

**DoS attack detection model of smart grid based on machine learning method**

**نام استاد: دکتر یزدی**

**نام دانشجو : حمزه قائدی**

**شماره دانشجویی: 9831419**

**زمستان 99**

**کدهای شبیه سازی مقاله**

*Wang Zhe, Cheng Wei\*, Li Chunlin*

**1/5**

افزودن کتابخانه ها به پروژه

import numpy as np

import pandas as pd

from matplotlib import pyplot as plt

import sklearn

from sklearn import svm

from sklearn.preprocessing import StandardScaler

from sklearn.metrics import f1\_score,recall\_score,precision\_score,confusion\_matrix,accuracy\_score

**کتابخانه های استفاده شده**



**پروژه در پلتفرم گوگل کولب و با زبان پایتون پیاده سازی شده است**



**Pandas: کار با فایل دیتاست**

**NumPy: محاسبات ماتریسی**

**Sklearn: کتابخانه ای برای یادگیری ماشین**

**Pyplot: رسم نمودار**



#-------------------------------------------------------------------------------

#-----------------------------GENERALS------------------------------------------

#-------------------------------------------------------------------------------

#--------------------------all features-----------------------------------------

features = ['duration', 'protocol\_type', 'service', 'flag', 'src\_bytes', 'dst\_bytes', 'land', 'wrong\_fragment',

            'urgent', 'hot', 'num\_failed\_logins', 'logged\_in', 'num\_compromised', 'root\_shell', 'su\_attempted',

            'num\_root', 'num\_file\_creations', 'num\_shells', 'num\_access\_files', 'num\_outbound\_cmds', 'is\_host\_login',

            'is\_guest\_login', 'count', 'srv\_count', 'serror\_rate', 'srv\_serror\_rate', 'rerror\_rate', 'srv\_rerror\_rate’,  'same\_srv\_rate', 'diff\_srv\_rate', 'srv\_diff\_host\_rate', 'dst\_host\_count', 'dst\_host\_srv\_count',

'dst\_host\_same\_srv\_rate', 'dst\_host\_diff\_srv\_rate', 'dst\_host\_same\_src\_port\_rate', 'dst\_host\_srv\_diff\_host\_rate',

'dst\_host\_serror\_rate','dst\_host\_srv\_serror\_rate', 'dst\_host\_rerror\_rate', 'dst\_host\_srv\_rerror\_rate','label']

#-------------------------------------------------------------------------------

#-------------selected featues--------------------------------------------------

#discrete features

discrete\_features = ['protocol\_type','service','flag','land','logged\_in',

                     'root\_shell','su\_attempted','is\_host\_login','is\_guest\_login']

#continuous features:

continuous\_features = ['duration','src\_bytes','dst\_bytes','wrong\_fragment',

                       'urgent','hot','num\_failed\_logins','num\_root',

                       'num\_access\_files','num\_outbound\_cmds',

         'count','srv\_count','serror\_rate','srv\_serror\_rate','rerror\_rate',

         'srv\_rerror\_rate','num\_file\_creations','num\_shells','same\_srv\_rate',

         'diff\_srv\_rate']

#-------------------------------------------------------------------------------

#-------------DoS attack types--------------------------------------------------

dos\_attacks = ['back.','land.','neptune.','pod.','smurf.','teardrop.']

#-------------------------------------------------------------------------------

اسامی تمام ویژگیها، ویژگیهای انخاب شده توسط مقاله و نیز اسامی اقسام حملات DoS موجود در دیتاست در قالب لیست هایی ذخیره شده است

**2/5**

**تعریف توابع کمکی**

#----------------------------------normalize------------------------------------

#purpose: z-scores features

#inputs:

#       dataset: numpy-like array of samples (continuous only)

#       feature\_list : list of continuous features to be normalized

#outputs: -

def normalize(dataset,feature\_list):

  scaler = StandardScaler() #from sklearn

  for i in feature\_list:

    dataset[i] = scaler.fit\_transform(dataset[i].values.reshape(-1,1))

#-------------------------------------------------------------------------------

تابع زیر، ویژگیهای پیوسته را استاندارد میکند

#---------------------------------adjust labels---------------------------------

#purpose: adjusting labels

#inputs:

#       dataset : numpy array of samples

#outputs:

#       returns proper labels for samples

def adjust\_labels(dataset):

  y = []

  for d in dataset['label'].values:

    if(d in dos\_attacks):

      y.append(1) #if DoS

    else:

      y.append(-1) #if Not DoS

  return y

#-------------------------------------------------------------------------------

برچسب خروجی نمونه ها، اسامی حملات مرسوم میباشد، تابع زیر به نمونه های مربوط به حمله DoS خروجی **یک** و سایر نمونه ها خروجی **منفی** **یک** نظیر میکند

