task_1_EDA

February 22, 2026

1 Mount Drive

[1]:
```
# 1) Mount Google Drive (DO NOT EDIT)

# from google.colab import drive
# drive.mount('/content/drive')

# High quality image
%config InlineBackend.figure_format = 'png'

# Imports
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import time

# Scikit-Learn Models & Tools
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GridSearchCV, train_test_split
from sklearn.metrics import classification_report, confusion_matrix,␣
 ↪accuracy_score, f1_score, log_loss
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.base import BaseEstimator, TransformerMixin
```

[2]:
```
# === EDIT THIS CELL ONLY ===
# set file name
# For Local Env Path (USED ONLY LOCAL ENVIRONMENT)
TRAIN_PATH = "../Group_Assignment_Database/UNSW_NB15_training-set.csv"
TEST_PATH = "../Group_Assignment_Database/UNSW_NB15_testing-set.csv"

# ====================================
train_csv_raw = pd.read_csv(TRAIN_PATH)
test_csv_raw = pd.read_csv(TEST_PATH)
```

2 Dataset Loader

[3]:
```
import pandas as pd
import os
import seaborn as sns
import matplotlib.pyplot as plt

def load_and_prep_data():
    train_path = '/content/drive/MyDrive/University/CSCI316 - Big Data Mining␣
 ↪Techniques/Group Assignment/UNSW_NB15_training-set.csv'
    test_path = '/content/drive/MyDrive/University/CSCI316 - Big Data Mining␣
 ↪Techniques/Group Assignment/UNSW_NB15_testing-set.csv'

    # Juwon's Local file path
    test_path = '/Users/jju/Documents/SIM/Semester 1, 2026/CSCI316 Big Data␣
 ↪Mining/Assignments/Group_Assignment_Database/UNSW_NB15_testing-set.csv'
    train_path = '/Users/jju/Documents/SIM/Semester 1, 2026/CSCI316 Big Data␣
 ↪Mining/Assignments/Group_Assignment_Database/UNSW_NB15_training-set.csv'

    # 1. Load both datasets
    print("Loading data...")
    df_train_orig = pd.read_csv(train_path)
    df_test_orig = pd.read_csv(test_path)

    # 2. Combine them
    df_full = pd.concat([df_train_orig, df_test_orig], axis=0).
 ↪reset_index(drop=True)

    missing_data = df_full.isnull().sum()
    if missing_data.sum() > 0:
        print("\n[WARNING] Missing values detected:")
        print(missing_data[missing_data > 0])
    else:
        print("\n[CHECK] No true missing values (NaN) found.")

    # 3. Define Features and Target
    # We use 'label' for stratification to maintain class balance
    drop_cols = ['label', 'id', 'attack_cat']
    X = df_full.drop(drop_cols, axis=1)
    y = df_full['label']

    # 4. Stratified Split: Train (70%) and Temp (30%)
    X_train, X_temp, y_train, y_temp = train_test_split(
        X, y, test_size=0.3, random_state=42, stratify=y
    )

    # 5. Split Temp into Validation (15%) and Test (15%)
```

```
    # 0.5 * 30% = 15%
    X_val, X_test, y_val, y_test = train_test_split(
        X_temp, y_temp, test_size=0.5, random_state=42, stratify=y_temp
    )

    # Now X_train, X_val, X_test/ y_train, y_val, y_test available

    # 6. Preprocessing (Label Encoding)
    le = LabelEncoder()
    categorical_cols = ['proto', 'service', 'state'] # attack_cat removed since already dropped
    for col in categorical_cols:
        # labelEncode each train/test/validation data
        X_train[col] = le.fit_transform(X_train[col].astype(str))
        X_val[col] = le.fit_transform(X_val[col].astype(str))
        X_test[col] = le.fit_transform(X_test[col].astype(str))



    print(f"Features used for training ({len(X.columns)} total):")
    print(list(X.columns))
    print("-" * 50)

    print(f"Data Loaded and Split:")
    print(f"Train: {X_train.shape}, Val: {X_val.shape}, Test: {X_test.shape}")

    return X_train, y_train, X_val, y_val, X_test, y_test

# Unpack the validation set
X_train, y_train, X_val, y_val, X_test, y_test = load_and_prep_data()

def check_distributions(y_train, y_val, y_test):
    sets = {'Train': y_train, 'Validation': y_val, 'Test': y_test}
    for name, set_data in sets.items():
        counts = set_data.value_counts(normalize=True) * 100
        print(f"{name} Distribution: Normal: {counts[0]:.2f}%, Attack: {counts[1]:.2f}%")

check_distributions(y_train, y_val, y_test)
```

```
Loading data…

[CHECK] No true missing values (NaN) found.
