AutoKeras code review (grupa Tojada)

Importy są nieuporządkowane oraz brak pliku `requirements.txt`

```
import autokeras as ak
from openml import datasets
from sklearn.metrics import accuracy_score,f1_score,roc_auc_score,recall_score,precision_s
from sklearn.model_selection import train_test_split
import pandas as pd
import numpy as np
import json
import time
from tensorflow.keras.models import load_model
```

Porozdzielanie rzeczy może pomóc z czytelnością kodu. Można też poprawić umiejscowienie komentarzy, gdyż czasem nie wiadomo, której linijki dotyczą.

```
def get_datasets(names):
    '''
    Funkcji podaje się listę nazw datasetów, ona szuka je w openml i zwraca je. Zwraca też
    '''

    ds = pd.DataFrame(datasets.list_datasets()).transpose().reset_index(drop = True)
    # zbieranie datasetów z api openmlowego
    matches = ds[np.in1d(ds.name,names)]  # to co się udalo znalezc po nazwie
    matches = matches[matches.status == "active"]   # wywalam nieaktywne
    matches.version = matches.version.astype(int)   # żeby mogło wybrać największą liczbę
    matches = matches.groupby("name").apply(lambda d: d.nlargest(1,columns = "version")) #
    matches.reset_index(drop = True, inplace = True)
    # jak wyszukiwałem po nazwie to nie znajdowało mi 9 datasetów
    unmatched = names[np.where(np.logical_not(np.in1d(names,ds.name)))[0]]    # to są te n
    # zbieram datasety z api openmlowego uzywając ich id
    ds_list = datasets.get_datasets(dataset_ids = matches.did)
    return ds_list, unmatched
```

```
def get_datasets(names):
    '''
    Funkcji podaje się listę nazw datasetów, ona szuka je w openml i zwraca je.
    Zwraca też listę nazw, których nie znalazło.
    '''
    ds = pd.DataFrame(datasets.list_datasets()).transpose().reset_index(drop = True)
    # zbieranie datasetów z api openmlowego

    matches = ds[np.in1d(ds.name,names)]  # to co się udalo znalezc po nazwie
    matches = matches[matches.status == "active"]   # wywalam nieaktywne
```

```
    matches.version = matches.version.astype(int)     # wywalam nieaktywne

    # żeby mogło wybrać największą liczbę
    matches = matches.groupby("name").apply(lambda d: d.nlargest(1,columns = "version"))

    # wybieram te z najnowszą wersją
    matches.reset_index(drop = True, inplace = True)

    # jak wyszukiwałem po nazwie to nie znajdowało mi 9 datasetów
    unmatched = names[np.where(np.logical_not(np.in1d(names,ds.name)))[0]]     # to są te n

    # zbieram datasety z api openmlowego uzywając ich id
    ds_list = datasets.get_datasets(dataset_ids = matches.did)

    return ds_list, unmatched
```

```
def run(X : pd.DataFrame,y : pd.DataFrame):

    start = time.time()

    train_X, test_X, train_y, test_y = train_test_split(X,y,random_state=420)

    clf = ak.StructuredDataClassifier()
    clf.fit(train_X, train_y)

    # następne trzy linijki bo były problemy z typami
    prediction = pd.DataFrame(clf.predict(test_X))
    test_y = test_y.apply(pd.to_numeric, errors='coerce').fillna(test_y)
    prediction = prediction.apply(pd.to_numeric, errors='coerce').fillna(prediction)


    accuracy = accuracy_score(test_y, prediction)
    recall = recall_score(test_y, prediction,average='micro')
    precision = precision_score(test_y, prediction,average='micro')
    f1 = f1_score(test_y, prediction, average='micro')
    # auc = roc_auc_score(test_y,prediction) # trzeba ogarnac pewnie dla multilabelow


    scores_dict = {
        'accuracy' : accuracy,
        'recall' : recall,
        'precision' : precision,
        'f1_micro' : f1
        # ,
        # 'auc' : auc
    }

    end = time.time()
    execution_time = end - start


    return clf, scores_dict, execution_time
```