**3/5**

#--------------------------------preprocess-------------------------------------

#purpose: preprocessing raw data

#inputs:

#       dataset: raw dataset

#output:

#       returns normalized, reduced version of the dataset

def preprocess(dataset):

  disc\_feats = dataset[discrete\_features] #extract selected discrete features

  cont\_feats = dataset[continuous\_features] #extract selected continuous features

  #sparse categorical encoding for discrete string-vaulued features

  disc\_feats['protocol\_type'] = disc\_feats['protocol\_type'].astype('category').cat.codes

  disc\_feats['service'] = disc\_feats['service'].astype('category').cat.codes

  disc\_feats['flag'] = disc\_feats['flag'].astype('category').cat.codes

  #normalizing continuous features

  normalize(cont\_feats,continuous\_features)

  #putting preprocessed discrete and continuous features together

  final\_dataset = np.hstack([disc\_feats,cont\_feats])

  return final\_dataset

#-------------------------------------------------------------------------------

تابع زیر، پیش پردازش های مطرح شده در مقاله را بر دیتاست اعمال میکند (ویژگیهای گسسته رشته ای را به معادل عددی تبدیل کرده و به کمک تابع قبلی، ویژگیهای پیوسته را استاندارد میکند و در انتها دیتاست آماده شده را بازمیگرداند

**4/5**

**پیاده سازی**

#----------------------------------------------------------------

#-----------------------Load Dataset-----------------------------

#----------------------------------------------------------------

train\_dataset = pd.read\_csv('http://kdd.ics.uci.edu/databases/kddcup99/kddcup.data\_10\_percent.gz',compression='gzip',header = 0,names = features)

test\_dataset = pd.read\_csv('http://kdd.ics.uci.edu/databases/kddcup99/corrected.gz',compression='gzip',header = 0,names = features)

#-----------------------------------------------------------------

#drop duplicates------------------------------

train\_dataset.drop\_duplicates(subset=features,keep='first',inplace=True)

test\_dataset.drop\_duplicates(subset=features,keep='first',inplace=True)

#---------------------------------------------

#----------adjust labels----------------------

y\_train = adjust\_labels(train\_dataset)

y\_test = adjust\_labels(test\_dataset)

#----------preprocess-------------------------

x\_train = preprocess(train\_dataset)

x\_test  = preprocess(test\_dataset)

#---------------------------------------------

اسکریپت زیِر، ابتدا دیتاستهای مربوط به آموزش و تست را دانلود کرده سپس به کمک توابع کمکی تعریف شده داده ها را آماده میکند

#-----------fitting an SVM Classifier---------

clf = svm.SVC(C = 0.8,gamma = 0.1)

clf.fit(x\_train,y\_train)

#---------------------------------------------

#---------Testing-----------------------------

p = clf.predict(x\_test)

#---------------------------------------------

در نهایت، اسکریپت زیر، به کمک کتابخانه sklearn یک کلاسیفایر SVM به داده ها برازش میکند

**بررسی نتایج**

print("Classification Results for SVM:",'-'\*10)

print("Accuracy: ",accuracy\_score(p,y\_test))

print("Precision: ",precision\_score(y\_test,p))

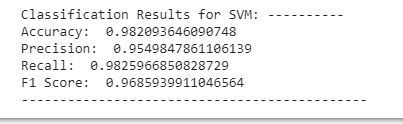
print("Recall: ",recall\_score(y\_test,p))

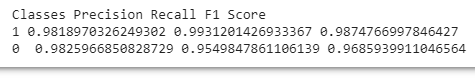
print("F1 Score: ",f1\_score(y\_test,p))

print('-'\*45)

بررسی نتایج (به کمک توابع موجود در کتابخانه sklearn)

**5/5**





**مشاهده میشود که نتایج حاصله، تفاوت اندکی با نتایج مقاله دارد**

conf\_mat = confusion\_matrix(y\_test,p)

TP = conf\_mat[0][0]

FP = conf\_mat[0][1]

FN = conf\_mat[1][0]

TN = conf\_mat[1][1]

prec\_0 = TP/(TP + FP)

rec\_0 = TP / (TP + FN)

f1\_0 = 2\* (prec\_0 \* rec\_0)/(prec\_0 + rec\_0)

prec\_1 = TN / (TN + FN)

rec\_1 = TN / (TN + FP)

f1\_1 = 2 \* (prec\_1 \* rec\_1)/(prec\_1  +  rec\_1)

print("Classes","Precision","Recall","F1 Score")

print("1",prec\_0,rec\_0,f1\_0)

print("0 ",prec\_1,rec\_1,f1\_1)

بررسی نتایج (به کمک توابع موجود در کتابخانه sklearn)