Features used for training (42 total):
['dur', 'proto', 'service', 'state', 'spkts', 'dpkts', 'sbytes', 'dbytes',
'rate', 'sttl', 'dttl', 'sload', 'dload', 'sloss', 'dloss', 'sinpkt', 'dinpkt',
'sjit', 'djit', 'swin', 'stcpb', 'dtcpb', 'dwin', 'tcprtt', 'synack', 'ackdat',
```

```
'smean', 'dmean', 'trans_depth', 'response_body_len', 'ct_srv_src',
'ct_state_ttl', 'ct_dst_ltm', 'ct_src_dport_ltm', 'ct_dst_sport_ltm',
'ct_dst_src_ltm', 'is_ftp_login', 'ct_ftp_cmd', 'ct_flw_http_mthd',
'ct_src_ltm', 'ct_srv_dst', 'is_sm_ips_ports']
--------------------------------------------------
Data Loaded and Split:
Train: (180371, 42), Val: (38651, 42), Test: (38651, 42)
Train Distribution: Normal: 36.09%, Attack: 63.91%
Validation Distribution: Normal: 36.09%, Attack: 63.91%
Test Distribution: Normal: 36.09%, Attack: 63.91%
```

[4]:
```
print("=" * 60)
print("=" * 60)
print("Checking All Unique Values in Each Feature")
print("=" * 60)

for col in train_csv_raw.columns:
    print("="*40)
    print(col)
    print()
    print(train_csv_raw[col].unique())
```

```
============================================================
============================================================
Checking All Unique Values in Each Feature
============================================================
============================================================
id

[     1      2      3 … 175339 175340 175341]
========================================
dur

[0.121478 0.649902 1.623129 … 3.71911  0.996503 1.557125]
========================================
proto

['tcp' 'udp' 'arp' 'ospf' 'icmp' 'igmp' 'rtp' 'ddp' 'ipv6-frag' 'cftp'
 'wsn' 'pvp' 'wb-expak' 'mtp' 'pri-enc' 'sat-mon' 'cphb' 'sun-nd' 'iso-ip'
 'xtp' 'il' 'unas' 'mfe-nsp' '3pc' 'ipv6-route' 'idrp' 'bna' 'swipe'
 'kryptolan' 'cpnx' 'rsvp' 'wb-mon' 'vmtp' 'ib' 'dgp' 'eigrp' 'ax.25'
 'gmtp' 'pnni' 'sep' 'pgm' 'idpr-cmtp' 'zero' 'rvd' 'mobile' 'narp' 'fc'
 'pipe' 'ipcomp' 'ipv6-no' 'sat-expak' 'ipv6-opts' 'snp' 'ipcv'
 'br-sat-mon' 'ttp' 'tcf' 'nsfnet-igp' 'sprite-rpc' 'aes-sp3-d' 'sccopmce'
 'sctp' 'qnx' 'scps' 'etherip' 'aris' 'pim' 'compaq-peer' 'vrrp' 'iatp'
 'stp' 'l2tp' 'srp' 'sm' 'isis' 'smp' 'fire' 'ptp' 'crtp' 'sps'
 'merit-inp' 'idpr' 'skip' 'any' 'larp' 'ipip' 'micp' 'encap' 'ifmp'
 'tp++' 'a/n' 'ipv6' 'i-nlsp' 'ipx-n-ip' 'sdrp' 'tlsp' 'gre' 'mhrp' 'ddx'
```

```
 'ippc' 'visa' 'secure-vmtp' 'uti' 'vines' 'crudp' 'iplt' 'ggp' 'ip'
 'ipnip' 'st2' 'argus' 'bbn-rcc' 'egp' 'emcon' 'igp' 'nvp' 'pup' 'xnet'
 'chaos' 'mux' 'dcn' 'hmp' 'prm' 'trunk-1' 'xns-idp' 'leaf-1' 'leaf-2'
 'rdp' 'irtp' 'iso-tp4' 'netblt' 'trunk-2' 'cbt']
=======================================
service

['-' 'ftp' 'smtp' 'snmp' 'http' 'ftp-data' 'dns' 'ssh' 'radius' 'pop3'
 'dhcp' 'ssl' 'irc']
=======================================
state

['FIN' 'INT' 'CON' 'ECO' 'REQ' 'RST' 'PAR' 'URN' 'no']
=======================================
spkts

[   6   14    8   12   10   62    2   24   28   30    1   16    4   36
   22  122   38   20   66   78   42   52   64   40   80    3   26  110
  106  360  354  366  352  108  362   18  342  332  336  308  322  328
  358  224  234  222  208  200   68   48   90  356   70  348  236   92
  240  452  450  368  434  420  238  432  288  112  650  656  364  548
   50  230  562  372  216  204   60  228  206  350  294  346  232   46
  226  218  296  190  286   88  338   32  344  326  340  318  212  242
