1_Data_Prepocessing

June 5, 2024

1 Data Cleanup and Preprocessing

Before using the CIC-IDS 2017 Dataset, the data has to be preprocessed and cleaned. The raw files are 7 csv files containing the recorded network traffic for 5 working days with benign traffic and various attacks (Brute Force Attack, Heart Bleed Attack, Botnet, Dos Attack, DDos Attack, Web Attack (SQL Injection, XSS, Brute Force), Infiltration Attack)

[1]:
```
import numpy as np
import pandas as pd
import os
```

1.1 1. Exploring one file from the dataset

To understand the dataset, one file is loaded and analyzed before processing all of the files

[2]:
```
dataset_path = r"CIC-IDS-2017\CSVs\GeneratedLabelledFlows\TrafficLabelling"
file_path = os.path.join(dataset_path, "Monday-WorkingHours.pcap_ISCX.csv")
df = pd.read_csv(file_path)
# Remove space in column names using strip() function
df.rename(columns=lambda x: x.strip(), inplace=True)
# Remove columns unnecessary for machine learning
df = df.drop(columns=['Flow ID', 'Source IP', 'Source Port', 'Destination IP',␣
 ↪'Timestamp'])
df.head()
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 529918 entries, 0 to 529917
Data columns (total 80 columns):
 #   Column                       Non-Null Count   Dtype
---  ------                       --------------   -----
 0   Destination Port             529918 non-null  int64
 1   Protocol                     529918 non-null  int64
 2   Flow Duration                529918 non-null  int64
 3   Total Fwd Packets            529918 non-null  int64
 4   Total Backward Packets       529918 non-null  int64
 5   Total Length of Fwd Packets  529918 non-null  float64
 6   Total Length of Bwd Packets  529918 non-null  float64
 7   Fwd Packet Length Max        529918 non-null  float64
```

```
8   Fwd Packet Length Min        529918 non-null  float64
9   Fwd Packet Length Mean       529918 non-null  float64
10  Fwd Packet Length Std        529918 non-null  float64
11  Bwd Packet Length Max        529918 non-null  float64
12  Bwd Packet Length Min        529918 non-null  float64
13  Bwd Packet Length Mean       529918 non-null  float64
14  Bwd Packet Length Std        529918 non-null  float64
15  Flow Bytes/s                 529854 non-null  float64
16  Flow Packets/s               529918 non-null  float64
17  Flow IAT Mean                529918 non-null  float64
18  Flow IAT Std                 529918 non-null  float64
19  Flow IAT Max                 529918 non-null  float64
20  Flow IAT Min                 529918 non-null  float64
21  Fwd IAT Total                529918 non-null  float64
22  Fwd IAT Mean                 529918 non-null  float64
23  Fwd IAT Std                  529918 non-null  float64
24  Fwd IAT Max                  529918 non-null  float64
25  Fwd IAT Min                  529918 non-null  float64
26  Bwd IAT Total                529918 non-null  float64
27  Bwd IAT Mean                 529918 non-null  float64
28  Bwd IAT Std                  529918 non-null  float64
29  Bwd IAT Max                  529918 non-null  float64
30  Bwd IAT Min                  529918 non-null  float64
31  Fwd PSH Flags                529918 non-null  int64
32  Bwd PSH Flags                529918 non-null  int64
33  Fwd URG Flags                529918 non-null  int64
34  Bwd URG Flags                529918 non-null  int64
35  Fwd Header Length            529918 non-null  int64
36  Bwd Header Length            529918 non-null  int64
37  Fwd Packets/s                529918 non-null  float64
38  Bwd Packets/s                529918 non-null  float64
39  Min Packet Length            529918 non-null  float64
40  Max Packet Length            529918 non-null  float64
41  Packet Length Mean           529918 non-null  float64
42  Packet Length Std            529918 non-null  float64
43  Packet Length Variance       529918 non-null  float64
44  FIN Flag Count               529918 non-null  int64
45  SYN Flag Count               529918 non-null  int64
46  RST Flag Count               529918 non-null  int64
47  PSH Flag Count               529918 non-null  int64
48  ACK Flag Count               529918 non-null  int64
49  URG Flag Count               529918 non-null  int64
50  CWE Flag Count               529918 non-null  int64
51  ECE Flag Count               529918 non-null  int64
52  Down/Up Ratio                529918 non-null  float64
53  Average Packet Size          529918 non-null  float64
54  Avg Fwd Segment Size         529918 non-null  float64
55  Avg Bwd Segment Size         529918 non-null  float64
```

```
 56  Fwd Header Length.1          529918 non-null  int64
 57  Fwd Avg Bytes/Bulk           529918 non-null  int64
 58  Fwd Avg Packets/Bulk         529918 non-null  int64
 59  Fwd Avg Bulk Rate            529918 non-null  int64
 60  Bwd Avg Bytes/Bulk           529918 non-null  int64
 61  Bwd Avg Packets/Bulk         529918 non-null  int64
 62  Bwd Avg Bulk Rate            529918 non-null  int64
 63  Subflow Fwd Packets          529918 non-null  int64
 64  Subflow Fwd Bytes            529918 non-null  int64
 65  Subflow Bwd Packets          529918 non-null  int64
 66  Subflow Bwd Bytes            529918 non-null  int64
 67  Init_Win_bytes_forward       529918 non-null  int64
 68  Init_Win_bytes_backward      529918 non-null  int64
 69  act_data_pkt_fwd             529918 non-null  int64
 70  min_seg_size_forward         529918 non-null  int64
 71  Active Mean                  529918 non-null  float64
 72  Active Std                   529918 non-null  float64
 73  Active Max                   529918 non-null  float64
 74  Active Min                   529918 non-null  float64
 75  Idle Mean                    529918 non-null  float64
 76  Idle Std                     529918 non-null  float64
 77  Idle Max                     529918 non-null  float64
 78  Idle Min                     529918 non-null  float64
 79  Label                        529918 non-null  object
dtypes: float64(45), int64(34), object(1)
memory usage: 323.4+ MB
```

