import pandas as pd

from sklearn.preprocessing import LabelEncoder

from sklearn.preprocessing import Normalizer

names = ["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"]

data = pd.read\_csv("kddcup.data\_10\_percent.csv", names=names)

pd.set\_option('display.width', 2000)

pd.set\_option('display.max\_column', 50)

pd.set\_option('precision', 3)

data1 = data

data1 = data1.drop('protocol\_type', axis=1)

data1 = data1.drop('service', axis=1)

data1 = data1.drop('flag', axis=1)

data1 = data1.drop('label', axis=1)

import matplotlib.pyplot as plt

cor = data1.corr()

fig = plt.figure()

subFig = fig.add\_subplot(111)

cax = subFig.matshow(cor, vmin=-1, vmax=1)

fig.colorbar(cax)

import numpy as np

ticks = np.arange(0, len(data1.columns))

subFig.set\_xticks(ticks)

subFig.set\_yticks(ticks)

subFig.set\_xticklabels(data1.columns, rotation='vertical', size=7)

subFig.set\_yticklabels(data1.columns, size=7)

plt.show()

x = list(cor.columns)

g = []

for i in x:

for j in x[:x.index(i)]:

if abs(cor[i][j]) >= 0.7:

g.append((i, j))

o, c = [], 0

for i in g:

for v in o:

if i[0] in [x for x in v] or i[1] in [x for x in v]:

c += 1

if c == 0:

s = {i[0], i[1]}

for j in g[g.index(i) + 1:]:

if i[0] == j[0] or i[0] == j[1] or i[1] == j[0] or i[1] == j[1]:

s.add(j[0])

s.add(j[1])

o.append(s)

c = 0

b = []

listd = set(b)

for i in o:

for j in list(i)[:-1]:

listd.add(j)

for i in listd:

data = data.drop(i, axis=1)

dataX = data.values[:, :data.shape[1] - 1]

dataY = data.values[:, data.shape[1] - 1]

dataY1 = dataY

dataX[:, 1] = LabelEncoder().fit\_transform(dataX[:, 1])

dataX[:, 2] = LabelEncoder().fit\_transform(dataX[:, 2])

dataX[:, 3] = LabelEncoder().fit\_transform(dataX[:, 3])

dataY = LabelEncoder().fit\_transform(dataY)

label = {}

for i in range(len(dataY1)):

label[dataY[i]] = dataY1[i]

def acuu(x, y):

l = []

from sklearn.metrics import accuracy\_score

from sklearn.model\_selection import train\_test\_split

x\_train, x\_val, y\_train, y\_val = train\_test\_split(x, y, test\_size=0.33, random\_state=7)

fit = Normalizer().fit(x\_train)

x\_train = fit.fit\_transform(x\_train)

from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier()

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

y\_pred = knn.predict(x\_val)

acc\_knn = round(accuracy\_score(y\_pred, y\_val) \* 100, 2)

print("MODEL-8: Accuracy of k-Nearest Neighbors : ", acc\_knn)

l.append(acc\_knn)

from sklearn.linear\_model import LogisticRegression

from sklearn.feature\_selection import RFE

model = LogisticRegression()

rfe = RFE(model, 17)

fit = rfe.fit(dataX, dataY)

train2 = fit.transform(dataX)

print("RFE")

acuu(train2, dataY)