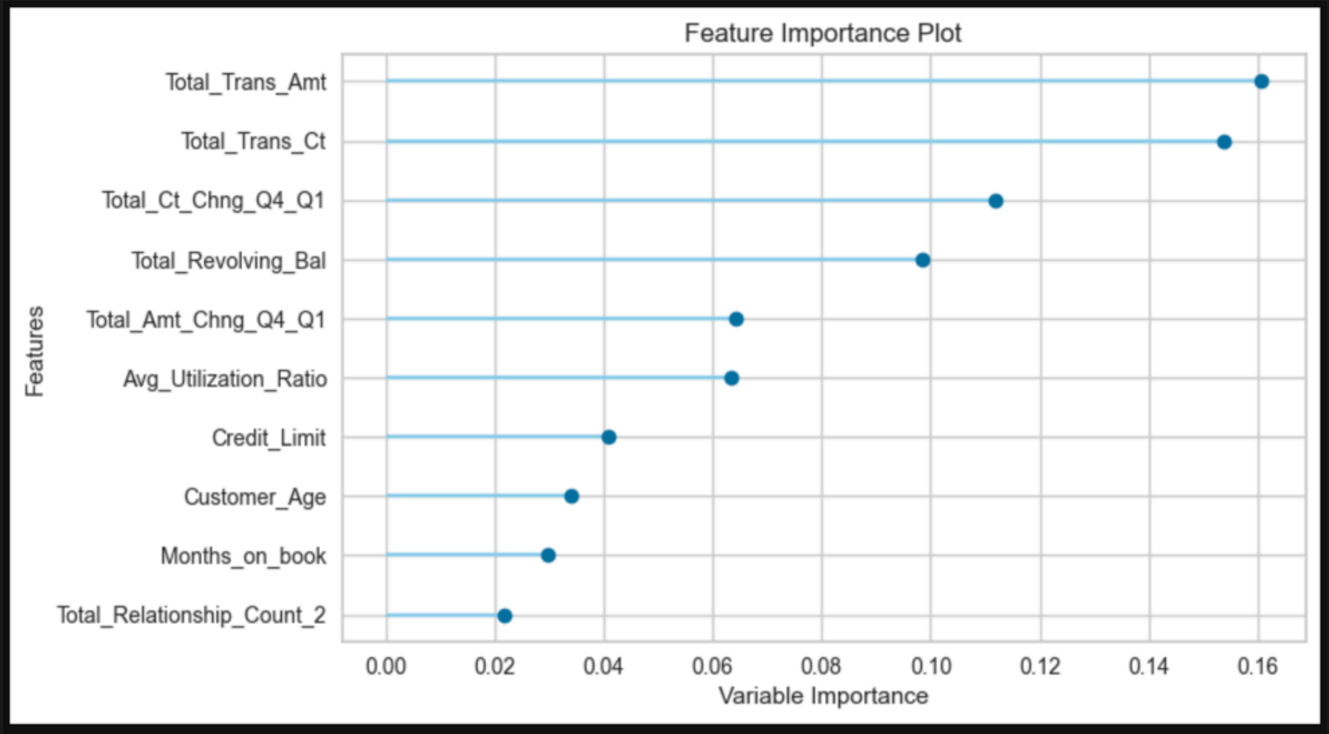
モデルを指定してROC曲線を描く例

```
rf = create_model('rf')
plot_model(rf)
```



# 説明変数の重要性（feature importance）をプロット

```
plot_model(rf, plot='feature')
```



# すこし手を入れたいと思ったらget_config関数

pycaret.classification.get_config(*variable: str*)

This function retrieves the global variables created when initializing the `setup` function. Following variables are accessible:

- X: Transformed dataset (X)
- y: Transformed dataset (y)
- X_train: Transformed train dataset (X)
- X_test: Transformed test/holdout dataset (X)
- y_train: Transformed train dataset (y)
- y_test: Transformed test/holdout dataset (y)
- seed: random state set through session_id
- prep_pipe: Transformation pipeline
- fold_shuffle_param: shuffle parameter used in Kfolds
- n_jobs_param: n_jobs parameter used in model training
- html_param: html_param configured through setup
- create_model_container: results grid storage container
- master_model_container: model storage container
- display_container: results display container

まだまだ沢山ある

https://pycaret.readthedocs.io/en/latest/api/classification.html#pycaret.classification.get_config

