Proj-Aplicado-XPE-MBA-Machine-Learning Documentation

Release 0.1

Diego Abreu

Jul 08, 2023

Sumário:

CAPÍTULO 1

teste

1.1 Importacao de dados

```
import pandas as pd
df = pd.read_csv('../data/external/casas.csv')
df.head()
```

	tamanho	ano	garagem	preco
0	159.0	2003	2	208500
1	117.0	1976	2	181500
2	166.0	2001	2	223500
3	160.0	1915	3	140000
4	204.0	2000	3	250000

```
X = df.drop('preco',axis=1)
y = df['preco'].copy()
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

```
import mlflow
mlflow.set_experiment('teste-mlflow')
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import math
```

1.2 Modelo 1

```
with mlflow.start_run(run_name='Regressao Linear'):
    lr = LinearRegression()
    lr.fit(X_train, y_train)
    mlflow.sklearn.log_model(lr,'lr')
    lr_predicted = lr.predict(X_test)
    mse = mean_squared_error(y_test, lr_predicted)
    rmse =  math.sqrt(mse)
    r2 = r2_score(y_test, lr_predicted)
    mlflow.log_metric('mse',mse)
    mlflow.log_metric('rmse',rmse)
    mlflow.log_metric('r2',r2)
```

1.3 Modelo 2

```
from xgboost import XGBRegressor
xgb_params = {
    'learning_rate':0.2,
    'n_estimators': 50,
    'random_state':42
}
with mlflow.start_run(run_name='Xgboost'):
    xgb = XGBRegressor(**xgb_params)
    xgb.fit(X_train, y_train)
    mlflow.xgboost.log_model(xgb,'xgboost')
    xgb_predicted = xgb.predict(X_test)
    mse = mean_squared_error(y_test, xgb_predicted)
    rmse =  math.sqrt(mse)
    r2 = r2_score(y_test, xgb_predicted)
    mlflow.log_metric('mse',mse)
    mlflow.log_metric('rmse',rmse)
    mlflow.log_metric('r2',r2)
```

```
/Users/diegoabreu/opt/anaconda3/lib/python3.8/site-packages/xgboost/data.py:262:␣
↪FutureWarning: pandas.Int64Index is deprecated and will be removed from pandas in a␣
↪future version. Use pandas.Index with the appropriate dtype instead.
  elif isinstance(data.columns, (pd.Int64Index, pd.RangeIndex)):
```

```
from glob import glob
import os
```

```
x = glob("../data/raw/dataset_faceforensics/fake_video/*mp4")
x[0]
```

```
'../data/raw/dataset_faceforensics/fake_video/000_003.mp4'
```

```
"../data/raw/dataset_faceforensics/fake_video/df_" + x[0].replace("../data/raw/dataset_
↪faceforensics/fake_video/","").replace(".mp4","")+ ".mp4"
```

```
'../data/raw/dataset_faceforensics/fake_video/df_000_003.mp4'
```

```
for i in glob("../data/raw/dataset_faceforensics/fake_video/*mp4"):
    os.rename(i, "../data/raw/dataset_faceforensics/fake_video/df_" + i.replace("../data/
↪raw/dataset_faceforensics/fake_video/","").replace(".mp4","")+ ".mp4")
```

```
for i in glob("../data/raw/dataset_faceforensics/fake_video/Face2Face/c23/videos/*mp4"):
    os.rename(i, "../data/raw/dataset_faceforensics/fake_video/Face2Face/c23/videos/f2f_
↪" + i.replace("../data/raw/dataset_faceforensics/fake_video/Face2Face/c23/videos/","").
↪replace(".mp4","")+ ".mp4")
```

```
for i in glob("../data/raw/dataset_faceforensics/fake_video/FaceSwap/c23/videos/*mp4"):
    os.rename(i, "../data/raw/dataset_faceforensics/fake_video/FaceSwap/c23/videos/fs_"␣
↪+ i.replace("../data/raw/dataset_faceforensics/fake_video/FaceSwap/c23/videos/","").
↪replace(".mp4","")+ ".mp4")
```

```
for i in glob("../data/raw/dataset_faceforensics/fake_video/NeuralTextures/c23/videos/
↪*mp4"):
    os.rename(i, "../data/raw/dataset_faceforensics/fake_video/NeuralTextures/c23/videos/
↪nt_" + i.replace("../data/raw/dataset_faceforensics/fake_video/NeuralTextures/c23/
↪videos/","").replace(".mp4","")+ ".mp4")
```