  202  214  324  314  292  244  284   58  334  220  306  290  280   11
   54   34   84  128   86  436  422  438  414  440  126  320  198  302
  370  130  398  210  460  448  396  454  400  442  462  464   82  412
  468  456  330  406  446  424  466  402  132  444  428  458  310  268
  276   72  376  426   74  430  282  312  278  266  390  124  270  264
  618  622  298  304  632  274  262  104  254  620  256  628  614  616
  260  624  272   56  174  176  184  172  178  182  470  180  316   44
  374  170  116  162  158  152  146  142  136  120 1624  250  118 4316
  490  682  150   96  164  154  168  574 5418  160  102 4808   94  378
  144 5494  668  816 4018  166   98  188  140  194 6974   76  246  114
  148 5386  594  156  134  138  512  826  532 1268 4094 7040  644  196
  300  100  954  542 5966 1876  890 2578 1388 2478 1604  584 5404 7292
 9094  186  666 9492  486 4232  598 1062 3086  392  248  702 5708  818
 1634  680  710 1630  706  672  804 2698 1522  252  538  258  630 1664
  488  558  576  832 5370 4026 4694 3758 1254 1832 3974  556 1654  708
 1026 4040  754 1816 4410 1096  192 6680  722  494 1866  566  384  564
 1638 4416 5062 5424  586 8070  864 1138  714 1488 7990  998 5702  740
 2740 2462  792  964 2014 4624 6430  472  776  752  634 3818  686 5666
 4608  704  660  930  742  790  516  410  912 4370 4360 4174 1600 5392
 1248  810 2320  582 5402  552  662 1874 7684  612 9616  514 1072  636
 1142 6836 8882  610 5652 1422  484  608  590 1006 2446 2690  670 4806
 2436  382 4404  716 2922 5382 6262 6616  568 6776  578 1246  592 3074
 1870  530 1704 3844 2228 1110 2004  788  822   21  600 1622 5254 1500
 5442 7226  482 4938 7252  698 1770 7908  694  836 4212 1458 8324  854
 1834 2700 1892  978  696 2734  720 5412 4920  732 4450 1708 1012  524
```

```
 2722  798  784 1596  782  678  952 1228 6118 2066 1686 6584 6332 5444
 2750 4458 7194 5376  734 7660 3882  554 5686 3822  658  760  758  800
 5184 1456 1544 1044]
=======================================
dpkts

[    4    38    16    12     6     8    28     0    10    58    20     2
    24    18    22   126    30    40    68    66    80    42    54    26
    44    60     3   114    82   746    36    14   116   438   388    70
    50    92   858    64    90   560   112  1278  1078    48    72    94
    78    52     1   400    62    86   124    34   806   122    84   754
    88   128   512    74   594   130   682    76   684   510   192   236
   238   234   232    56   196   240   194   190   106   606    46  1716
    32  1452   252  2940   150   140   402   368    96   704   164   100
   274   242   810   288   324   148   350   160   570  1084   166   638
   158   156   748   230   108   318   412   138   268   110  2150   228
   558   828   132   626   340   286   394   182  2588   162  1036   826
   210   332   118   300   366  1530   170   244   208  3402   674   200
   602  1046   448   226  1234   146    98   346   316  1382   296   254
   304   102   562   168   370   152   362   918   616   310   144   364
   276   224   174   802   308   588   392   154   554   272   250   814
   142   104  1106   690   184   342  1368   656   204   120  1414   328
   702   612   636   222   650   280   198   564  2748   178   486   472
   816  1978  4560   808   860   134   136   306  5254  2636   352   260
   334  4526   800   444  1250   416  2272   248   338   426   220   336
