# -\*- coding: utf-8 -\*-

"""

Created on Sun Aug 26 06:19:16 2018

@author: madhuakshi

"""

import lightgbm as lgb

import pandas as pd

#from sklearn.metrics import mean\_squared\_error

# load or create your dataset

df\_train = pd.read\_csv("train\_final.csv")

df\_test = pd.read\_csv("test\_final.csv")

features=df\_train.iloc[:,:-1].values

labels=df\_train.ROLE

from sklearn.preprocessing import LabelEncoder,OneHotEncoder

encode=LabelEncoder()

features[:,24]=encode.fit\_transform(features[:,24])

labels[:]=encode.fit\_transform(labels[:])

hotencode=OneHotEncoder(categorical\_features=[24])

features=hotencode.fit\_transform(features).toarray()

from sklearn.cross\_validation import train\_test\_split

x\_train,x\_test,y\_train,y\_test=train\_test\_split(features,labels,test\_size=0.3)

# create dataset for lightgbm

lgb\_train = lgb.Dataset(x\_train, y\_train)

lgb\_eval = lgb.Dataset(x\_test, y\_test, reference=lgb\_train)

# specify your configurations as a dict

params = {

'boosting\_type': 'gbdt',

'objective': 'multiclass',

'num\_class':df\_train.ROLE.nunique(),

'metric': 'multi\_logloss',

'learning\_rate': 0.05,

'max\_depth': 7,

'num\_leaves': 31,

'feature\_fraction': 0.9,

'bagging\_fraction': 0.8,

'bagging\_freq': 5}

print('Start training...')

# train

gbm = lgb.train(params,

lgb\_train,

num\_boost\_round=10,

valid\_sets=lgb\_train,

categorical\_feature=[df\_train])

print('Save model...')

# save model to file

gbm.save\_model('model.txt')

print('Start predicting...')

# predict

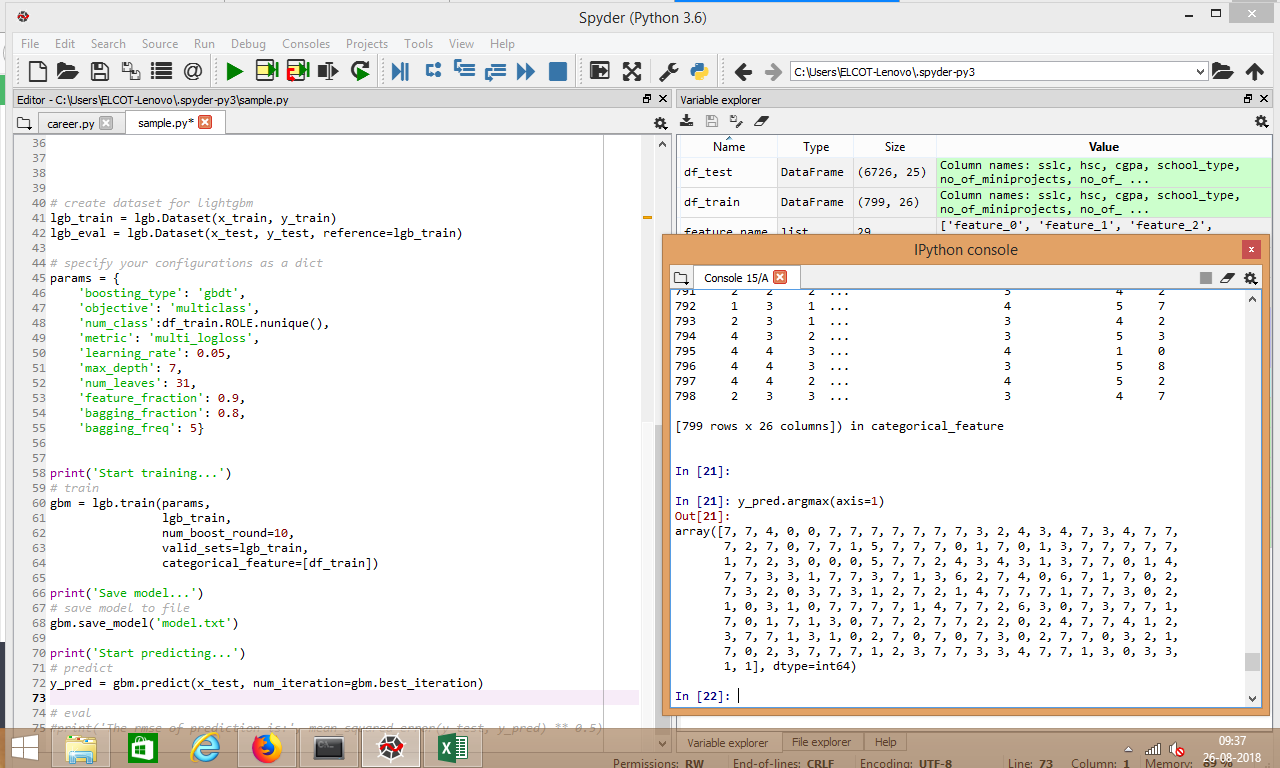
y\_pred = gbm.predict(x\_test, num\_iteration=gbm.best\_iteration)

print((y\_pred == y\_test).sum()/len(y\_pred))

# eval

#print('The rmse of prediction is:', mean\_squared\_error(y\_test, y\_pred) \*\* 0.5)

**OUTPUT:**

****