```
json_files[0]
```

```
'../data/raw/dataset_dfdc/a_identificar/dfdc_train_part_0/metadata.json'
```

```
pastas = glob("../data/raw/dataset_dfdc/a_identificar/*")
json_files=[]
for pasta in pastas:
    try:
        json_files.append(glob(i+"/*json")[0])
    except:
        print("erro: ", i)
json_files
```

```
import json
json_meta = json.load(open(json_files[0]))
#json_meta
```

```
import json
from glob import glob
import os
import shutil
pastas = glob("../data/raw/dataset_dfdc/a_identificar/*")
json_files=[]
for pasta in pastas:
    #print(glob(pasta+"/*json")[0])
    json_meta_file = glob(pasta+"/*json")[0]
    json_meta = json.load(open(json_meta_file))
    #print(json_meta)
```

(continues on next page)

(continued from previous page)

```
    videos = glob(pasta+"/*mp4")
    for video in videos:
        z = video.replace( pasta+"/", "")
        if json_meta[z]['label'] == 'FAKE':
            shutil.move(video,"../data/raw/dataset_dfdc/fake_video/")
        else:
            shutil.move(video,"../data/raw/dataset_dfdc/real_video/")
```

```
len(glob("../data/raw/dataset_dfdc/real_video/*"))
```

```
19154
```

```
for i in glob(pastas[0]+"/*mp4"):
    z = i.replace( "../data/raw/dataset_dfdc/a_identificar/dfdc_train_part_0/", "")
    if json_meta[z]['label'] == 'FAKE':
        print('fake')
    else:
        print('real')
```

```
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
real
fake
fake
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
real
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
real
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
real
fake
fake
fake
fake
fake
real
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
real
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
real
fake
real
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
real
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
real
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
fake
fake
fake
real
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
real
fake
real
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
real
real
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
real
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
fake
fake
real
real
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
real
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
real
fake
fake
real
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
real
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
real
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
fake
fake
fake
real
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
fake
```

(continues on next page)

(continued from previous page)

```
fake
fake
```

```
json_meta[z]['label']
```

```
'FAKE'
```

CAPÍTULO 2

Análise Exploratória

2.1 Datasets:

Os datasets serão armazenados no diretório data conforme a estrutura a seguir:

...

```
├── data                            <- Diretório de dados.
│   ├── external                    <- Diretório de dados externos(temporário).
│   │   └── input_video             <- Diretório para armazenar o vídeo recebido via
│   │                                  upload para classificação.
│   │
│   ├── interim                     <- Diretório de dados em processamento (temporário).
│   │   └── input_face              <- Diretório para armazenar os frames processados
│   │                                  do vídeo recebido via upload para classificação.
│   │
│   ├── processed                   <- Diretório de dados processados.
│   │   ├── dataset_celebdf         <- Diretório de faces do dataset celebdf
│   │   │   └── real_face           <- Diretório de faces reais
│   │   │   └── fake_face           <- Diretório de faces falsas
│   │   │
│   │   ├── dataset_faceforensics   <- Diretório de faces do dataset faceforensics
│   │   │   └── real_face           <- Diretório de faces reais
│   │   │   └── fake_face           <- Diretório de faces falsas
│   │   │
│   │   └── dataset_dfdc            <- Diretório de faces do dataset dfdc
│   │       └── real_face           <- Diretório de faces reais
│   │       └── fake_face           <- Diretório de faces falsas
│   │
│   └── raw                         <- Diretório de dados brutos.
│       ├── dataset_celebdf         <- Diretório de vídeos do dataset celebdf
│       │   └── real_video          <- Diretório de vídeos reais
```

(continues on next page)

(continued from previous page)

```
│       │   └── fake_video          <- Diretório de vídeos falsos
│       │
│       ├── dataset_faceforensics   <- Diretório de vídeos do dataset faceforensics
│       │   └── real_video          <- Diretório de vídeos reais
│       │   └── fake_video          <- Diretório de vídeos falsos
│       │
│       └── dataset_dfdc            <- Diretório de vídeos do dataset dfdc
│           └── real_video          <- Diretório de vídeos reais
│           └── fake_video          <- Diretório de vídeos falsos
```

...

