cs-nslkdd-preproc

May 20, 2023

[32]:
```
import pandas as pd
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import train_test_split
```

[33]:
```
# Load the training dataset
train_data = pd.read_csv('KDDTrain+.txt')

# Load the testing dataset
test_data = pd.read_csv('KDDTest+.txt')
```

[44]:
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
from joblib import dump, load
from sklearn.metrics import accuracy_score, f1_score,␣
↪precision_score,recall_score
from sklearn.linear_model import Perceptron
from sklearn.linear_model import LogisticRegression
from sklearn.neural_network import MLPClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, VotingClassifier
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import AdaBoostClassifier
import matplotlib.pyplot as plt

from sklearn import svm, datasets
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.utils.multiclass import unique_labels
from sklearn.metrics import roc_curve, auc
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import label_binarize
from sklearn.multiclass import OneVsRestClassifier
from scipy import interp
from itertools import cycle
import seaborn as sns
from sklearn.datasets import make_classification
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import roc_curve
from sklearn.metrics import roc_auc_score
import sklearn.metrics as metrics

# Input data files are available in the "../input/" directory.
# For example, running this (by clicking run or pressing Shift+Enter) will list␣
↪all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

[45]:
```
import pandas as pd
import numpy as np
import sys
import keras
import sklearn
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Embedding
from keras.layers import LSTM, SimpleRNN, GRU, Bidirectional,␣
↪BatchNormalization,Convolution1D,MaxPooling1D, Reshape,␣
↪GlobalAveragePooling1D
from keras.utils import to_categorical
import sklearn.preprocessing
from sklearn import metrics
from scipy.stats import zscore
from tensorflow.keras.utils import get_file, plot_model
from sklearn.model_selection import train_test_split
from tensorflow.keras.callbacks import EarlyStopping
import matplotlib.pyplot as plt
print(pd.__version__)
print(np.__version__)
print(sys.version)
print(sklearn.__version__)
```

```
1.4.4
1.23.5
3.9.13 (main, Aug 25 2022, 23:51:50) [MSC v.1916 64 bit (AMD64)]
1.0.2
```

[46]:
```
#Loading training set into dataframe
df = pd.read_csv('KDDTrain+.txt', header=None)
df.head()
```

[46]:
```
   0    1         2   3    4     5  6  7  8  9  ...    33    34    35  \
0  0  tcp  ftp_data  SF  491     0  0  0  0  0  ...  0.17  0.03  0.17
1  0  udp     other  SF  146     0  0  0  0  0  ...  0.00  0.60  0.88
2  0  tcp   private  S0    0     0  0  0  0  0  ...  0.10  0.05  0.00
3  0  tcp      http  SF  232  8153  0  0  0  0  ...  1.00  0.00  0.03
4  0  tcp      http  SF  199   420  0  0  0  0  ...  1.00  0.00  0.00

     36    37    38    39    40       41  42
0  0.00  0.00  0.00  0.05  0.00   normal  20
1  0.00  0.00  0.00  0.00  0.00   normal  15
2  0.00  1.00  1.00  0.00  0.00  neptune  19
3  0.04  0.03  0.01  0.00  0.01   normal  21
4  0.00  0.00  0.00  0.00  0.00   normal  21

[5 rows x 43 columns]
```

[48]:
```python
#Loading testing set into dataframe
qp = pd.read_csv('KDDTest+.txt', header=None)
qp.head()
```

[48]:
```
   0     1         2     3      4   5  6  7  8  9  ...    33    34    35  \
0  0   tcp   private   REJ      0   0  0  0  0  0  ...  0.04  0.06  0.00
1  0   tcp   private   REJ      0   0  0  0  0  0  ...  0.00  0.06  0.00
2  2   tcp  ftp_data    SF  12983   0  0  0  0  0  ...  0.61  0.04  0.61
3  0  icmp     eco_i    SF     20   0  0  0  0  0  ...  1.00  0.00  1.00
4  1   tcp    telnet  RSTO      0  15  0  0  0  0  ...  0.31  0.17  0.03

     36   37   38    39    40       41  42
0  0.00  0.0  0.0  1.00  1.00  neptune  21
1  0.00  0.0  0.0  1.00  1.00  neptune  21
2  0.02  0.0  0.0  0.00  0.00   normal  21
3  0.28  0.0  0.0  0.00  0.00    saint  15
4  0.02  0.0  0.0  0.83  0.71    mscan  11

[5 rows x 43 columns]
```

[49]:
```python
#Reset column names for training set
df.columns = ['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', 'subclass', 'difficulty_level']
df.head()
```

[49]:
```
   duration protocol_type   service flag  src_bytes  dst_bytes  land  \
0         0           tcp  ftp_data   SF        491          0     0
1         0           udp     other   SF        146          0     0
2         0           tcp   private   S0          0          0     0
3         0           tcp      http   SF        232       8153     0
4         0           tcp      http   SF        199        420     0

   wrong_fragment  urgent  hot  …  dst_host_same_srv_rate  \
0               0       0    0  …                    0.17
1               0       0    0  …                    0.00
2               0       0    0  …                    0.10
3               0       0    0  …                    1.00
4               0       0    0  …                    1.00

   dst_host_diff_srv_rate  dst_host_same_src_port_rate  \
0                    0.03                         0.17
1                    0.60                         0.88
2                    0.05                         0.00
3                    0.00                         0.03
4                    0.00                         0.00

   dst_host_srv_diff_host_rate  dst_host_serror_rate  \
0                         0.00                  0.00
1                         0.00                  0.00
2                         0.00                  1.00
3                         0.04                  0.03
4                         0.00                  0.00

   dst_host_srv_serror_rate  dst_host_rerror_rate  dst_host_srv_rerror_rate  \
0                      0.00                  0.05                      0.00
1                      0.00                  0.00                      0.00
2                      1.00                  0.00                      0.00
3                      0.01                  0.00                      0.01
4                      0.00                  0.00                      0.00

   subclass  difficulty_level
0    normal                20
1    normal                15
2   neptune                19
3    normal                21
4    normal                21
```

```
[5 rows x 43 columns]
```

