AdvDSI-A2-beer-fast-api-catboost

March 20, 2022

1 AdvDSI - Assignment 2: Multi-Class Classification - Beer Style Predictor - Fastapi - Catboost

Train a machine *learning model* (using sklearn) or a *custom neural networks* (using pytorch) that will

accurately predict a type of beer based on some users' rating criterias such as appearance, aroma, palate or taste.

You will also need to build a web app and deploy it online (using Heroku) in order to serve your model for real time predictions.

Student Name: Nathan Fragar

Student No. : 93087548

Week: 6

Date: 20MAR2022

Description: This notebook is prepare functions and processing for FastAPI Docker Notebook

1.1 1. Load Dataset

[**1.1**] Task: Import required packages: Pandas, Numpy, joblib etc

[230]:
```python
# Task: Import the pandas, numpy and joblib package
import pandas as pd
import numpy as np
from joblib import load
import joblib as job

from starlette.responses import JSONResponse
from typing import Optional
from pydantic import BaseModel

# Algorithm - Catboost
import catboost
from catboost import *
from catboost import CatBoostClassifier
from catboost import Pool
```

[162]:
```
! python --version
```

```
Python 3.9.10
```

[64]:
```
# Change Working Directory: /home/jovyan/work
```

[2]:
```
cd /home/jovyan/work
```

```
/home/jovyan/work
```

[3]:
```
# Task: Launch the magic commands for auto-relaoding external modules
%load_ext autoreload
%autoreload 2
```

[**1.2**] Task: Load Dataset - Beer Styles and Breweries

[348]:
```
# file_url
file_path_beer_style = 'data/reference/beer_style.csv'
file_path_brewery = 'data/reference/breweries.csv'
file_path_beer_min_max = 'data/reference/beer_min_max_scale.csv'

# Load files into df_raw data frames
df_beer_style = pd.read_csv(file_path_beer_style)
df_brewery = pd.read_csv(file_path_brewery)
df_brewery_id = pd.read_csv(file_path_brewery, index_col='brewery_id')
df_beer_min_max_scale = pd.read_csv(file_path_beer_min_max, index_col='col')
```

[308]:
```
df_brewery_id.head()
```

[308]:
```
                       brewery_name
brewery_id
13160             't Hofbrouwerijke
17863         (512) Brewing Company
16873         10 Barrel Brewing Co.
4473            1516 Brewing Company
20688        16 Mile Brewing Company
```

[**1.3**] Display Datasets - Beer Styles and Breweries

[5]:
```
df_beer_style.head()
```

[5]:
```
                   beer_style  beer_style_cat
0                     Altbier               0
1       American Adjunct Lager              1
2     American Amber / Red Ale              2
3   American Amber / Red Lager              3
4          American Barleywine              4
```

[7]:
```
df_brewery.head()
```

[7]:
```
              brewery_name  brewery_id
0         't Hofbrouwerijke       13160
1    (512) Brewing Company       17863
2     10 Barrel Brewing Co.       16873
3      1516 Brewing Company        4473
4  16 Mile Brewing Company       20688
```

[8]:
```
df_beer_min_max_scale.head()
```

[8]:
```
                   min   max
col
review_aroma         1   5.0
review_appearance    0   5.0
review_palate        1   5.0
review_taste         1   5.0
beer_abv             0  57.7
```

[**1.4**] Create and Load Catboost Model

[6]:
```
# Load saved catboost model into cb_model
cb_model = load('models/catboost_model5.joblib')
```

[**1.5**] Check model operation

[7]:
```
#89
df_test_record_type_89 = [11031,0.176634,0.750,0.9,1.000,0.750]
cb_model.predict(df_test_record_type_89,prediction_type='Class')
```

[7]:
```
array([89])
```

[**1.6**] Create Min-Max Scaler to pre-process input

[564]:
```
def MinMaxScaleFrame(df, cols, inplace=True):

    for col in cols:
        x_min = df_beer_min_max_scale.loc[col,'min']
        x_max = df_beer_min_max_scale.loc[col,'max']
        col_std = col + '_std'
        col_scaled = col + '_scaled'

        if inplace==True:
            df[col] = (df[col] - x_min)/(x_max - x_min)
        else:
            df[col_std] = (df[col] - x_min)/(x_max - x_min)

    return df
```