  1594   258   348   270  1226  5164   314   694   528   188   548  1542
   446  1434   822   732   592  1418   282   398   652   216 10974   372
   422   292   668   408   374   202   172   556  2786   660   572  1154
  2144   996   356   320   176   246   516   670   818   780   618   278
  1216  1104   386  6676   878   186   214   566   624   180   492   354
   474   382 10850   390  4396  1394   490  1998   976   376   646   862
  2630  2612   312   738  4550   716   428  1560   812  2646  6088   536
  1052   700   696   360   662  1244   798   482   264   586   796   698
   610   824   326  1206   832   298  1458   544  1174  1272   266   550
   206  3350   600   664   418   420  1842   358  1030  4548  1350   468
   856  9346  3698  3664   344  3696   396   450  1632   432   724   552
   380   322   710  1256   290  2620  3036  1796   946   986  1032  4476
   302  1538   256  1048   330   804   576  3206   648   686  1222  1262
   642   568   436  2076  1088   520   762   378   424  1622  1110  1200
  1502   462  9660  1260  1364   514   434   608   864  1500  1310  2014
   466  6494  1644  3604   714  2642   430   692   476  2306  1388  1264
   928  1018   644   688   964   834  1114   596   680  3314   212  1160
   294  4910  5276   792  2004   384  1742  1656   404  2002  6994]
=======================================
sbytes

[   258    734    364 … 272070  69997  12601]
=======================================
```

```
dbytes

[   172  42014  13186 …  68382   3084 426483]
========================================
rate

[74.08749  78.473372 14.170161 … 49.171955 31.468251 33.612649]
========================================
sttl

[252  62 254   0   1  31  63  64  60  29 255]
========================================
dttl

[254 252   0  29  60  31]
========================================
sload

[14158.94238   8395.112305  1572.271851 …  7185.126465 62427.87109
  8826.286133]
========================================
dload

[  8495.365234 503571.3125    60929.23047  … 129476.7813     9586.899414
   4903.492188]
========================================
sloss

[   0    2    1    3   28    5    4    6   11   12   15   30    7   14
   18   26   21   19   38   55   47   27   35   36   34   24  144   13
   33   37   39   17   54    8   29   53  130  132   10   58   32    9
   88   84   85   83   78   76   73   70   68   65   63   57   51   52
   79   20  133   16  809   56 2157  337   69   45  118   22  140   74
   81   43  171   42  131  284   31 2706  113   97   92   49   77   71
  183  167   48 2403   41   62   44   72  157  186  111 2746  331  406
 2003   80   46   91   59   67   60   94  110   23 3484  135   95   50
  120   86 2690  295  151   75   64   40  105   66  129  410  631 2045
 3518  152  102  221  319  168   25  147   87   96  125  108  473  268
  170 2982  934  438  142 1288  691 1236  182  149  799  289 2699  278
 3643 4546   90  177  329 4745  240 2110  210  296  122   61  528  127
  165  124  313  164 1540  143  192  121 2851  154  340  156   89  349
  332 1346  758  123  266  126  107  137  119  315  830  109  216  114
  244  276  103  106  338  285  413 2682 2011 2343   82 1877  624  913
 1985  207  162  275   99  353  512  230 2019  376  100  138  184  905
 2203  548   93  145 3338  358  930  280  189  279  816 2205  181 2530
 2709  288 4033  451  566  354  741 3993  239  211 2848  196  307  366