[50]:
```
#Reset column names for testing set
qp.columns = ['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', 'subclass', 'difficulty_level']
qp.head()
```

[50]:
```
   duration protocol_type   service  flag  src_bytes  dst_bytes  land  \
0         0           tcp   private   REJ          0          0     0
1         0           tcp   private   REJ          0          0     0
2         2           tcp  ftp_data    SF      12983          0     0
3         0          icmp     eco_i    SF         20          0     0
4         1           tcp    telnet  RSTO          0         15     0

   wrong_fragment  urgent  hot  …  dst_host_same_srv_rate  \
0               0       0    0  …                    0.04
1               0       0    0  …                    0.00
2               0       0    0  …                    0.61
3               0       0    0  …                    1.00
4               0       0    0  …                    0.31

   dst_host_diff_srv_rate  dst_host_same_src_port_rate  \
0                    0.06                         0.00
1                    0.06                         0.00
2                    0.04                         0.61
3                    0.00                         1.00
4                    0.17                         0.03

   dst_host_srv_diff_host_rate  dst_host_serror_rate  \
0                         0.00                   0.0
1                         0.00                   0.0
2                         0.02                   0.0
3                         0.28                   0.0
4                         0.02                   0.0

   dst_host_srv_serror_rate  dst_host_rerror_rate  dst_host_srv_rerror_rate  \
0                        0.0                    1.00                       1.00
1                        0.0                    1.00                       1.00
2                        0.0                    0.00                       0.00
3                        0.0                    0.00                       0.00
4                        0.0                    0.83                       0.71

   subclass  difficulty_level
0   neptune                21
1   neptune                21
2    normal                21
3     saint                15
4     mscan                11

[5 rows x 43 columns]
```

[51]:
```
#accessing names of training columns
lst_names = df.columns # returns a list of column names
lst_names
```

[51]:
```
Index(['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', 'subclass', 'difficulty_level'],
      dtype='object')
```

[52]:
```
#accessing names of testing columns
testlst_names = qp.columns
testlst_names
```

[52]:
```
Index(['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', 'subclass', 'difficulty_level'],
      dtype='object')
```

[53]:
```
#Dropping the last columns of training set
df = df.drop('difficulty_level', 1) # we don't need it in this project
df.shape
```

```
C:\Users\pappu\AppData\Local\Temp\ipykernel_20220\129585363.py:2: FutureWarning:
In a future version of pandas all arguments of DataFrame.drop except for the
argument 'labels' will be keyword-only.
  df = df.drop('difficulty_level', 1) # we don't need it in this project
```

[53]:
```
(125973, 42)
```

[54]:
```
#Dropping the last columns of testing set
qp = qp.drop('difficulty_level', 1)
qp.shape
```

```
C:\Users\pappu\AppData\Local\Temp\ipykernel_20220\700357697.py:2: FutureWarning:
In a future version of pandas all arguments of DataFrame.drop except for the
argument 'labels' will be keyword-only.
  qp = qp.drop('difficulty_level', 1)
```

[54]:
```
(22544, 42)
```

[55]:
```
df.isnull().values.any()
```

[55]:
```
False
```

[56]:
```
qp.isnull().values.any()
```

[56]:
```
False
```

[57]:
```
#defining col list
cols = ['protocol_type','service','flag']
cols
```

[57]:
```
['protocol_type', 'service', 'flag']
```

[58]:
```
#One-hot encoding
def one_hot(df, cols):
    """
    @param df pandas DataFrame
    @param cols a list of columns to encode
    @return a DataFrame with one-hot encoding
    """
    for each in cols:
        dummies = pd.get_dummies(df[each], prefix=each, drop_first=False)
        df = pd.concat([df, dummies], axis=1)
        df = df.drop(each, 1)
    return df
```

[59]:
```
#Merging train and test data
combined_data = pd.concat([df,qp])
```

[60]:
```
#Applying one hot encoding to combined data
combined_data = one_hot(combined_data,cols)
```

```
C:\Users\pappu\AppData\Local\Temp\ipykernel_20220\445847675.py:11:
FutureWarning: In a future version of pandas all arguments of DataFrame.drop
except for the argument 'labels' will be keyword-only.
  df = df.drop(each, 1)
C:\Users\pappu\AppData\Local\Temp\ipykernel_20220\445847675.py:11:
FutureWarning: In a future version of pandas all arguments of DataFrame.drop
except for the argument 'labels' will be keyword-only.
  df = df.drop(each, 1)
C:\Users\pappu\AppData\Local\Temp\ipykernel_20220\445847675.py:11:
FutureWarning: In a future version of pandas all arguments of DataFrame.drop
except for the argument 'labels' will be keyword-only.
  df = df.drop(each, 1)
```

[61]:
```
#Function to min-max normalize
def normalize(df, cols):
    """
    @param df pandas DataFrame
    @param cols a list of columns to encode
    @return a DataFrame with normalized specified features
    """
    result = df.copy() # do not touch the original df
    for feature_name in cols:
        max_value = df[feature_name].max()
        min_value = df[feature_name].min()
        if max_value > min_value:
            result[feature_name] = (df[feature_name] - min_value) / (max_value␣
 ↪- min_value)
    return result
```

[62]:
```
#Dropping subclass column for training set
tmp = combined_data.pop('subclass')
```

