Code review kamenia milowego 2 grupy "Moja grupa"

Damian Skowroński | warsztaty badawcze | praca domowa 3

5 maja 2022

1 Wprowadzenie

Grupa zajmuje się pakietem *Auto-Sklearn 2.0* i funkcje do przeprowadzenia benchmarka zawarli w pliku *AutoML-preprocess.py*. Są to:

- fit_model
- preprocess
- cross_valid
- funkcja

Z czego funkcje *fit_automl* i *preprocess* są funkcjami pomocniczymi do *cross_valid* i *funkcja*.

2 Czy ten kod osiąga cel, który postawiono?

Celem było strzorzyć skrypt zawierający funkcję, która przyjumuje zbiór danych objeśniających i kolumnę objaśnianą, i wykonuje na nich predykcję używając systemu AutoML.
Kod spełnia cel, ponieważ taka funkcja istnieje, jest to *funkcja*, w której zachodzi preprocessing i crossvalidacja. Dzięki niej można uzyskać wyniki accuracy i auc, a także modele jakie zostały wytrenowane.

3 Czy w kodzie są jakieś oczywiste błędy logiczne?

Moim zdaniem nie.

4 Czy patrząc na wymagania zawarte podczas prezentacji są one w pełni zaimplementowane?

Nie jestem pewien, czy o to chodzi w tym pytaniu, ale uważam, że plik, w którym mają być liczone i prezentowane rezultaty, czyli *KM_2_AutoML-last-run.ipynb* mógłby mieć policzony score dla każdego ze zbiorów. Oprócz tego nie udało mi się odtworzyć go w tej formie jakiej jest bo nie działa mi funkcja *model_data*, a dokładniej błąd pojawia się w funkcji *cross_valid*.

5 Czy są jakieś obszary, w których kod mógłby zostać poprawiony?

Nie znalazłem takich.

6 Czy dokumentacja i komentarze są wystarczające?

Myślę, że przydałaby się lepsza dokumentacja. Brakuje mi może komentarzy tłumaczących wybrane argumenty w konstruktorze AutoSklearn2Classifier. Oprócz tego są wystraczające. Jest ich niewiele, ale tłumaczą miejsca, które nie są oczywiste. Reszta jest całkiem zrozumiała, więc komentarze nie są potrzebne.

7 Czy udało się odtworzyć zamieszone przykłady w kodzie?

Tak, jednak musiałem troche nad tym posiedzieć. Autosklearn 2.0 działa tylko na systemach linux, więc użyłem do tego google colaba, gdzie do zainstalowania użyłem następującego kodu.

```
!sudo apt-get install build-essential swig
!curl https://raw.githubusercontent.com/automl/auto-sklearn/master/requirements.txt | xargs -n 1 -L 1 pip install
!pip install auto-sklearn
```

Po czym dopiero po zrestartowaniu importy działały. Nie udało mi się natomiast odtworzyć wyników w sposób jaki jest podany w pliku .ipynb, ponieważ wydaję mi się, że autorzy wczytują wytrenowane modele z pickle, do których nie mam dostępu.
Wobec tego sam napisałem do tego kod i odpaliłem go dla zbiorów danych **airlines** i **albert**, czyli jedynych, które były umieszczone w kodzie.

```
[7] import openml

    airlines = openml.datasets.get_dataset(1169)

    target_name = airlines.default_target_attribute
    X, y, categorical_indicator, attribute_names = airlines.get_data()

    y = pd.DataFrame(X[target_name])
    X = X.drop(columns = [target_name])
```

```
[15] models, result = funkcja(X,y)
```

```
[16] result
```

```
{'accuracy': [0.5545523647082816,
  0.5545523647082816,
  0.5545523647082816,
  0.5545441061960028,
  0.554562646000964,
  0.554562646000964,
  0.554562646000964,
  0.554562646000964,
  0.554562646000964,
  0.554562646000964],
 'auc': [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]}
```

```
[22] albert = openml.datasets.get_dataset(41147)

     target_name = albert.default_target_attribute
     X, y, categorical_indicator, attribute_names = albert.get_data()

     y = pd.DataFrame(X[target_name])
     X = X.drop(columns = [target_name])
```

```
[23] models, result = funkcja(X,y)
```

```
[24] result

     {'accuracy': [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5],
      'auc': [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]}
```

Moje wyniki okazały się takie same jak wyniki autorów:

```
score
```

```
{'airlines': {'accuracy': [0.5545523647082816,
   0.5545523647082816,
   0.5545523647082816,
   0.554562646000964,
   0.554562646000964,
   0.554562646000964,
   0.554562646000964,
   0.554562646000964,
   0.554562646000964,
   0.5545441061960028],
  'auc': [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]},
 'albert': {'accuracy': [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5],
  'auc': [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5]}}
```

8 Czy udało się użyć przygotowanych kodów na nowych danych?

Tak, wykorzystałem kod do zbioru **mushroom** i zadziałał dobrze.

```
[25] mushroom = openml.datasets.get_dataset(24)

     target_name = mushroom.default_target_attribute
     X, y, categorical_indicator, attribute_names = mushroom.get_data()

     y = pd.DataFrame(X[target_name])
     X = X.drop(columns = [target_name])
```

```
[26] models, result = funkcja(X,y)
```

```
[27] result
```

```
{'accuracy': [1.0,
  0.998769987699877,
  1.0,
  1.0,
  0.9987684729064039,
  0.9987684729064039,
  1.0,
  1.0,
  0.9987684729064039,
  1.0],
 'auc': [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]}
```

9 Podsumowanie

Uważam, że kod w skrypcie jest dobry, choć mógłby być troche lepiej opisany. Natomiast notatnik, który miał przedstawiać wyniki zawiera tylko dwa zbiory i nie udało mi się go odpalić. Tak naprawdę musiałem napisać to sam.