Test Model Test the model with the following records

[9]:
```
# Brewery Name: Brouwerij De Molen (11031) - BeerStyle: Russian Imperial␣
 ↪Stout(89)
df_test_11031_89 = pd.DataFrame([[11031,0.176634,0.750,0.9,1.000,0.750]],␣
 ↪columns =␣
 ↪['brewery_id','beer_abv','review_palate','review_aroma','review_appearance','review_taste']
#df_test_15438_89 = MinMaxScaleFrame(df_test_15438_65, minmax_scale_cols,␣
 ↪inplace=True)
df_test_11031_89.head()
```

[9]:
```
   brewery_id  beer_abv  review_palate  review_aroma  review_appearance  \
0       11031  0.176634           0.75           0.9                1.0

   review_taste
0          0.75
```

[11]:
```
cb_model.predict(df_test_11031_89,prediction_type='Class')
```

[11]:
```
array([[89]])
```

[**1.6**] Check with API Input

[12]:
```
minmax_scale_cols =␣
 ↪['beer_abv','review_palate','review_aroma','review_appearance','review_taste']
```

[13]:
```
df_test_15438_65 = pd.DataFrame([[15438,5.0,1.5,2.0,2.5,1.5]], columns =␣
 ↪['brewery_id','beer_abv','review_palate','review_aroma','review_appearance','review_taste']
df_test_15438_65 = MinMaxScaleFrame(df_test_15438_65, minmax_scale_cols,␣
 ↪inplace=True)
df_test_15438_65.head()
```

[13]:
```
   brewery_id  beer_abv  review_palate  review_aroma  review_appearance  \
0       15438  0.086655          0.125          0.25                0.5

   review_taste
0         0.125
```

[14]:
```
cb_model.predict(df_test_15438_65,prediction_type='Class')
```

[14]:
```
array([[1]])
```

1.2 2. Build Calculate Beer Type/s

Catboost Model Properties There are 6 numerical columns, no categorical columns and 1 label encoded target column

- brewery_id - integer

- beer_abv - float
- review_aroma - float
- review_appearance - float
- review_palate - float
- review_taste - float
- beer_style_cat - label

[**Task 2.1**] '/beer/type/' (GET): Returning prediction for a single input only

[15]:
```
df_test_5438_65 = pd.DataFrame([[15438,5.0,1.5,2.0,2.5,1.5]], columns =␣
 ↪['brewery_id','beer_abv','review_palate','review_aroma','review_appearance','review_taste']
```

[16]:
```
# Beer (65): Hefeweizen
#brewery_name = None
brewery_name = 'Gulf Brewery'
brewery_id = 15438
#brewery_id = 15597
beer_abv = 5.0
review_palate = 1.5
review_aroma = 2.0
review_appearance = 2.5
review_taste = 1.5
```

[37]:
```

```

[262]:
```
# Caldera Brewing Company,1480,7.2,2.5,1.5,2.5,2.0,Oatmeal Stout : Beer Style:␣
 ↪Oatmeal Stout (83)
#brewery_name = None
brewery_name = 'Caldera Brewing Company'
brewery_id = 1480
beer_abv = 7.2
review_palate = 2.5
review_aroma = 1.5
review_appearance = 2.5
review_taste = 2.0
```

[ ]:
```

```

[18]:
```
def format_features(brewery_name: str, brewery_id: int,beer_abv: float,␣
 ↪review_palate: float, review_aroma: float, review_appearance: float,␣
 ↪review_taste: float):
    if brewery_name == None and  brewery_id != None:
        query_string = 'brewery_id==' + str(brewery_id)
        brewery_name = str(df_brewery.query(query_string)['brewery_name'].
 ↪values[0])

    if brewery_name != None and  brewery_id == None:
        query_string = 'brewery_name=="' + brewery_name + '"'
        brewery_id = df_brewery.query(query_string)['brewery_id'].values[0]

    return {
        'brewery_name': [brewery_name],
        'brewery_id': [brewery_id],
        'beer_abv': [beer_abv],
        'review_palate': [review_palate],
        'review_aroma': [review_aroma],
        'review_appearance': [review_appearance],
        'review_taste': [review_taste],
    }
```