Convert label type to category

[3]:
```
convert_dict = {'Label': 'category'}
df = df.astype(convert_dict)
```

Check for infinity and null values in one of the columns

[4]:
```
print(f"Infinity values of flow_byts_s: {df[df['Flow Bytes/s'] == np.
  ↪inf]['Destination Port'].count()}")
print(f"Null values of flow_byts_s: {df[df['Flow Bytes/s'].
  ↪isnull()]['Destination Port'].count()}")
```

```
Infinity values of flow_byts_s: 373
Null values of flow_byts_s: 64
```

1.2 2. Cleanup all Files

The "Thursday-WorkingHours-Morning-WebAttacks.pcap_ISCX.csv" contains a problematic character that has to be replaced before processing all files and empty lines at the end of the file.

[5]:
```
problematic_file = "Thursday-WorkingHours-Morning-WebAttacks.pcap_ISCX.csv"
problematic_file_path = os.path.join(dataset_path, problematic_file)
```

```
with open(problematic_file_path, 'rb') as file:
    content = file.read()
# Replace the problematic character (0x96) with a hyphen (-)
content_fixed = content.replace(b'\x96', b'-')
# Split the content into lines and remove lines that contain only commas
lines = content_fixed.decode('utf-8').split('\n')
cleaned_lines = [line for line in lines if not all(char == ',' for char in line.
 ↪strip())]
with open(problematic_file_path, 'wb') as file:
    file.write('\n'.join(cleaned_lines).encode('utf-8'))
```

To cleanup the rest of the files: 1. Trim column names of whitespaces and convert them to lowercase 2. Drop the columns that are unnecessary for machine learning