[63]:
```
tmp
```

```
[63]: 0          normal
      1          normal
      2         neptune
      3          normal
      4          normal
                 …
      22539      normal
      22540      normal
      22541        back
      22542      normal
      22543       mscan
      Name: subclass, Length: 148517, dtype: object
```

```
[64]: #Normalizing training set
      new_train_df = normalize(combined_data,combined_data.columns)
      new_train_df
```

```
[64]:        duration     src_bytes     dst_bytes  land  wrong_fragment  urgent  \
      0           0.0  3.558064e-07  0.000000e+00   0.0             0.0     0.0
      1           0.0  1.057999e-07  0.000000e+00   0.0             0.0     0.0
      2           0.0  0.000000e+00  0.000000e+00   0.0             0.0     0.0
      3           0.0  1.681203e-07  6.223962e-06   0.0             0.0     0.0
      4           0.0  1.442067e-07  3.206260e-07   0.0             0.0     0.0
      …            …            …            …   …              …     …
      22539       0.0  5.753774e-07  2.542106e-07   0.0             0.0     0.0
      22540       0.0  2.297162e-07  7.160648e-07   0.0             0.0     0.0
      22541       0.0  3.952277e-05  6.346868e-06   0.0             0.0     0.0
      22542       0.0  3.043558e-08  3.206260e-08   0.0             0.0     0.0
      22543       0.0  0.000000e+00  0.000000e+00   0.0             0.0     0.0

                  hot  num_failed_logins  logged_in  num_compromised  …  flag_REJ  \
      0      0.000000                0.0        0.0         0.000000  …       0.0
      1      0.000000                0.0        0.0         0.000000  …       0.0
      2      0.000000                0.0        0.0         0.000000  …       0.0
      3      0.000000                0.0        1.0         0.000000  …       0.0
      4      0.000000                0.0        1.0         0.000000  …       0.0
      …           …                …        …              …  …       …
      22539  0.000000                0.0        1.0         0.000000  …       0.0
      22540  0.000000                0.0        1.0         0.000000  …       0.0
      22541  0.019802                0.0        1.0         0.000134  …       0.0
      22542  0.000000                0.0        0.0         0.000000  …       0.0
      22543  0.000000                0.0        0.0         0.000000  …       1.0

             flag_RSTO  flag_RSTOS0  flag_RSTR  flag_S0  flag_S1  flag_S2  flag_S3  \
      0            0.0          0.0        0.0      0.0      0.0      0.0      0.0
      1            0.0          0.0        0.0      0.0      0.0      0.0      0.0
      2            0.0          0.0        0.0      1.0      0.0      0.0      0.0
```

```
3            0.0          0.0          0.0        0.0        0.0        0.0        0.0
4            0.0          0.0          0.0        0.0        0.0        0.0        0.0
…           …          …          …        …        …        …        …
22539        0.0          0.0          0.0        0.0        0.0        0.0        0.0
22540        0.0          0.0          0.0        0.0        0.0        0.0        0.0
22541        0.0          0.0          0.0        0.0        0.0        0.0        0.0
22542        0.0          0.0          0.0        0.0        0.0        0.0        0.0
22543        0.0          0.0          0.0        0.0        0.0        0.0        0.0

       flag_SF  flag_SH
0          1.0      0.0
1          1.0      0.0
2          0.0      0.0
3          1.0      0.0
4          1.0      0.0
…         …       …
22539      1.0      0.0
22540      1.0      0.0
22541      1.0      0.0
22542      1.0      0.0
22543      0.0      0.0

[148517 rows x 122 columns]
```

[65]:
```
#Fixing labels for training set
classlist = []
check1 =␣
 ↪("apache2","back","land","neptune","mailbomb","pod","processtable","smurf","teardrop","udps
check2 = ("ipsweep","mscan","nmap","portsweep","saint","satan")
check3 =␣
 ↪("buffer_overflow","loadmodule","perl","ps","rootkit","sqlattack","xterm")
check4 =␣
 ↪("ftp_write","guess_passwd","httptunnel","imap","multihop","named","phf","sendmail","Snmpge

DoSCount=0
ProbeCount=0
U2RCount=0
R2LCount=0
NormalCount=0

for item in tmp:
    if item in check1:
        classlist.append("DOS")
        DoSCount=DoSCount+1
    elif item in check2:
        classlist.append("Probe")
        ProbeCount=ProbeCount+1
    elif item in check3:
        classlist.append("U2R")
        U2RCount=U2RCount+1
    elif item in check4:
        classlist.append("R2L")
        R2LCount=R2LCount+1
    else:
        classlist.append("Normal")
        NormalCount=NormalCount+1
```