[226]:
```
def predict_beer(brewery_name: str, brewery_id: int,beer_abv: float,␣
↪review_palate: float, review_aroma: float, review_appearance: float,␣
↪review_taste: float):
    # Config: Columns to Min Max Scale
    minmax_scale_cols =␣
↪['beer_abv','review_palate','review_aroma','review_appearance','review_taste']
    # Format input parameters as a JSON row
    features = format_features(brewery_name, brewery_id, beer_abv,␣
↪review_palate, review_aroma,review_appearance,review_taste  )
    # Convert features to Pandas DataFrame
    obs = pd.DataFrame(features)
    # Drop brewery_name column
    obs_predict = obs.drop(['brewery_name'], axis=1)
    # Perform MinMax Scaling
    obs_predict = MinMaxScaleFrame(obs_predict, minmax_scale_cols, inplace=True)
    # Perform Prediction
    pred = cb_model.predict(obs_predict)

    pred_beer_style_cat = pred[0].tolist()
    pred_beer_style_cat = pred_beer_style_cat[0]

    df_pred_beer_style  = df_beer_style
    df_pred_beer_style =␣
↪df_pred_beer_style[df_pred_beer_style['beer_style_cat'] ==␣
↪pred_beer_style_cat]

    beer_style = df_pred_beer_style['beer_style'].item()

    obs['beer_style_cat_predicted'] = pred_beer_style_cat
    obs['beer_style_predicted'] = beer_style

    obs_dict = obs.to_dict()

    # Return Prediction
    #return JSONResponse(pred.tolist())
    return obs_dict
```

[227]:
```
prediction = predict_beer(brewery_name=brewery_name, brewery_id=brewery_id,␣
↪beer_abv=beer_abv, review_palate=review_palate,␣
↪review_aroma=review_aroma,review_appearance=review_appearance,review_taste=review_taste)
```

[228]:
```
prediction
```

[228]:
```
{'brewery_name': {0: 'Caldera Brewing Company'},
 'brewery_id': {0: 1480},
 'beer_abv': {0: 7.2},
 'review_palate': {0: 2.5},
 'review_aroma': {0: 1.5},
 'review_appearance': {0: 2.5},
 'review_taste': {0: 2.0},
 'beer_style_cat_predicted': {0: 12},
 'beer_style_predicted': {0: 'American IPA'}}
```

[22]:
```
beer_style_cat = prediction.item(0)
```

[23]:
```
print(beer_style_cat)
```

```
12
```

[456]:
```
df_test_5438_65 = pd.DataFrame([['1516 Brewing Company',5.0,1.5,2.0,2.5,1.
↪5],['16 Mile Brewing Company',5.0,1.5,2.0,2.5,1.5]], columns =␣
↪['brewery_name','beer_abv','review_palate','review_aroma','review_appearance','review_taste
```

[442]:
```
df_test_5438_65['beer_style_cat_predicted'] = 16
```

[457]:
```
df_test_5438_65.head()
```

[457]:
```
              brewery_name  beer_abv  review_palate  review_aroma  \
0     1516 Brewing Company       5.0            1.5           2.0
1  16 Mile Brewing Company       5.0            1.5           2.0

   review_appearance  review_taste
0                2.5           1.5
1                2.5           1.5
```

[ ]:
```
df_test_5438_65.drop(['beer_style_predicted'],axis=1)
```

[445]:
```
df_test_5438_65['beer_style_predicted'] = df_beer_style['beer_style'].
↪loc[df_test_5438_65['beer_style_cat_predicted'].astype(int).values[0]]
```

[446]:
```
df_test_5438_65.head()
```

```
[446]:               brewery_name  beer_abv  review_palate  review_aroma  \
       0     1516 Brewing Company       5.0            1.5           2.0
       1  16 Mile Brewing Company       5.0            1.5           2.0

          review_appearance  review_taste  beer_style_cat_predicted  \
       0                2.5           1.5                        16
       1                2.5           1.5                        16

             beer_style_predicted
       0  American Pale Wheat Ale
       1  American Pale Wheat Ale
```

```
[447]: df_brewery.head()
```

```
[447]:               brewery_name  brewery_id
       0         't Hofbrouwerijke       13160
       1    (512) Brewing Company       17863
       2     10 Barrel Brewing Co.       16873
       3     1516 Brewing Company        4473
       4  16 Mile Brewing Company       20688
```

```
[473]: def find_brewery_id(df):
           for i, row in df.iterrows():
               brewery_name = row['brewery_name']
               brewery_name_query = 'brewery_name=="' + brewery_name + '"'
               brewery_id = df_brewery.query(brewery_name_query)['brewery_id'].
        ↪astype(int).values[0]
               brewery_id = int(brewery_id)
               df.at[i,'brewery_id'] = brewery_id
           # convert back to int
           df = df.astype({'brewery_id':'int'})
           return df
```

```
[471]: df_test_5438_65 = find_brewery_id(df_test_5438_65)
```

```
[475]: df_test_5438_65.head()
```

```
[475]:               brewery_name  beer_abv  review_palate  review_aroma  \
       0     1516 Brewing Company       5.0            1.5           2.0
       1  16 Mile Brewing Company       5.0            1.5           2.0

          review_appearance  review_taste  brewery_id
       0                2.5           1.5        4473
       1                2.5           1.5       20688
```