# ちょっとだけパイプラインの中身を見てみる

```
type(get_config("prep_pipe"))
```

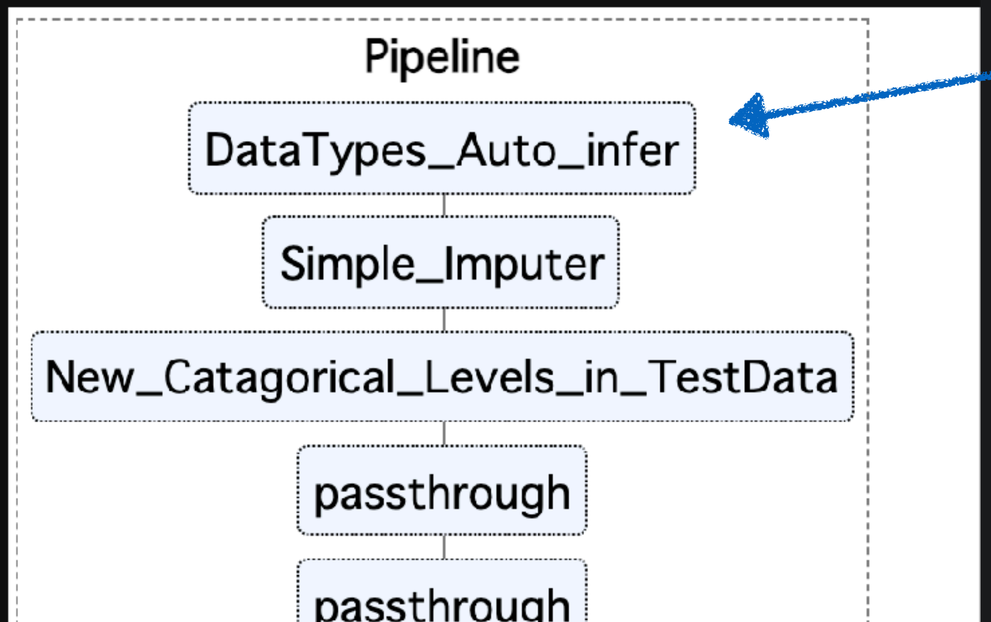
```
sklearn.pipeline.Pipeline
```

パイプライン自体はscikit-learnのインスタンス

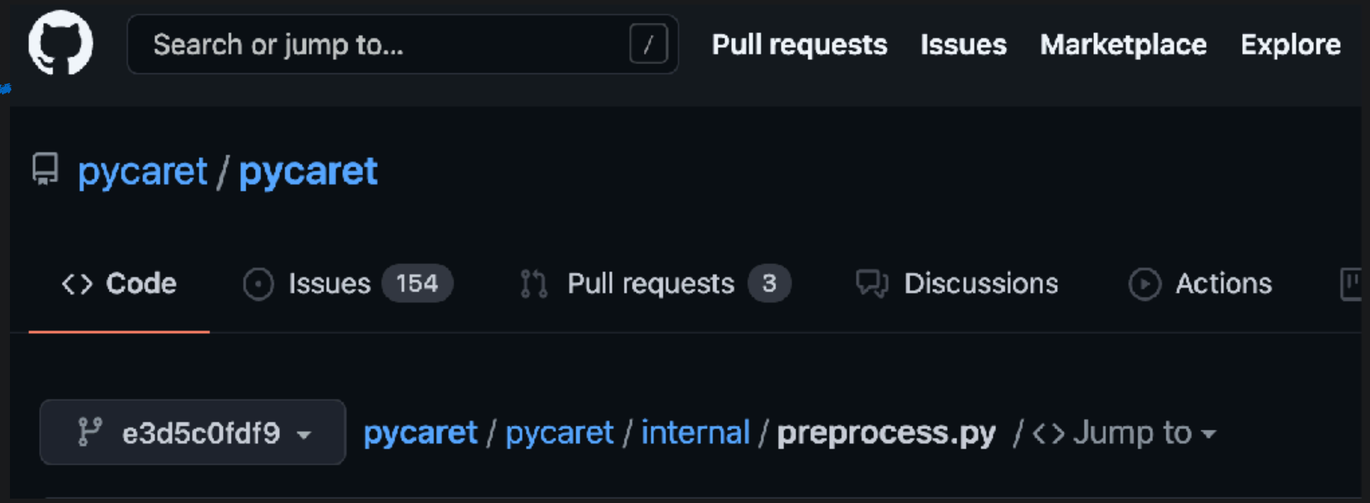
```
from sklearn import set_config

set_config(display="diagram")
```

```
get_config("prep_pipe")
```




パイプライン可視化のための準備




内部ではPyCaret製のクラスが使われている

## X_testを使った予測のやり方

```
predict_model(best_model)
```

これだけでできてしまうけど・・・

```
# 最初のセットアップで作られたテストデータ
X_test = get_config('X_test')
# こうしてもOK
best_model.predict(X_test)
```

```
array([1, 0, 1, ..., 0, 1, 1])
```

```
# 元のデータと同じ形のテストデータを使うときは、以下のコード
predict_model(best_model, data=X_test_original)
```

X_test_originalをパイプラインで前処理してからモデルに入力してくれる

## このほかいろいろ

- 個別モデルのパラメータチューニング
- モデルのデプロイ
  - ドキュメントにはAWSの例
- もちろん回帰にも対応
- クラスタリングもできる
  - Plotlyとの連携してすごい可視化が簡単にできる！（やってみる価値あります）