[66]:
```
classlist
```

[66]:
```
['Normal',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'DOS',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'R2L',
 'DOS',
 'DOS',
 'Normal',
 'Probe',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Probe',
 'DOS',
 'Normal',
 'Probe',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'R2L',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Probe',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'Probe',
 'Normal',
 'DOS',
 'Normal',
 'Probe',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Probe',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Probe',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Probe',
 'Normal',
 'R2L',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'Probe',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Probe',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'Probe',
 'Probe',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'R2L',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Probe',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'Probe',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'R2L',
 'DOS',
 'Probe',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Probe',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Probe',
 'DOS',
 'Normal',
 'Probe',
 'Normal',
 'Normal',
 'Probe',
 'Probe',
 'DOS',
 'Normal',
 'Normal',
 'Probe',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Probe',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Probe',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Probe',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'R2L',
 'Normal',
 'Probe',
 'Normal',
 'Normal',
 'Probe',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Probe',
 'Probe',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Probe',
 'Probe',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Probe',
 'Probe',
 'Normal',
 'DOS',
 'Normal',
 'Probe',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Probe',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'DOS',
 'Probe',
 'DOS',
 'Probe',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Probe',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Probe',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Probe',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'Probe',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'Probe',
 'Normal',
 'Normal',
 'Probe',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'R2L',
 'Probe',
 'Normal',
 'Probe',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Probe',
 'DOS',
 'Probe',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'R2L',
 'Probe',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Probe',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Probe',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'R2L',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Probe',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Probe',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Probe',
 'Normal',
 'Probe',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Probe',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Probe',
 'Probe',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Probe',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Probe',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'Probe',
 'Normal',
 'Probe',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Probe',
 'Probe',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'R2L',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Probe',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Probe',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'R2L',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'Probe',
 'DOS',
 'DOS',
 'Probe',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'R2L',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Probe',
 'R2L',
 'Probe',
 'Normal',
 'DOS',
 'Probe',
 'Normal',
 'Normal',
 'Normal',
 'Probe',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Probe',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Probe',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Probe',
 'Normal',
 'Normal',
 'Probe',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Probe',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Probe',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Probe',
 'DOS',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'Probe',
 'Normal',
 'Normal',
 'Probe',
 'Normal',
 'Normal',
 'Normal',
 'Probe',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Probe',
 'Probe',
 'DOS',
 'Probe',
 'Normal',
 'Probe',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Probe',
 'DOS',
 'DOS',
 'DOS',
 'DOS',
 'Probe',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Probe',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Probe',
 'Normal',
 'DOS',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'Normal',
 'DOS',
 'Normal',
 'DOS',
 'Normal',
 'Probe',
 'DOS',
 'Normal',
 'DOS',
 'Probe',
 'DOS',
 'DOS',
 'Normal',
 'Normal',
 'DOS',
 'DOS',
 'Normal',
 …]
```

[67]:
```
#Appending class column to training set
new_train_df["Class"] = classlist
new_train_df
```

```
C:\Users\pappu\AppData\Local\Temp\ipykernel_20220\803719109.py:2:
PerformanceWarning: DataFrame is highly fragmented.  This is usually the result
of calling `frame.insert` many times, which has poor performance.  Consider
joining all columns at once using pd.concat(axis=1) instead. To get a de-
fragmented frame, use `newframe = frame.copy()`
  new_train_df["Class"] = classlist
```