 1367  348 1225  395  112  476 1004 2309 3213  233  385  116  375  314
 1906 2832  153 2303  327  464  370  394  257  159 2184 2179 2085  217
```

```
  797  146 2693  304  357  622  402 1157 2698  273  325  328  204 3838
  303 4803 2014  532  209  568 3417 4439  302 2822  150  232  241  495
 1220 1341  155 2401 1214  270  101  322  188 2201  355 1458 2688 3129
 3307  175  218  281 3386  294  286  619  339  166  352 1536  932  850
 1922 1111  552  999  391  408  297  808 2623  271  747  334 2718 3607
  238 2468  161 3624  346  881 3951  344  414 2101 4158  115  426  911
 1347  185  305  178  943 1364  356 2703 2457 2225  847  502  261 1358
  213  398  392  795  390  336  287  611 3057 1030  840 3290 3164  219
 2719 1372  399 2226 3592 2687  364 3830 1940 2840 1910  377 2591  726
  770  393  309]
======================================
dloss

[   0   17    6    3    1    8    2    5    4   27   11   10    9   32
    7   13   26   33   15   14   31  370  197  174   28   18   39  390
  253  583  491  180   24   36  366  342  231   30  269  199   20  369
  230   37   59   67   44  274   16   22  856   25  724   21   12 1467
   73  194  181   46  349  182   35   80  134  119  141   23  160  135
   72   45  172   75  283  540   77  317   74  113   52  157  204   34
  115  132   54   29   53 1072  111  276   95  124  168   56  195 1291
   79   78   19  103  164   57   51  148   64  763   83  120  102 1699
  335  221  615   42   71   68   41   47   43  171  155   40  689  146
   55  125  149   49  279  150  131   76   82  183  178  114   63  306
  136  110   84  398  152  292  275   38   69  192   50  343   86  326
  100   58  277  162  304  109  323  138   97   48  153   89 1368   96
  240  406  123  987 2277  402   70   66  151 2627 1315  173  165 2262
  220  623  116  122  206 1133  167  211  108  166  797  126  611 2576
  118  345   61  262   91  272  769  715   81  707  196  133  106 5484
  144   88  201  332   99  344 1390  284  572 1070   62   93  176  158
  179   85  256  407  307  310  137  606  550  191 3338  295   90  324
  105  234  280  121  244  175  203 5425 2198  694  243  997  117 1312
 1303  154 2273  356  212  142  185  778  403 1320 3039  266  163  524
  703  329  620  397  238  130  288  396  347  303  161  601  414  727
  270  139  294  634  273   92  101 1674  330  184  207  921  177 2272
  521 4672 1846 1831 1848  412  814  265  214  188 1316  410  625 1307
 1517  895 2234  298  767  522  399 1601  341  609  302  629  235  319
  282  147  112  216 1035  257  186  210  805  749 4829  680  254  215
  748  653 1005  301 3246  820 1799  143 1318  255  236 1151  692  233
  555  296   98 1654  104  321   87 2454 2633   65 1000  127  190  869
  825  200  281  140  999 3497]
======================================
sinpkt

[ 24.2956    49.915    231.875571 …  52.447526  57.671293  54.400111]
======================================
dinpkt

[  8.375     15.432865 102.737203 …  33.93738   73.69143   66.98057 ]
```

```
========================================
sjit

[   30.177547    61.426934 17179.58686  …  3005.256004  3661.213103
  3721.068786]
========================================
djit

[   11.830604  1387.77833  11420.92623  …  2479.497222   112.41807
   120.177727]
========================================
swin

[255   0  31 232  14 192 103  45  87 172 168 167  42]
========================================
stcpb