[6]:
```
import re
files = {
    "Monday-WorkingHours.pcap_ISCX.csv": "Monday.csv",
    "Tuesday-WorkingHours.pcap_ISCX.csv": "Tuesday.csv",
    "Wednesday-workingHours.pcap_ISCX.csv": "Wednesday.csv",
    "Thursday-WorkingHours-Morning-WebAttacks.pcap_ISCX.csv":␣
 ↪"Thursday-Morning-WebAttacks.csv",
    "Thursday-WorkingHours-Afternoon-Infilteration.pcap_ISCX.csv":␣
 ↪"Thursday-Afternoon-Infiltration.csv",
    "Friday-WorkingHours-Morning.pcap_ISCX.csv": "Friday-Morning.csv",
    "Friday-WorkingHours-Afternoon-PortScan.pcap_ISCX.csv":␣
 ↪"Friday-Afternoon-Portscan.csv",
    "Friday-WorkingHours-Afternoon-DDos.pcap_ISCX.csv": "Friday-Afternoon-DDos.
 ↪csv"}
column_name_regex = re.compile(r"\W", re.IGNORECASE)
processed_dir = "processed"
processed_path = os.path.join(dataset_path, processed_dir)
def trim_column_names(df):
    return [column_name_regex.sub('_', c.lower()) for c in df.columns]
if not os.path.exists(processed_path):
    os.mkdir(processed_path)
for file_in, file_out in files.items():
    file_path = os.path.join(dataset_path, file_in)
    output_path = os.path.join(processed_path, file_out)
    df = pd.read_csv(file_path)
    df.rename(columns=lambda x: x.strip(), inplace=True)
    df = df.drop(columns=['Flow ID', 'Source IP', 'Source Port', 'Destination␣
 ↪IP', 'Timestamp'])
    df.columns = trim_column_names(df)
    df.to_csv(output_path, index=False)
    print("Labels for file:", file_in)
    print(df['label'].value_counts())
```

```
Labels for file: Monday-WorkingHours.pcap_ISCX.csv
```

```
label
BENIGN    529918
Name: count, dtype: int64
Labels for file: Tuesday-WorkingHours.pcap_ISCX.csv
label
BENIGN         432074
FTP-Patator      7938
SSH-Patator      5897
Name: count, dtype: int64
Labels for file: Wednesday-workingHours.pcap_ISCX.csv
label
BENIGN              440031
DoS Hulk            231073
DoS GoldenEye        10293
DoS slowloris         5796
DoS Slowhttptest      5499
Heartbleed              11
Name: count, dtype: int64
Labels for file: Thursday-WorkingHours-Morning-WebAttacks.pcap_ISCX.csv
label
BENIGN                        168186
Web Attack - Brute Force        1507
Web Attack - XSS                 652
Web Attack - Sql Injection        21
Name: count, dtype: int64
Labels for file: Thursday-WorkingHours-Afternoon-Infilteration.pcap_ISCX.csv
label
BENIGN          288566
Infiltration        36
Name: count, dtype: int64
Labels for file: Friday-WorkingHours-Morning.pcap_ISCX.csv
label
BENIGN    189067
Bot         1966
Name: count, dtype: int64
Labels for file: Friday-WorkingHours-Afternoon-PortScan.pcap_ISCX.csv
label
PortScan    158930
BENIGN      127537
Name: count, dtype: int64
Labels for file: Friday-WorkingHours-Afternoon-DDos.pcap_ISCX.csv
label
DDoS      128027
BENIGN     97718
Name: count, dtype: int64
```

1.3 3. Data Preparation

All of the processed datasets are grouped into one Pandas dataframe to analyze the content. The data is then saved into one single csv file.