[67]:
```
       duration     src_bytes     dst_bytes  land  wrong_fragment  urgent  \
0           0.0  3.558064e-07  0.000000e+00   0.0             0.0     0.0
1           0.0  1.057999e-07  0.000000e+00   0.0             0.0     0.0
2           0.0  0.000000e+00  0.000000e+00   0.0             0.0     0.0
3           0.0  1.681203e-07  6.223962e-06   0.0             0.0     0.0
4           0.0  1.442067e-07  3.206260e-07   0.0             0.0     0.0
…           …           …           …   …             …     …
22539       0.0  5.753774e-07  2.542106e-07   0.0             0.0     0.0
22540       0.0  2.297162e-07  7.160648e-07   0.0             0.0     0.0
```

```
22541       0.0  3.952277e-05  6.346868e-06   0.0            0.0     0.0
22542       0.0  3.043558e-08  3.206260e-08   0.0            0.0     0.0
22543       0.0  0.000000e+00  0.000000e+00   0.0            0.0     0.0

            hot  num_failed_logins  logged_in  num_compromised  …  \
0      0.000000                0.0        0.0         0.000000  …
1      0.000000                0.0        0.0         0.000000  …
2      0.000000                0.0        0.0         0.000000  …
3      0.000000                0.0        1.0         0.000000  …
4      0.000000                0.0        1.0         0.000000  …
…           …                …        …              …  …
22539  0.000000                0.0        1.0         0.000000  …
22540  0.000000                0.0        1.0         0.000000  …
22541  0.019802                0.0        1.0         0.000134  …
22542  0.000000                0.0        0.0         0.000000  …
22543  0.000000                0.0        0.0         0.000000  …

       flag_RSTO  flag_RSTOS0  flag_RSTR  flag_S0  flag_S1  flag_S2  flag_S3  \
0            0.0          0.0        0.0      0.0      0.0      0.0      0.0
1            0.0          0.0        0.0      0.0      0.0      0.0      0.0
2            0.0          0.0        0.0      1.0      0.0      0.0      0.0
3            0.0          0.0        0.0      0.0      0.0      0.0      0.0
4            0.0          0.0        0.0      0.0      0.0      0.0      0.0
…            …          …        …      …      …      …      …
22539        0.0          0.0        0.0      0.0      0.0      0.0      0.0
22540        0.0          0.0        0.0      0.0      0.0      0.0      0.0
22541        0.0          0.0        0.0      0.0      0.0      0.0      0.0
22542        0.0          0.0        0.0      0.0      0.0      0.0      0.0
22543        0.0          0.0        0.0      0.0      0.0      0.0      0.0

       flag_SF  flag_SH   Class
0          1.0      0.0  Normal
1          1.0      0.0  Normal
2          0.0      0.0     DOS
3          1.0      0.0  Normal
4          1.0      0.0  Normal
…          …      …     …
22539      1.0      0.0  Normal
22540      1.0      0.0  Normal
22541      1.0      0.0     DOS
22542      1.0      0.0  Normal
22543      0.0      0.0   Probe

[148517 rows x 123 columns]
```

```
[68]: new_train_df["Class"].value_counts()
```

```
[68]: Normal    77232
      DOS       53387
      Probe     14077
      R2L        3702
      U2R         119
      Name: Class, dtype: int64
```

```
[69]: new_train_df.isnull().values.any()
```

```
[69]: False
```

```
[70]: y_train=new_train_df["Class"]
      y_train
```

```
[70]: 0        Normal
      1        Normal
      2           DOS
      3        Normal
      4        Normal
                …
      22539    Normal
      22540    Normal
      22541       DOS
      22542    Normal
      22543     Probe
      Name: Class, Length: 148517, dtype: object
```

```
[71]: y_train.isnull().values.any()
```

```
[71]: False
```

```
[72]: combined_data_X = new_train_df.drop('Class', 1)
      combined_data_X
```

```
C:\Users\pappu\AppData\Local\Temp\ipykernel_20220\3277739742.py:1:
FutureWarning: In a future version of pandas all arguments of DataFrame.drop
except for the argument 'labels' will be keyword-only.
  combined_data_X = new_train_df.drop('Class', 1)
```

```
[72]:        duration     src_bytes     dst_bytes  land  wrong_fragment  urgent  \
      0           0.0  3.558064e-07  0.000000e+00   0.0             0.0     0.0
      1           0.0  1.057999e-07  0.000000e+00   0.0             0.0     0.0
      2           0.0  0.000000e+00  0.000000e+00   0.0             0.0     0.0
      3           0.0  1.681203e-07  6.223962e-06   0.0             0.0     0.0
      4           0.0  1.442067e-07  3.206260e-07   0.0             0.0     0.0
      …           …           …           …    …              …       …
      22539       0.0  5.753774e-07  2.542106e-07   0.0             0.0     0.0
```

```
22540       0.0  2.297162e-07  7.160648e-07   0.0             0.0     0.0
22541       0.0  3.952277e-05  6.346868e-06   0.0             0.0     0.0
22542       0.0  3.043558e-08  3.206260e-08   0.0             0.0     0.0
22543       0.0  0.000000e+00  0.000000e+00   0.0             0.0     0.0

            hot  num_failed_logins  logged_in  num_compromised  …  flag_REJ  \
0      0.000000                0.0        0.0         0.000000  …       0.0
1      0.000000                0.0        0.0         0.000000  …       0.0
2      0.000000                0.0        0.0         0.000000  …       0.0
3      0.000000                0.0        1.0         0.000000  …       0.0
4      0.000000                0.0        1.0         0.000000  …       0.0
…           …                …        …                …  …       …
22539  0.000000                0.0        1.0         0.000000  …       0.0
22540  0.000000                0.0        1.0         0.000000  …       0.0
22541  0.019802                0.0        1.0         0.000134  …       0.0
22542  0.000000                0.0        0.0         0.000000  …       0.0
22543  0.000000                0.0        0.0         0.000000  …       1.0

       flag_RSTO  flag_RSTOS0  flag_RSTR  flag_S0  flag_S1  flag_S2  flag_S3  \
0            0.0          0.0        0.0      0.0      0.0      0.0      0.0
1            0.0          0.0        0.0      0.0      0.0      0.0      0.0
2            0.0          0.0        0.0      1.0      0.0      0.0      0.0
3            0.0          0.0        0.0      0.0      0.0      0.0      0.0
4            0.0          0.0        0.0      0.0      0.0      0.0      0.0
…          …          …        …      …      …      …      …
22539        0.0          0.0        0.0      0.0      0.0      0.0      0.0
22540        0.0          0.0        0.0      0.0      0.0      0.0      0.0
22541        0.0          0.0        0.0      0.0      0.0      0.0      0.0
22542        0.0          0.0        0.0      0.0      0.0      0.0      0.0
22543        0.0          0.0        0.0      0.0      0.0      0.0      0.0

       flag_SF  flag_SH
0          1.0      0.0
1          1.0      0.0
2          0.0      0.0
3          1.0      0.0
4          1.0      0.0
…        …      …
22539      1.0      0.0
22540      1.0      0.0
22541      1.0      0.0
22542      1.0      0.0
22543      0.0      0.0

[148517 rows x 122 columns]
```

[113]:
```
from collections import Counter

# Count the occurrences of each value
value_counts = Counter(classlist)

# Print unique values and their counts
for value, count in value_counts.items():
    print(f'{value}: {count}')
```

```
Normal: 77232
DOS: 53387
R2L: 3702
Probe: 14077
U2R: 119
```

[73]:
```
oos_pred = []
```

[74]:
```
from sklearn.model_selection import StratifiedKFold
```

[75]:
```
kfold = StratifiedKFold(n_splits=10,shuffle=True,random_state=42)
kfold.get_n_splits(combined_data_X,y_train)
```

[75]:
```
10
```

[80]:
```
batch_size = 32
model = Sequential()
model.add(Convolution1D(64, kernel_size=122, padding="same", activation="relu",␣
  ↪input_shape=(122, 1)))
model.add(MaxPooling1D(pool_size=5))
model.add(BatchNormalization())
model.add(Bidirectional(LSTM(64, return_sequences=False)))
model.add(Reshape((128, 1), input_shape=(128,)))

model.add(MaxPooling1D(pool_size=5))
model.add(BatchNormalization())
model.add(Bidirectional(LSTM(128, return_sequences=False)))

model.add(Dropout(0.5))
model.add(Dense(5))
model.add(Activation('softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam',␣
  ↪metrics=['accuracy'])
```

