

Wed May 25 05:29:34 2022

Processes:						
GPU	GI	CI	PID	Type	Process name	GPU Memory Usage
	ID	ID				
=====						
No running processes found						

```

Downloading torch_cubert-3.1.1-py3-none-any.whl (33 kB)
Collecting optuna
  Downloading optuna-2.10.0-py3-none-any.whl (308 kB)
    |████████████████████████████████████████| 308 kB 12.2 MB/s
Requirement already satisfied: tqdm<5.0,>=4.36 in /usr/local/lib/python3.7/dist-packages (from optuna)
Requirement already satisfied: scikit_learn>0.21 in /usr/local/lib/python3.7/dist-packages (from optuna)
Requirement already satisfied: scipy>1.4 in /usr/local/lib/python3.7/dist-packages (from optuna)
Requirement already satisfied: numpy<2.0,>=1.17 in /usr/local/lib/python3.7/dist-packages (from optuna)
Requirement already satisfied: torch<2.0,>=1.2 in /usr/local/lib/python3.7/dist-packages (from optuna)
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.7/dist-packages (from optuna)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from optuna)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dist-packages (from optuna)
Collecting alembic
  Downloading alembic-1.7.7-py3-none-any.whl (210 kB)
    |████████████████████████████████████████| 210 kB 67.2 MB/s
Requirement already satisfied: PyYAML in /usr/local/lib/python3.7/dist-packages (from alembic)
Collecting cliff
  Downloading cliff-3.10.1-py3-none-any.whl (81 kB)
    |████████████████████████████████████████| 81 kB 12.5 MB/s
Collecting colorlog
  Downloading colorlog-6.6.0-py2.py3-none-any.whl (11 kB)
Collecting cmaes>=0.8.2
  Downloading cmaes-0.8.2-py3-none-any.whl (15 kB)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.7/dist-packages (from cmaes)
Requirement already satisfied: sqlalchemy>=1.1.0 in /usr/local/lib/python3.7/dist-packages (from cmaes)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python3.7/dist-packages (from cmaes)
Requirement already satisfied: greenlet!=0.4.17 in /usr/local/lib/python3.7/dist-packages (from cmaes)
Requirement already satisfied: importlib-metadata in /usr/local/lib/python3.7/dist-packages (from cmaes)
Collecting Mako
  Downloading Mako-1.2.0-py3-none-any.whl (78 kB)
    |████████████████████████████████████████| 78 kB 9.1 MB/s
Requirement already satisfied: importlib-resources in /usr/local/lib/python3.7/dist-packages (from Mako)
Collecting pbr!=2.1.0,>=2.0.0

```

```

Downloading pbr-5.9.0-py2.py3-none-any.whl (112 kB)
|████████████████████████████████████████| 112 kB 75.7 MB/s
Collecting stevedore>=2.0.1
  Downloading stevedore-3.5.0-py3-none-any.whl (49 kB)
|████████████████████████████████████████| 49 kB 7.4 MB/s
Collecting cmd2>=1.0.0
  Downloading cmd2-2.4.1-py3-none-any.whl (146 kB)
|████████████████████████████████████████| 146 kB 55.0 MB/s
Collecting autopage>=0.4.0
  Downloading autopage-0.5.0-py3-none-any.whl (29 kB)
Requirement already satisfied: PrettyTable>=0.7.2 in /usr/local/lib/python3.7/dist-packages
Collecting pyperclip>=1.6
  Downloading pyperclip-1.8.2.tar.gz (20 kB)
Requirement already satisfied: attrs>=16.3.0 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: wcwidth>=0.1.7 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: MarkupSafe>=0.9.2 in /usr/local/lib/python3.7/dist-packages
Building wheels for collected packages: pyperclip
  Building wheel for pyperclip (setup.py) ... done
  Created wheel for pyperclip: filename=pyperclip-1.8.2-py3-none-any.whl size=1113
  Stored in directory: /root/.cache/pip/wheels/9f/18/84/8f69f8b08169c7bae2dde6bd7d
Successfully built pyperclip
Installing collected packages: pyperclip, pbr, stevedore, Mako, cmd2, autopage, co
Successfully installed Mako-1.2.0 alembic-1.7.7 autopage-0.5.0 cliff-3.10.1 cmaes-