[7]:
```
import glob
csv_files = glob.glob(os.path.join(processed_path, '*.csv'))
df = pd.concat((pd.read_csv(f) for f in csv_files))
```

[8]:
```
df.head()
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 2830743 entries, 0 to 225744
Data columns (total 80 columns):
 #   Column                       Dtype
---  ------                       -----
 0   destination_port             int64
 1   protocol                     int64
 2   flow_duration                int64
 3   total_fwd_packets            int64
 4   total_backward_packets       int64
 5   total_length_of_fwd_packets  float64
 6   total_length_of_bwd_packets  float64
 7   fwd_packet_length_max        float64
 8   fwd_packet_length_min        float64
 9   fwd_packet_length_mean       float64
 10  fwd_packet_length_std        float64
 11  bwd_packet_length_max        float64
 12  bwd_packet_length_min        float64
 13  bwd_packet_length_mean       float64
 14  bwd_packet_length_std        float64
 15  flow_bytes_s                 float64
 16  flow_packets_s               float64
 17  flow_iat_mean                float64
 18  flow_iat_std                 float64
 19  flow_iat_max                 float64
 20  flow_iat_min                 float64
 21  fwd_iat_total                float64
 22  fwd_iat_mean                 float64
 23  fwd_iat_std                  float64
 24  fwd_iat_max                  float64
 25  fwd_iat_min                  float64
 26  bwd_iat_total                float64
 27  bwd_iat_mean                 float64
 28  bwd_iat_std                  float64
 29  bwd_iat_max                  float64
 30  bwd_iat_min                  float64
 31  fwd_psh_flags                int64
```

```
32  bwd_psh_flags               int64
33  fwd_urg_flags               int64
34  bwd_urg_flags               int64
35  fwd_header_length           int64
36  bwd_header_length           int64
37  fwd_packets_s               float64
38  bwd_packets_s               float64
39  min_packet_length           float64
40  max_packet_length           float64
41  packet_length_mean          float64
42  packet_length_std           float64
43  packet_length_variance      float64
44  fin_flag_count              int64
45  syn_flag_count              int64
46  rst_flag_count              int64
47  psh_flag_count              int64
48  ack_flag_count              int64
49  urg_flag_count              int64
50  cwe_flag_count              int64
51  ece_flag_count              int64
52  down_up_ratio               float64
53  average_packet_size         float64
54  avg_fwd_segment_size        float64
55  avg_bwd_segment_size        float64
56  fwd_header_length_1         int64
57  fwd_avg_bytes_bulk          int64
58  fwd_avg_packets_bulk        int64
59  fwd_avg_bulk_rate           int64
60  bwd_avg_bytes_bulk          int64
61  bwd_avg_packets_bulk        int64
62  bwd_avg_bulk_rate           int64
63  subflow_fwd_packets         int64
64  subflow_fwd_bytes           int64
65  subflow_bwd_packets         int64
66  subflow_bwd_bytes           int64
67  init_win_bytes_forward      int64
68  init_win_bytes_backward     int64
69  act_data_pkt_fwd            int64
70  min_seg_size_forward        int64
71  active_mean                 float64
72  active_std                  float64
73  active_max                  float64
74  active_min                  float64
75  idle_mean                   float64
76  idle_std                    float64
77  idle_max                    float64
78  idle_min                    float64
79  label                       object
```

```
dtypes: float64(45), int64(34), object(1)
memory usage: 1.7+ GB
```