[81]:
```
for layer in model.layers:
    print(layer.output_shape)
```

```
(None, 122, 64)
```

```
(None, 24, 64)
(None, 24, 64)
(None, 128)
(None, 128, 1)
(None, 25, 1)
(None, 25, 1)
(None, 256)
(None, 256)
(None, 5)
(None, 5)
```

```
[82]: model.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d_1 (Conv1D)           (None, 122, 64)           7872

 max_pooling1d (MaxPooling1D  (None, 24, 64)           0
 )

 batch_normalization (BatchN  (None, 24, 64)           256
 ormalization)

 bidirectional (Bidirectiona  (None, 128)              66048
 l)

 reshape (Reshape)           (None, 128, 1)            0

 max_pooling1d_1 (MaxPooling  (None, 25, 1)            0
 1D)

 batch_normalization_1 (Batc  (None, 25, 1)            4
 hNormalization)

 bidirectional_1 (Bidirectio  (None, 256)              133120
 nal)

 dropout (Dropout)           (None, 256)               0

 dense (Dense)               (None, 5)                 1285

 activation (Activation)     (None, 5)                 0

=================================================================
Total params: 208,585
Trainable params: 208,455
```

```
Non-trainable params: 130
_________________________________________________________________
```

[83]:
```
for train_index, test_index in kfold.split(combined_data_X,y_train):
    train_X, test_X = combined_data_X.iloc[train_index], combined_data_X.
 ↪iloc[test_index]
    train_y, test_y = y_train.iloc[train_index], y_train.iloc[test_index]

    print("train index:",train_index)
    print("test index:",test_index)

    x_columns_train = new_train_df.columns.drop('Class')
    x_train_array = train_X[x_columns_train].values
    x_train_1=np.reshape(x_train_array, (x_train_array.shape[0], x_train_array.
 ↪shape[1], 1))

    dummies = pd.get_dummies(train_y) # Classification
    outcomes = dummies.columns
    num_classes = len(outcomes)
    y_train_1 = dummies.values

    x_columns_test = new_train_df.columns.drop('Class')
    x_test_array = test_X[x_columns_test].values
    x_test_2=np.reshape(x_test_array, (x_test_array.shape[0], x_test_array.
 ↪shape[1], 1))

    dummies_test = pd.get_dummies(test_y) # Classification
    outcomes_test = dummies_test.columns
    num_classes = len(outcomes_test)
    y_test_2 = dummies_test.values

    model.fit(x_train_1, y_train_1,validation_data=(x_test_2,y_test_2),␣
 ↪epochs=10)

    pred = model.predict(x_test_2)
    pred = np.argmax(pred,axis=1)
    y_eval = np.argmax(y_test_2,axis=1)
    score = metrics.accuracy_score(y_eval, pred)
    oos_pred.append(score)
    print("Validation score: {}".format(score))
```

```
train index: [     0      2      3 … 148514 148515 148516]
test index: [     1      7     18 … 148506 148511 148513]
Epoch 1/10
4178/4178 [==============================] - 139s 32ms/step - loss: 0.1068 -
accuracy: 0.9663 - val_loss: 0.0592 - val_accuracy: 0.9775
Epoch 2/10
```

```
4178/4178 [==============================] - 129s 31ms/step - loss: 0.0621 -
accuracy: 0.9787 - val_loss: 0.0545 - val_accuracy: 0.9807
Epoch 3/10
4178/4178 [==============================] - 140s 34ms/step - loss: 0.0513 -
accuracy: 0.9820 - val_loss: 0.0412 - val_accuracy: 0.9846
Epoch 4/10
4178/4178 [==============================] - 143s 34ms/step - loss: 0.0433 -
accuracy: 0.9847 - val_loss: 0.0405 - val_accuracy: 0.9852
Epoch 5/10
4178/4178 [==============================] - 133s 32ms/step - loss: 0.0379 -
accuracy: 0.9868 - val_loss: 0.0394 - val_accuracy: 0.9854
Epoch 6/10
4178/4178 [==============================] - 133s 32ms/step - loss: 0.0354 -
accuracy: 0.9876 - val_loss: 0.0346 - val_accuracy: 0.9875
Epoch 7/10
4178/4178 [==============================] - 147s 35ms/step - loss: 0.0324 -
accuracy: 0.9885 - val_loss: 0.0324 - val_accuracy: 0.9893
Epoch 8/10
4178/4178 [==============================] - 139s 33ms/step - loss: 0.0309 -
accuracy: 0.9890 - val_loss: 0.0347 - val_accuracy: 0.9883
Epoch 9/10
4178/4178 [==============================] - 135s 32ms/step - loss: 0.0300 -
accuracy: 0.9893 - val_loss: 0.0305 - val_accuracy: 0.9892
Epoch 10/10
4178/4178 [==============================] - 139s 33ms/step - loss: 0.0279 -
accuracy: 0.9898 - val_loss: 0.0294 - val_accuracy: 0.9886
465/465 [==============================] - 7s 11ms/step
Validation score: 0.9885537301373553
train index: [     0      1      2 … 148514 148515 148516]
test index: [    12     15     17 … 148456 148488 148504]
Epoch 1/10
4178/4178 [==============================] - 135s 32ms/step - loss: 0.0281 -
accuracy: 0.9899 - val_loss: 0.0261 - val_accuracy: 0.9908
Epoch 2/10
4178/4178 [==============================] - 134s 32ms/step - loss: 0.0268 -
accuracy: 0.9903 - val_loss: 0.0222 - val_accuracy: 0.9916
Epoch 3/10
4178/4178 [==============================] - 134s 32ms/step - loss: 0.0252 -
accuracy: 0.9906 - val_loss: 0.0252 - val_accuracy: 0.9906
Epoch 4/10
4178/4178 [==============================] - 136s 32ms/step - loss: 0.0254 -
accuracy: 0.9906 - val_loss: 0.0273 - val_accuracy: 0.9902
Epoch 5/10
4178/4178 [==============================] - 134s 32ms/step - loss: 0.0236 -
accuracy: 0.9913 - val_loss: 0.0233 - val_accuracy: 0.9906
Epoch 6/10
4178/4178 [==============================] - 133s 32ms/step - loss: 0.0235 -
accuracy: 0.9910 - val_loss: 0.0245 - val_accuracy: 0.9906
```

```
Epoch 7/10
4178/4178 [==============================] - 134s 32ms/step - loss: 0.0229 -
accuracy: 0.9912 - val_loss: 0.0282 - val_accuracy: 0.9904
Epoch 8/10
4178/4178 [==============================] - 136s 33ms/step - loss: 0.0229 -
accuracy: 0.9912 - val_loss: 0.0249 - val_accuracy: 0.9910
Epoch 9/10
4178/4178 [==============================] - 133s 32ms/step - loss: 0.0221 -
accuracy: 0.9918 - val_loss: 0.0261 - val_accuracy: 0.9917
Epoch 10/10
4178/4178 [==============================] - 134s 32ms/step - loss: 0.0217 -
accuracy: 0.9919 - val_loss: 0.0233 - val_accuracy: 0.9923
465/465 [==============================] - 5s 11ms/step
Validation score: 0.9923242660921088
train index: [     0      1      2 … 148513 148514 148516]
test index: [    16     26     38 … 148490 148499 148515]
Epoch 1/10
4178/4178 [==============================] - 135s 32ms/step - loss: 0.0218 -
accuracy: 0.9917 - val_loss: 0.0216 - val_accuracy: 0.9916
Epoch 2/10
4178/4178 [==============================] - 133s 32ms/step - loss: 0.0202 -
accuracy: 0.9920 - val_loss: 0.0224 - val_accuracy: 0.9915
Epoch 3/10
4178/4178 [==============================] - 134s 32ms/step - loss: 0.0207 -
accuracy: 0.9921 - val_loss: 0.0230 - val_accuracy: 0.9908
Epoch 4/10
4178/4178 [==============================] - 134s 32ms/step - loss: 0.0192 -
accuracy: 0.9922 - val_loss: 0.0217 - val_accuracy: 0.9931
Epoch 5/10