```

```

import pandas as pd
import numpy as np
from pytorch_tabnet.tab_model import TabNetClassifier
from sklearn.metrics import accuracy_score, classification_report
import optuna as opt
import torch
import os
import joblib

def make_save_cv_model(i,model_name,model,best_params,optim,output_path="./drive/MyDrive/S

''' This function saves cross validation model in the corresponding directory ( if the

if os.path.exists(os.path.join(output_path,f"{i}_{model_name}_{optim}")):
    joblib.dump(model, os.path.join(output_path,f"{i}_{model_name}_{optim}/{i}_model.z
    with open(os.path.join(output_path,f"{i}_{model_name}_{optim}/model_params.txt"),'
        file.write(str(best_params))
else:
    os.mkdir(os.path.join(output_path,f"{i}_{model_name}_{optim}"))
    joblib.dump(model, os.path.join(output_path,f"{i}_{model_name}_{optim}/{i}_model.z
    with open(os.path.join(output_path,f"{i}_{model_name}_{optim}/model_params.txt"),'
        file.write(str(best_params))

def train(fold_dict,fold,model_name,sc_df,tar_col,optim,optim_trial,k_folds=10,tar_cols=""

''' this function is used to train the model with parameters optimization using optuna

y = sc_df[tar_col]

```

```

x = sc_df.drop([tar_col],axis=1)
model_name = model_name
def objective(trial):
    train_index = fold_dict[fold]["train"]
    test_index = fold_dict[fold]["test"]
    clf = TabNetClassifier(n_d=trial.suggest_int("n_d", 8, 64),
                           n_a=trial.suggest_int("n_a", 8, 64),
                           n_steps = trial.suggest_int("n_steps",3,10),
                           gamma =trial.suggest_float("gamma", 1.0, 2.0),
                           n_independent = trial.suggest_int("n_independent",1,5),
                           n_shared = trial.suggest_int("n_shared",1,5),
                           momentum = trial.suggest_float("momentum", 0.01, 0.4),
                           optimizer_fn = torch.optim.Adam,
                           # scheduler_fn = torch.optim.lr_scheduler,
                           # scheduler_params = {"gamma" :trial.suggest_float("sch-gamm
                           verbose = verbose,
                           device_name = "auto"
                           )

    # print(f" train_index :: {train_index}")
    # print(f" test_index :: {test_index}")
    X_train,X_test = x.iloc[train_index,:], x.iloc[test_index,:]
    # print(X_train.shape, X_test.shape)
    X_train, X_test = X_train.to_numpy(dtype=np.float64), X_test.to_numpy(dtype=np.float
    Y_train, Y_test = y.iloc[train_index], y.iloc[test_index]
    Y_train, Y_test = Y_train.to_numpy(dtype=np.float64), Y_test.to_numpy(dtype=np.float
    print(Y_train.shape, Y_test.shape)
    clf.fit(X_train, Y_train,
            eval_set=[(X_test, Y_test)],
            eval_metric=['accuracy'])
    Y_pred = clf.predict(X_test)
    print(classification_report(Y_test, Y_pred, labels=[x for x in range(6)]))
    acc = accuracy_score(Y_pred, Y_test)
    return acc

print(f"Starting optimization for fold : [{fold}/{k_folds}]")
study = opt.create_study(direction='maximize')
study.optimize(objective, n_trials=optim_trial)
best_params = study.best_params
print(f" Best params for fold: [{fold}/{k_folds}]")
print(best_params)
joblib.dump(best_params,f"./drive/MyDrive/SOLAR_CELL/ML_PROCESSED_DATA/outputs/{model_
with open(f"./drive/MyDrive/SOLAR_CELL/ML_PROCESSED_DATA/outputs/{model_name}/best_par
print(f"Saved best_params at : outputs/{model_name}/best_params/fold_{fold}_best_param
train_index = fold_dict[fold]["train"]
test_index = fold_dict[fold]["test"]
X_train,X_test = x.iloc[train_index,:], x.iloc[test_index,:]
# print(X_train.shape, X_test.shape)
X_train, X_test = X_train.to_numpy(dtype=np.float64), X_test.to_numpy(dtype=np.float64
Y_train, Y_test = y.iloc[train_index], y.iloc[test_index]
Y_train, Y_test = Y_train.to_numpy(dtype=np.float64), Y_test.to_numpy(dtype=np.float64
clf_model = TabNetClassifier(**study.best_params)
clf_model.fit(X_train,Y_train)
Y_pred = clf_model.predict(X_test)
clf_report = classification_report(Y_test, Y_pred, labels=[x for x in range(6)])
with open(f"./drive/MyDrive/SOLAR_CELL/ML_PROCESSED_DATA/outputs/classification_report