```
# Importação de Bibliotecas:
from glob import glob
```

```
videos_reais = glob("../data/raw/dataset_celebdf/real_video/*.mp4")
```

```
from IPython.display import HTML
from base64 import b64encode
import os

tipo = 'real_video'
pasta = 'Proj-Aplicado-XPE-MBA-Machine-Learning/data/raw/dataset_celebdf'

def play_video(video_file, subset=tipo):
    '''
    Display video
    param: video_file - the name of the video file to display
    param: subset - the folder where the video file is located (can be TRAIN_SAMPLE_
↪FOLDER or TEST_Folder)
    '''
    video_url = open(os.path.join(pasta, subset,video_file),'rb').read()
    data_url = "data:video/mp4;base64," + b64encode(video_url).decode()
    return HTML("""<video width=500 controls><source src="%s" type="video/mp4"></video>""
↪" % data_url)
```

```
play_video(fake_videos[0])
```

2.1.1 CELEB-DF:

Link:

- Celeb-DF: https://www.kaggle.com/datasets/reubensuju/celeb-df-v2

```
# Lista de vídeos:
videos_reais = glob("../data/raw/dataset_celebdf/real_video/*.mp4")
videos_fakes= glob("../data/raw/dataset_celebdf/fake_video/*.mp4")
# Totais:
total_reais = len(videos_reais)
print("Total de vídeos reais: ", total_reais)
total_fakes = len(videos_fakes)
```

(continues on next page)

(continued from previous page)

```
print("Total de vídeos falsos: ", total_fakes)
print("Total de vídeos: ", total_reais + total_fakes)
```

```
Total de vídeos reais:  890
Total de vídeos falsos:  5639
Total de vídeos:  6529
```

```
# Amostra reais:
```

```
videos_reais[0]
```

```
'../data/raw/dataset_celebdf/real_video/00001.mp4'
```

```
from IPython.display import HTML
from base64 import b64encode
import os

video_url = open('../data/raw/dataset_dfdc/id0_id1_0000-hevcmp4.mp4','rb').read()
data_url = "data:video/mp4;base64," + b64encode(video_url).decode()
HTML("""<video width=500 controls><source src="%s" type="video/mp4"></video>""" % data_
↪url)
```

```
<IPython.core.display.HTML object>
```

```
from IPython.display import HTML
from base64 import b64encode
import os

tipo = 'real_video'
pasta = 'Proj-Aplicado-XPE-MBA-Machine-Learning/data/raw/dataset_celebdf'

def play_video(video_file, subset=tipo):
    '''
    Display video
    param: video_file - the name of the video file to display
    param: subset - the folder where the video file is located (can be TRAIN_SAMPLE_
↪FOLDER or TEST_Folder)
    '''
    video_url = open(os.path.join(pasta, subset,video_file),'rb').read()
    data_url = "data:video/mp4;base64," + b64encode(video_url).decode()
    return HTML("""<video width=500 controls><source src="%s" type="video/mp4"></video>""
↪" % data_url)
```

```
# Amostra falsos:
```

2.1.2 FaceForensics++:

Link:

- FaceForensics: https://www.kaggle.com/datasets/sorokin/faceforensics

```
total_real = glob("../data/raw/dataset_faceforensics/real_video/*.mp4")
print("Total de vídeos reais: ", len(total_real))
total_fake = glob("../data/raw/dataset_faceforensics/fake_video/*.mp4")
print("Total de vídeos falsos: ", len(total_fake))
print("Total de vídeos: ", len(total_real)+len(total_fake))
```

```
Total de vídeos reais:  1000
Total de vídeos falsos:  4000
Total de vídeos:  5000
```

2.1.3 DFDC:

Link:

- DFDC: https://www.kaggle.com/competitions/deepfake-detection-challenge/data

```
total_real = glob("../data/raw/dataset_dfdc/real_video/*.mp4")
print("Total de vídeos reais: ", len(total_real))
total_fake = glob("../data/raw/dataset_dfdc/fake_video/*.mp4")
print("Total de vídeos falsos: ", len(total_fake))
print("Total de vídeos: ", len(total_real)+len(total_fake))
```

```
Total de vídeos reais:  0
Total de vídeos falsos:  0
Total de vídeos:  0
```