4178/4178 [==============================] - 133s 32ms/step - loss: 0.0201 -
accuracy: 0.9921 - val_loss: 0.0226 - val_accuracy: 0.9914
Epoch 6/10
4178/4178 [==============================] - 135s 32ms/step - loss: 0.0198 -
accuracy: 0.9924 - val_loss: 0.0226 - val_accuracy: 0.9923
Epoch 7/10
4178/4178 [==============================] - 133s 32ms/step - loss: 0.0201 -
accuracy: 0.9921 - val_loss: 0.0238 - val_accuracy: 0.9909
Epoch 8/10
4178/4178 [==============================] - 134s 32ms/step - loss: 0.0195 -
accuracy: 0.9925 - val_loss: 0.0229 - val_accuracy: 0.9923
Epoch 9/10
4178/4178 [==============================] - 135s 32ms/step - loss: 0.0185 -
accuracy: 0.9925 - val_loss: 0.0275 - val_accuracy: 0.9915
Epoch 10/10
4178/4178 [==============================] - 134s 32ms/step - loss: 0.0191 -
accuracy: 0.9925 - val_loss: 0.0258 - val_accuracy: 0.9896
465/465 [==============================] - 5s 11ms/step
Validation score: 0.9896310261244277
```

```
train index: [     0      1      2 … 148514 148515 148516]
test index: [    10     42     44 … 148465 148474 148507]
Epoch 1/10
4178/4178 [==============================] - 139s 33ms/step - loss: 0.0186 -
accuracy: 0.9927 - val_loss: 0.0158 - val_accuracy: 0.9940
Epoch 2/10
4178/4178 [==============================] - 137s 33ms/step - loss: 0.0195 -
accuracy: 0.9925 - val_loss: 0.0155 - val_accuracy: 0.9929
Epoch 3/10
4178/4178 [==============================] - 136s 32ms/step - loss: 0.0188 -
accuracy: 0.9928 - val_loss: 0.0176 - val_accuracy: 0.9931
Epoch 4/10
4178/4178 [==============================] - 133s 32ms/step - loss: 0.0186 -
accuracy: 0.9930 - val_loss: 0.0157 - val_accuracy: 0.9943
Epoch 5/10
4178/4178 [==============================] - 136s 32ms/step - loss: 0.0186 -
accuracy: 0.9926 - val_loss: 0.0164 - val_accuracy: 0.9937
Epoch 6/10
4178/4178 [==============================] - 134s 32ms/step - loss: 0.0184 -
accuracy: 0.9929 - val_loss: 0.0195 - val_accuracy: 0.9925
Epoch 7/10
4178/4178 [==============================] - 134s 32ms/step - loss: 0.0182 -
accuracy: 0.9927 - val_loss: 0.0184 - val_accuracy: 0.9925
Epoch 8/10
4178/4178 [==============================] - 137s 33ms/step - loss: 0.0179 -
accuracy: 0.9929 - val_loss: 0.0155 - val_accuracy: 0.9942
Epoch 9/10
4178/4178 [==============================] - 138s 33ms/step - loss: 0.0176 -
accuracy: 0.9931 - val_loss: 0.0150 - val_accuracy: 0.9942
Epoch 10/10
4178/4178 [==============================] - 136s 33ms/step - loss: 0.0174 -
accuracy: 0.9932 - val_loss: 0.0162 - val_accuracy: 0.9932
110/465 [======>…] - ETA: 3s
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_20220\1338108876.py in <module>
     26     model.fit(x_train_1, y_train_1,validation_data=(x_test_2,y_test_2),
  ↪epochs=10)
     27 
---> 28     pred = model.predict(x_test_2)
     29     pred = np.argmax(pred,axis=1)
     30     y_eval = np.argmax(y_test_2,axis=1)

~\anaconda3\lib\site-packages\keras\utils\traceback_utils.py in␣
  ↪error_handler(*args, **kwargs)
     63         filtered_tb = None
     64         try:
---> 65             return fn(*args, **kwargs)
     66         except Exception as e:
     67             filtered_tb = _process_traceback_frames(e.__traceback__)

~\anaconda3\lib\site-packages\keras\engine\training.py in predict(self, x,␣
 ↪batch_size, verbose, steps, callbacks, max_queue_size, workers,␣
 ↪use_multiprocessing)
   2380                         for step in data_handler.steps():
   2381                             callbacks.on_predict_batch_begin(step)
-> 2382                             tmp_batch_outputs = self.
 ↪predict_function(iterator)
   2383                             if data_handler.should_sync:
   2384                                 context.async_wait()

~\anaconda3\lib\site-packages\tensorflow\python\util\traceback_utils.py in␣
 ↪error_handler(*args, **kwargs)
    148     filtered_tb = None
    149     try:
--> 150       return fn(*args, **kwargs)
    151     except Exception as e:
    152       filtered_tb = _process_traceback_frames(e.__traceback__)

~\anaconda3\lib\site-packages\tensorflow\python\eager\polymorphic_function\polymorphic_functi
 ↪py in __call__(self, *args, **kwds)
    892
    893       with OptionalXlaContext(self._jit_compile):
--> 894         result = self._call(*args, **kwds)
    895
    896       new_tracing_count = self.experimental_get_tracing_count()

~\anaconda3\lib\site-packages\tensorflow\python\eager\polymorphic_function\polymorphic_functi
 ↪py in _call(self, *args, **kwds)
    931       # In this case we have not created variables on the first call. So␣
 ↪we can
    932       # run the first trace but we should fail if variables are created.
--> 933       results = self._variable_creation_fn(*args, **kwds)
    934       if self._created_variables and not ALLOW_DYNAMIC_VARIABLE_CREATION:
    935         raise ValueError("Creating variables on a non-first call to a␣
 ↪function"

~\anaconda3\lib\site-packages\tensorflow\python\eager\polymorphic_function\tracing_compiler.
 ↪py in __call__(self, *args, **kwargs)
    141       (concrete_function,
    142        filtered_flat_args) = self._maybe_define_function(args, kwargs)
--> 143     return concrete_function._call_flat(

    144         filtered_flat_args, captured_inputs=concrete_function.
 ↪captured_inputs)  # pylint: disable=protected-access
```

```
    145

~\anaconda3\lib\site-packages\tensorflow\python\eager\polymorphic_function\monomorphic_functi
 ↪py in _call_flat(self, args, captured_inputs, cancellation_manager)
   1755         and executing_eagerly):
   1756       # No tape is watching; skip to running the function.
-> 1757       return self._build_call_outputs(self._inference_function.call(

   1758           ctx, args, cancellation_manager=cancellation_manager))
   1759     forward_backward = self._select_forward_and_backward_functions(

~\anaconda3\lib\site-packages\tensorflow\python\eager\polymorphic_function\monomorphic_functi
 ↪py in call(self, ctx, args, cancellation_manager)
    379       with _InterpolateFunctionError(self):
    380         if cancellation_manager is None:
--> 381           outputs = execute.execute(

    382               str(self.signature.name),
    383               num_outputs=self._num_outputs,

~\anaconda3\lib\site-packages\tensorflow\python\eager\execute.py in␣
 ↪quick_execute(op_name, num_outputs, inputs, attrs, ctx, name)
     50   try:
     51     ctx.ensure_initialized()
---> 52     tensors = pywrap_tfe.TFE_Py_Execute(ctx._handle, device_name,␣
 ↪op_name,

     53                                         inputs, attrs, num_outputs)
     54   except core._NotOkStatusException as e:

KeyboardInterrupt:
```