[ 621772692 1417884146 2116150707 …    5604755 1932059121 3518776216]
========================================
dtcpb

[2202533631 3077387971 2963114973 …  575257391 2472223109 3453092386]
========================================
dwin

[255   0 244  70  48  37  40]
========================================
tcprtt

[0.       0.111897 0.128381 … 0.044995 0.045137 0.09944 ]
========================================
synack

[0.       0.061458 0.071147 … 0.025596 0.008571 0.036895]
========================================
ackdat

[0.       0.050439 0.057234 … 0.037629 0.036566 0.062545]
========================================
smean

[  43   52   46 … 1440 1344  688]
========================================
dmean

[  43 1106  824 … 1137  603  747]
========================================
trans_depth
```

```
[  0   1   2  80 155   3   5   4 163 172  39]
========================================
response_body_len

[    0   103   109 … 12836 18663  3021]
========================================
ct_srv_src

[ 1 43  7 11  2  3  6  5 12 63  4  9  8 14 13 16 10 17 15 19 21 20 22 18
 27 23 37 46 39 28 36 34 29 40 25 31 38 35 52 33 24 26 44 45 30 32 47 41
 42 49 51 50]
========================================
ct_state_ttl

[0 1 2 3 6]
========================================
ct_dst_ltm

[ 1  2  3  6  7 10  5  8  9 12  4 11 13 14 24 15 17 18 16 21 23 37 46 20
 40 25 22 41 35 45 27 19 26 44 31 32 38 33 39 36 43 48 29 34 47 30 28 42
 50 51]
========================================
ct_src_dport_ltm

[ 1  2  3  6 10  5  8  9  7  4 11 14 13 15 12 21 17 18 23 37 46 16 20 40
 25 19 33 22 31 27 34 28 26 32 29 30 24 36 45 35 39 42 41 38 50 51 43]
========================================
ct_dst_sport_ltm

[ 1  2  3 23 37 46 14 20 40  4  5  6  9  8 10  7 18 17 15 16 13 12 11 24
 25 28 22 19 26 21 27 31]
========================================
ct_dst_src_ltm

[ 1  2  3 40  4  7  6 63  5  8  9 10 13 11 12 14 19 26 15 23 37 46 39 20
 28 36 34 22 29 16 25 32 38 35 51 33 18 31 41 21 30 17 27 44 43 45 24 54
 42 65 47 49 50 52]
========================================
is_ftp_login

[0 1 2 4]
========================================
ct_ftp_cmd

[0 1 2 4]
========================================
ct_flw_http_mthd
```

```
[ 0  1  4  2  9 12  6 25 16 30  3]
========================================
ct_src_ltm

[ 1  2  3  6 12 10  5  8  4  9 13  7 17 19 11 14 15 16 21 18 24 20 23 37
 46 40 25 22 45 27 26 39 41 44 34 30 28 35 29 42 47 43 31 60 33 36 38 32
 50 51]
========================================
ct_srv_dst

[ 1  6 39  3  2  8  5 11  9  7 62 10  4 12 21 17 16 13 18 14 19 15 24 20
 26 23 37 46 28 36 34 22 29 30 40 25 31 38 35 51 33 27 44 42 45 32 41 43
 47 49 50 52]
========================================
is_sm_ips_ports

[0 1]
========================================
attack_cat

['Normal' 'Backdoor' 'Analysis' 'Fuzzers' 'Shellcode' 'Reconnaissance'
 'Exploits' 'DoS' 'Worms' 'Generic']
========================================
label

[0 1]
```

[5]:
```
print("train_csv_raw:\n", train_csv_raw.info())
print("test_csv_raw:\n", test_csv_raw.info())
print("train_csv_raw describe:\n", train_csv_raw.describe())
print("test_csv_raw describe:\n", test_csv_raw.describe())


# All column non-null dropna is not required
# outlier check, preprocessing based on int64, float 64, object type
train_num_cols = train_csv_raw.select_dtypes(include=['int64', 'float64']).