Check Dictionary Use for pydantic processing in FastAPI POST

- Not successful in processing pydantic input

[523]:
```
dict_list = [{
    "beer_abv": 1,
    "review_palate": 1,
    "review_aroma": 1,
    "review_appearance": 1,
    "review_taste": 1,
    "brewery_name": "16 Mile Brewing Company",
    "brewery_id": None
  }
    ,
  {
    "beer_abv": 2,
    "review_palate": 2,
    "review_aroma": 2,
    "review_appearance": 2,
    "review_taste": 2,
    "brewery_name": "1516 Brewing Company",
    "brewery_id": None
  }]
```

[524]:
```
dict_list
```

[524]:
```
[{'beer_abv': 1,
  'review_palate': 1,
  'review_aroma': 1,
  'review_appearance': 1,
  'review_taste': 1,
  'brewery_name': '16 Mile Brewing Company',
  'brewery_id': None},
 {'beer_abv': 2,
  'review_palate': 2,
  'review_aroma': 2,
  'review_appearance': 2,
  'review_taste': 2,
  'brewery_name': '1516 Brewing Company',
  'brewery_id': None}]
```

[507]:
```
def find_brewery_id(datalist):
    for dic_item in datalist:
        brewery_name = dic_item['brewery_name']
        brewery_name_query = 'brewery_name=="' + brewery_name + '"'
        print(brewery_name_query)
        dic_item['brewery_id'] = df_brewery.
 ↪query(brewery_name_query)['brewery_id'].astype(int).values[0]
    return datalist
```

[525]:
```
datalist = find_brewery_id(dict_list)
```

```
brewery_name=="16 Mile Brewing Company"
brewery_name=="1516 Brewing Company"
```

[527]:
```
dict_list
```

[527]:
```
[{'beer_abv': 1,
  'review_palate': 1,
  'review_aroma': 1,
  'review_appearance': 1,
  'review_taste': 1,
  'brewery_name': '16 Mile Brewing Company',
  'brewery_id': 20688},
 {'beer_abv': 2,
  'review_palate': 2,
  'review_aroma': 2,
  'review_appearance': 2,
  'review_taste': 2,
  'brewery_name': '1516 Brewing Company',
  'brewery_id': 4473}]
```

[529]:
```
## Pydantic Modellingclass beer_style(BaseModel):
```

[531]:
```
class beer_style(BaseModel):
    beer_abv: float
    review_palate: float
    review_aroma: float
    review_appearance: float
    review_taste: float
    brewery_name: str
    brewery_id: Optional[int]=None
```

[559]:
```
beer = "100.0,1.5,3.0,2.5,1.5,1516 Brewing Company"
```

[548]:
```
beerRow = beer.split(',')
```

[551]:
```
# beer_abv, review_palate, review_aroma, review_appearance,␣
 ↪review_tastebrewery_name, brewery_id [Optional]
beer_abv = float(beerRow[0])
review_palate = float(beerRow[1])
review_aroma = float(beerRow[2])
review_appearance = float(beerRow[3])
review_taste = float(beerRow[4])
brewery_name = str(beerRow[5])
```

[553]:
```
review_taste
```

[553]: 1.5

[555]:
```
def convert_beer_input(beerRow):
    beerRow = beer.split(',')
    beer_abv = float(beerRow[0])
    review_palate = float(beerRow[1])
    review_aroma = float(beerRow[2])
    review_appearance = float(beerRow[3])
    review_taste = float(beerRow[4])
    brewery_name = str(beerRow[5])
    return beer_abv, review_palate, review_aroma, review_appearance,␣
 ↪review_taste, brewery_name
```

[560]:
```
beer_abv, review_palate, review_aroma, review_appearance, review_taste,␣
 ↪brewery_name = convert_beer_input(beer)
```

[561]:
```
beer_abv
```

[561]: 100.0