インストール

**pip install pycaret**

# 仮想環境を作ってから

- とにかく依存するパッケージが多い
- パッケージごとにバージョン指定があるので、すでにインストールされているパッケージのバージョンが変わってしまう可能性も
- 公式にはPython3.6〜3.8までの対応
- pip install pycaretではjupyterlabとpandasは入らないので追加必要

仮想環境についてはみんなのPython勉強会#67「Pythonのインストールと環境設定」を参考にしてください
http://tsjshg.info/20210310_TSUJI.pdf

# ところでそこのあなた

pipを利用したあと最後の行にときどき出るこのwarning
見て見ぬふりしていませんか？

```
WARNING: You are using pip version 21.1.1; however, version 21.2.4 is available.
You should consider upgrading via the '/Library/Frameworks/Python.framework/Versions/3.9/bin/python3.9
 -m pip install --upgrade pip' command.
```

# PyCaretのインストールは
# pipのバージョンを21.xにしてから

# python3 -m pip install -U pip

WindowsではPython3の代わりにpyランチャーが便利

# パッケージの依存関係を解決する方法がpipバージョン21から変わっている

普段pipを使っていてこんなエラーが出ていることに気が付いたことはないでしょうか？

```
    Running setup.py install for databricks-cli ... done
    Running setup.py install for pynndescent ... done
    Running setup.py install for umap-learn ... done
    Running setup.py install for cufflinks ... done
ERROR: After October 2020 you may experience errors when installing or updating packages. This is because
 pip will change the way that it resolves dependency conflicts.

We recommend you use --use-feature=2020-resolver to test your packages with the new resolver before it be
comes the default.

pyldavis 3.3.1 requires numpy>=1.20.0, but you'll have numpy 1.19.5 which is incompatible.
Successfully installed Boruta-0.3 Flask-2.0.1 IPython-7.27.0 Mako-1.1.5 MarkupSafe-2.0.1 PyWavelets-1.1.1
 PyYAML-5.4.1 Send2Trash-1.8.0 Werkzeug-2.0.1 alembic-1.4.1 appnope-0.1.2 argon2-cffi-21.1.0 attrs-21.2.0
 backcall-0.2.0 bleach-4.1.0 blis-0.7.4 bottleneck-1.3.2 catalogue-1.0.0 certifi-2021.5.30 cffi-1.14.6 ch
arset-normalizer-2.0.4 click-8.0.1 cloudpickle-1.6.0 colorlover-0.3.0 cufflinks-0.17.3 cycler-0.10.0 cymem
```

21から`--use-feature=2020-resolver`がデフォルトの動き

# まだまだ開発途上

# compare_models()のあと

```
/Users/shingo/.venv/pycaret/lib/python3.8/site-packages/sklearn/utils/deprecation.
py:101: FutureWarning: Attribute standard_coef_ was deprecated in version 0.23 and
will be removed in 0.25.
  warnings.warn(msg, category=FutureWarning)
/Users/shingo/.venv/pycaret/lib/python3.8/site-packages/sklearn/utils/deprecation.
py:101: FutureWarning: Attribute standard_intercept_ was deprecated in version 0.2
3 and will be removed in 0.25.
  warnings.warn(msg, category=FutureWarning)
/Users/shingo/.venv/pycaret/lib/python3.8/site-packages/sklearn/utils/deprecation.
py:101: FutureWarning: Attribute average_coef_ was deprecated in version 0.23 and
will be removed in 0.25.
  warnings.warn(msg, category=FutureWarning)
/Users/shingo/.venv/pycaret/lib/python3.8/site-packages/sklearn/utils/deprecation.
py:101: FutureWarning: Attribute average_intercept_ was deprecated in version 0.23
and will be removed in 0.25.
  warnings.warn(msg, category=FutureWarning)
/Users/shingo/.venv/pycaret/lib/python3.8/site-packages/sklearn/utils/deprecation.
```

同じようなFutureWarningが40個くらい並ぶ

pycaretがなかでsklearnを利用していて、sklearnからの警告

ライブラリ同士が複雑に依存し合う最近のプログラミング環境ではよく見る光景

# n_jobsパラメータにご注意

setupの出力（抜粋）

| | | |
|---|---|---|
| 13 | Shuffle Train-Test | True |
| 14 | Stratify Train-Test | False |
| 15 | Fold Generator | StratifiedKFold |
| 16 | Fold Number | 10 |
| 17 | CPU Jobs | -1 |
| 18 | Use GPU | False |
| 19 | Log Experiment | False |
| 20 | Experiment Name | clf-default-name |

CPUコアあるだけ使う

ちょっと前に謎のエラー（BrokenProcessPool）が出て困った

すこし調べてsetupに n_jobs=1 を渡したら直った

ただしエラーを再現できなかったので、ちょっと不確かな情報（すんません）

# まとめ

- AutoMLはlowcodeな機械学習
- PyCaretはスゴイ便利なライブラリ
  - 教師あり学習では複数のモデルを一括トレーニング
  - ROC曲線など性能評価の可視化もほとんど1行で描ける
- すごい勢いで開発が進んでいるフェーズ
  - インストールは仮想環境
  - エラーや警告が出ても焦らない