Poniższe funkcje nie są opisane, jednak w swojej naturze są dosyć oczywiste więc może nie potrzeba.

```
def save_json(key,content):
    '''
    Zapisywanie wyników do jsona, żeby potem ich użyć w ipynb/prezentacji.
    '''
    with open("results.json","r") as f:
        loaded = json.load(f)

    loaded[key] = content

    with open("results.json", "w") as output:
        json.dump(loaded,output, indent=4)


def save_model(clf,dataset_name):
    model = clf.export_model()
    try:
        model.save("models/model_" + dataset_name , save_format="tf")
    except Exception:
        model.save("models/model_" + dataset_name + ".h5")

def get_model(dataset_name):
    return load_model("models/model_" + dataset_name, custom_objects=ak.CUSTOM_OBJECTS)
```

```
def main():
    dataset_names = np.array([       # nazwy wzięte ręcznie z dokumentu, bo nie działają l
        "adult", "airlines",
        # "albert",  #możliwe, że działa, mi zacina
        "amazon employee...", "apsfailure" ,"australian" ,
        "bank-marketing",
        "blood-transfusion" ,
        # "christine" , #wywala blad
        "credit-g" ,
        "guiellermo" ,
        "higgs" ,"jasmine" ,"kc1" ,"kddcup09 appetency" ,"kr-vs-kp" ,"miniboone" ,
        "nomao" ,"numerai28.6" ,"phoneme" ,
        # "riccardo" , #wywala blad
        "sylvine",
        "car" ,"cnae-9" ,"connect-4" ,"covertype",
        # "dilbert" ,"dionis" ,"fabert" , # wywalają błąd
        "fashion-mnist" ,
        # "helena" , # wywala kernela
        #"jannis" , # wywala kernala
        "jungle chess...","mfeat-factors" ,
        # "robert" , #wywala blad
        "segment" ,
        "shuttle" ,"vehicle" ,
        #"volkert" # wywala blad
        ])

    datas, unmatched = get_datasets(dataset_names)

    save_json("unmatched", list(unmatched.astype(str)))
```

```
    for data in datas:
        print(data.name + " in progress...")
        # ogólnie to y zawsze jest None, a targetowa zmienna wydaje się być zawsze na końc
        X, y, categorical_indicator, attribute_names = data.get_data()
        y = pd.DataFrame(X.iloc[:,X.shape[1]-1])
        X = X.iloc[:,0:(X.shape[1]-1)]
        try:
            clf, scores_dict, execution_time = run(X,y)
            save_model(clf = clf,dataset_name = data.name)
            save_json(data.name,{"scores" : scores_dict, "time" : execution_time})
        except:
            save_json(data.name,"failed")
        print(data.name + " done")
```

```
main()
```

```
JSONDecodeError: Expecting value: line 1 column 1 (char 0)
```

Hide error details | Search on Stack Overflow

```
---------------------------------------------------------------
JSONDecodeError                           Traceback (most re
<ipython-input-42-263240bbee7e> in <module>
----> 1 main()

<ipython-input-41-4d80a74b37ec> in main()
     27     datas, unmatched = get_datasets(dataset_names)
     28
---> 29     save_json("unmatched", list(unmatched.astype(str
     30
     31     for data in datas:

<ipython-input-40-4da6b4357ccd> in save_json(key, content)
      4     '''
      5     with open("results.json","r") as f:
----> 6         loaded = json.load(f)
      7
      8     loaded[key] = content

/usr/local/lib/python3.7/json/__init__.py in load(fp, cls, c
    294         cls=cls, object_hook=object_hook,
    295         parse_float=parse_float, parse_int=parse_int
--> 296         parse_constant=parse_constant, object_pairs_
    297
    298

/usr/local/lib/python3.7/json/__init__.py in loads(s, encodi
    346             parse_int is None and parse_float is Non
    347             parse_constant is None and object_pairs_
--> 348         return _default_decoder.decode(s)
    349     if cls is None:
    350         cls = JSONDecoder
```

Tutaj reprodukcja sie wywala, gdyż nie działa mi funkcja `save_json`.

Czy ten kod osiąga cel, który postawiono?

Sądząc po wynikach osiągniętych przez grupę Tojada, kod nie działa dla niektórych zbiorów.

Czy w kodzie są jakieś oczywiste błędy logiczne?

Nie zauważyłem żadnych jednak nie zdołałem też w pełni uruchomić kodu.

Czy kod jest zgodny z istniejącymi wytycznymi stylistycznymi? (czy kod jest zgodny z PEP 8)

Ogólnie tak, jednak miejscami można się przyczepić do formatowania kodu i może warto przed wysłaniem puścić przez jakiś autoformater kodu.

Czy są jakieś obszary, w których kod mógłby zostać poprawiony? (skrócić, przyspieszyć, itp.)

Nie znalazłem takich.

Czy dokumentacja i komentarze są wystarczające?

Dla mnie tak, jednak dla osoby niezwiązanej na codzień z AutoML może wymagać więcej szczegółów.

Czy udało się odtworzyć zamieszone przykłady w kodzie?

Nie.

Czy udało się użyć przygotowanych kodów na znowych danych?

Nie udało się w ogóle odtworzyć kodu.