Nous pourrions laisser l'algorithme croitre son accuracy, mais pour des raisons de processus nous avons interrompu le kernel

[84]:
```
oos_pred
```

[84]:
```
[0.9885537301373553, 0.9923242660921088, 0.9896310261244277]
```

[85]:
```
dummies_test.columns
```

[85]:
```
Index(['DOS', 'Normal', 'Probe', 'R2L', 'U2R'], dtype='object')
```

[86]:
```
test_y.value_counts()
```

[86]:
```
Normal    7723
DOS       5339
Probe     1408
R2L        370
```

```
U2R           12
Name: Class, dtype: int64
```

[95]:
```
#End of preprocessing step
# Save the preprocessed dataset
df.to_csv('preprocessed_dataset_NSLKDD.csv', index=False)
```

[96]:
```
df.shape
```

[96]:
```
(125973, 42)
```

[108]:
```
# Get the unique classes in the target column
df['subclass'].value_counts()
```

[108]:
```
normal             67343
neptune            41214
satan               3633
ipsweep             3599
portsweep           2931
smurf               2646
nmap                1493
back                 956
teardrop             892
warezclient          890
pod                  201
guess_passwd          53
buffer_overflow       30
warezmaster           20
land                  18
imap                  11
rootkit               10
loadmodule             9
ftp_write              8
multihop               7
phf                    4
perl                   3
spy                    2
Name: subclass, dtype: int64
```

[109]:
```
# Get the unique classes in the target column
df['subclass'].unique()
```

[109]:
```
array(['normal', 'neptune', 'warezclient', 'ipsweep', 'portsweep',
       'teardrop', 'nmap', 'satan', 'smurf', 'pod', 'back',
       'guess_passwd', 'ftp_write', 'multihop', 'rootkit',
       'buffer_overflow', 'imap', 'warezmaster', 'phf', 'land',
       'loadmodule', 'spy', 'perl'], dtype=object)
```

```
[ ]:
```