```

```

accuracy = accuracy_score(Y_pred, Y_test)
with open(f"./drive/MyDrive/SOLAR_CELL/ML_PROCESSED_DATA/outputs/{model_name}/{model_n
try:
    print("[++] Saving the model and parameters in corresponding directories")
    make_save_cv_model(fold,model_name,clf_model,best_params,optim=optim)
except:
    print("[-] Failed to save the model")

use_df = pd.read_csv("./drive/MyDrive/SOLAR_CELL/ML_PROCESSED_DATA/outputs/data/trainable_
tar_col = "PCE_categorical"
model_name = "pytorch_tabnet"
optimizer = "Adam"
fold_dict = joblib.load("./drive/MyDrive/SOLAR_CELL/ML_PROCESSED_DATA/inputs/fold_vals/fol
fold = 7

train(fold_dict = fold_dict,
      fold = fold,
      model_name=model_name,
      sc_df=use_df,
      tar_col=tar_col,
      optim=optimizer,
      optim_trial = 15)
print(f"[++] Ended the training process for fold {fold}")

epoch 45 | loss: 0.4851 | 0:01:05s
epoch 46 | loss: 0.48369 | 0:01:05s
epoch 47 | loss: 0.47319 | 0:01:06s
epoch 48 | loss: 0.47693 | 0:01:08s
epoch 49 | loss: 0.47355 | 0:01:09s
epoch 50 | loss: 0.46744 | 0:01:10s
epoch 51 | loss: 0.46172 | 0:01:12s
epoch 52 | loss: 0.4594 | 0:01:13s
epoch 53 | loss: 0.45782 | 0:01:15s
epoch 54 | loss: 0.45932 | 0:01:16s
epoch 55 | loss: 0.45315 | 0:01:17s
epoch 56 | loss: 0.45807 | 0:01:19s
epoch 57 | loss: 0.45774 | 0:01:20s
epoch 58 | loss: 0.45175 | 0:01:22s
epoch 59 | loss: 0.45385 | 0:01:23s
epoch 60 | loss: 0.45236 | 0:01:24s
epoch 61 | loss: 0.45587 | 0:01:26s
epoch 62 | loss: 0.45409 | 0:01:27s
epoch 63 | loss: 0.45178 | 0:01:29s
epoch 64 | loss: 0.44594 | 0:01:30s
epoch 65 | loss: 0.44727 | 0:01:31s
epoch 66 | loss: 0.43736 | 0:01:33s
epoch 67 | loss: 0.43979 | 0:01:34s
epoch 68 | loss: 0.4311 | 0:01:35s
epoch 69 | loss: 0.43137 | 0:01:37s
epoch 70 | loss: 0.43087 | 0:01:38s
epoch 71 | loss: 0.43689 | 0:01:40s
epoch 72 | loss: 0.43686 | 0:01:41s
epoch 73 | loss: 0.43248 | 0:01:42s
epoch 74 | loss: 0.42672 | 0:01:44s
epoch 75 | loss: 0.42122 | 0:01:45s
epoch 76 | loss: 0.42182 | 0:01:47s
epoch 77 | loss: 0.41965 | 0:01:48s

```

```

epoch 78 | loss: 0.42008 | 0:01:49s
epoch 79 | loss: 0.42371 | 0:01:51s
epoch 80 | loss: 0.42514 | 0:01:52s
epoch 81 | loss: 0.42842 | 0:01:53s
epoch 82 | loss: 0.42069 | 0:01:55s
epoch 83 | loss: 0.41499 | 0:01:56s
epoch 84 | loss: 0.41409 | 0:01:58s
epoch 85 | loss: 0.41446 | 0:01:59s
epoch 86 | loss: 0.41736 | 0:02:00s
epoch 87 | loss: 0.41127 | 0:02:02s
epoch 88 | loss: 0.41259 | 0:02:03s
epoch 89 | loss: 0.40798 | 0:02:05s
epoch 90 | loss: 0.41016 | 0:02:06s
epoch 91 | loss: 0.40146 | 0:02:07s
epoch 92 | loss: 0.40841 | 0:02:09s
epoch 93 | loss: 0.40159 | 0:02:10s
epoch 94 | loss: 0.39688 | 0:02:11s
epoch 95 | loss: 0.41724 | 0:02:13s
epoch 96 | loss: 0.40704 | 0:02:14s
epoch 97 | loss: 0.40777 | 0:02:16s
epoch 98 | loss: 0.40029 | 0:02:17s
epoch 99 | loss: 0.40125 | 0:02:18s
[++] Saving the model and parameters in corresponding directories
[++] Ended the training process for fold 7

```

Fold 0 has started running on 20-05-22

Fold 0 has completed sucessfully on 17:00 20-05-22

Fold 1 has started running at 15:15 21-05-22

Fold 2 has started running at 09:45 22-05-22

Fold 2 has completed sucessfully on 10:58 22-05-22

Fold 3 has started running at 18:40 22-05-22

Fold 3 has completed sucessfully on 22-05-22

Fold 4 completed sucessfully on 21:04 on 22-05-22

Fold 5 started at 18:21 on 23-05-22

Fold 5 completed sucessfully on 19:44 on 23-05-22

Fold 6 started at 12:53 on 24-05-22

Fold 6 has completed at 14:14 on 24-05-22

Fold 7 started at 14:18 on 24-05-22

Fold 7 execution failed due to colab gpu time limit

Fold 7 trial 1 started at 11:00 on 25-05-22

Fold 7 has completed sucessfully at 12:14 on 25-05-22

✓

1h 12m 43s

completed at 12:12 PM

×