Creating labels for the attacks

[9]:
```
import re
df['is_attack'] = df.label.apply(lambda x: 0 if x == "BENIGN" else 1)
convert_dict = {'label': 'category'}
df = df.astype(convert_dict)
# Having attack types as integers can be helpful for some machine learning␣
 ↪algorithms
df['label_code'] = df['label'].cat.codes
attacks = df["label"].value_counts().index.tolist()
for attack in attacks:
    if attack != "BENIGN":
        attack = attack.lower().replace('-', '')
        l = "is_" + re.sub(r'\s+', '_', attack)
        df[l] = df.label.apply(lambda x: 1 if x == attack else 0)
df.info(verbose = True)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 2830743 entries, 0 to 225744
Data columns (total 96 columns):
 #   Column                       Dtype
---  ------                       -----
 0   destination_port             int64
 1   protocol                     int64
 2   flow_duration                int64
 3   total_fwd_packets            int64
 4   total_backward_packets       int64
 5   total_length_of_fwd_packets  float64
 6   total_length_of_bwd_packets  float64
 7   fwd_packet_length_max        float64
 8   fwd_packet_length_min        float64
 9   fwd_packet_length_mean       float64
 10  fwd_packet_length_std        float64
 11  bwd_packet_length_max        float64
 12  bwd_packet_length_min        float64
 13  bwd_packet_length_mean       float64
 14  bwd_packet_length_std        float64
 15  flow_bytes_s                 float64
 16  flow_packets_s               float64
 17  flow_iat_mean                float64
 18  flow_iat_std                 float64
 19  flow_iat_max                 float64
 20  flow_iat_min                 float64
 21  fwd_iat_total                float64
 22  fwd_iat_mean                 float64
 23  fwd_iat_std                  float64
```

```
24  fwd_iat_max                 float64
25  fwd_iat_min                 float64
26  bwd_iat_total               float64
27  bwd_iat_mean                float64
28  bwd_iat_std                 float64
29  bwd_iat_max                 float64
30  bwd_iat_min                 float64
31  fwd_psh_flags               int64
32  bwd_psh_flags               int64
33  fwd_urg_flags               int64
34  bwd_urg_flags               int64
35  fwd_header_length           int64
36  bwd_header_length           int64
37  fwd_packets_s               float64
38  bwd_packets_s               float64
39  min_packet_length           float64
40  max_packet_length           float64
41  packet_length_mean          float64
42  packet_length_std           float64
43  packet_length_variance      float64
44  fin_flag_count              int64
45  syn_flag_count              int64
46  rst_flag_count              int64
47  psh_flag_count              int64
48  ack_flag_count              int64
49  urg_flag_count              int64
50  cwe_flag_count              int64
51  ece_flag_count              int64
52  down_up_ratio               float64
53  average_packet_size         float64
54  avg_fwd_segment_size        float64
55  avg_bwd_segment_size        float64
56  fwd_header_length_1         int64
57  fwd_avg_bytes_bulk          int64
58  fwd_avg_packets_bulk        int64
59  fwd_avg_bulk_rate           int64
60  bwd_avg_bytes_bulk          int64
61  bwd_avg_packets_bulk        int64
62  bwd_avg_bulk_rate           int64
63  subflow_fwd_packets         int64
64  subflow_fwd_bytes           int64
65  subflow_bwd_packets         int64
66  subflow_bwd_bytes           int64
67  init_win_bytes_forward      int64
68  init_win_bytes_backward     int64
69  act_data_pkt_fwd            int64
70  min_seg_size_forward        int64
71  active_mean                 float64
```

```
 72  active_std                   float64
 73  active_max                   float64
 74  active_min                   float64
 75  idle_mean                    float64
 76  idle_std                     float64
 77  idle_max                     float64
 78  idle_min                     float64
 79  label                        category
 80  is_attack                    int64
 81  label_code                   int8
 82  is_dos_hulk                  int64
 83  is_portscan                  int64
 84  is_ddos                      int64
 85  is_dos_goldeneye             int64
 86  is_ftppatator                int64
 87  is_sshpatator                int64
 88  is_dos_slowloris             int64
 89  is_dos_slowhttptest          int64
 90  is_bot                       int64
 91  is_web_attack_brute_force    int64
 92  is_web_attack_xss            int64
 93  is_infiltration              int64
 94  is_web_attack_sql_injection  int64
 95  is_heartbleed                int64
dtypes: category(1), float64(45), int64(49), int8(1)
memory usage: 2.0 GB
```

Saving the grouped dataset to a single file

[10]:
```
output_path = os.path.join(processed_path, "ids2017_processed.csv")
df.to_csv(output_path, index = False)
```