  ↪columns
train_obj_cols = train_csv_raw.select_dtypes(include=['object']).columns
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 175341 entries, 0 to 175340
Data columns (total 45 columns):
 #   Column             Non-Null Count   Dtype
---  ------             --------------   -----
 0   id                 175341 non-null  int64
 1   dur                175341 non-null  float64
 2   proto              175341 non-null  object
```

```
 3   service            175341 non-null  object
 4   state              175341 non-null  object
 5   spkts              175341 non-null  int64
 6   dpkts              175341 non-null  int64
 7   sbytes             175341 non-null  int64
 8   dbytes             175341 non-null  int64
 9   rate               175341 non-null  float64
 10  sttl               175341 non-null  int64
 11  dttl               175341 non-null  int64
 12  sload              175341 non-null  float64
 13  dload              175341 non-null  float64
 14  sloss              175341 non-null  int64
 15  dloss              175341 non-null  int64
 16  sinpkt             175341 non-null  float64
 17  dinpkt             175341 non-null  float64
 18  sjit               175341 non-null  float64
 19  djit               175341 non-null  float64
 20  swin               175341 non-null  int64
 21  stcpb              175341 non-null  int64
 22  dtcpb              175341 non-null  int64
 23  dwin               175341 non-null  int64
 24  tcprtt             175341 non-null  float64
 25  synack             175341 non-null  float64
 26  ackdat             175341 non-null  float64
 27  smean              175341 non-null  int64
 28  dmean              175341 non-null  int64
 29  trans_depth        175341 non-null  int64
 30  response_body_len  175341 non-null  int64
 31  ct_srv_src         175341 non-null  int64
 32  ct_state_ttl       175341 non-null  int64
 33  ct_dst_ltm         175341 non-null  int64
 34  ct_src_dport_ltm   175341 non-null  int64
 35  ct_dst_sport_ltm   175341 non-null  int64
 36  ct_dst_src_ltm     175341 non-null  int64
 37  is_ftp_login       175341 non-null  int64
 38  ct_ftp_cmd         175341 non-null  int64
 39  ct_flw_http_mthd   175341 non-null  int64
 40  ct_src_ltm         175341 non-null  int64
 41  ct_srv_dst         175341 non-null  int64
 42  is_sm_ips_ports    175341 non-null  int64
 43  attack_cat         175341 non-null  object
 44  label              175341 non-null  int64
dtypes: float64(11), int64(30), object(4)
memory usage: 60.2+ MB
train_csv_raw:
 None
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 82332 entries, 0 to 82331
```

```
Data columns (total 45 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   id                  82332 non-null  int64
 1   dur                 82332 non-null  float64
 2   proto               82332 non-null  object
 3   service             82332 non-null  object
 4   state               82332 non-null  object
 5   spkts               82332 non-null  int64
 6   dpkts               82332 non-null  int64
 7   sbytes              82332 non-null  int64
 8   dbytes              82332 non-null  int64
 9   rate                82332 non-null  float64
 10  sttl                82332 non-null  int64
 11  dttl                82332 non-null  int64
 12  sload               82332 non-null  float64
 13  dload               82332 non-null  float64
 14  sloss               82332 non-null  int64
 15  dloss               82332 non-null  int64
 16  sinpkt              82332 non-null  float64
 17  dinpkt              82332 non-null  float64
 18  sjit                82332 non-null  float64
 19  djit                82332 non-null  float64
 20  swin                82332 non-null  int64
 21  stcpb               82332 non-null  int64
 22  dtcpb               82332 non-null  int64
 23  dwin                82332 non-null  int64
 24  tcprtt              82332 non-null  float64
 25  synack              82332 non-null  float64
 26  ackdat              82332 non-null  float64
 27  smean               82332 non-null  int64
 28  dmean               82332 non-null  int64
 29  trans_depth         82332 non-null  int64
 30  response_body_len   82332 non-null  int64
 31  ct_srv_src          82332 non-null  int64
 32  ct_state_ttl        82332 non-null  int64
 33  ct_dst_ltm          82332 non-null  int64
 34  ct_src_dport_ltm    82332 non-null  int64
 35  ct_dst_sport_ltm    82332 non-null  int64
 36  ct_dst_src_ltm      82332 non-null  int64
 37  is_ftp_login        82332 non-null  int64
 38  ct_ftp_cmd          82332 non-null  int64
 39  ct_flw_http_mthd    82332 non-null  int64
 40  ct_src_ltm          82332 non-null  int64
 41  ct_srv_dst          82332 non-null  int64
 42  is_sm_ips_ports     82332 non-null  int64
 43  attack_cat          82332 non-null  object
 44  label               82332 non-null  int64
```

```
dtypes: float64(11), int64(30), object(4)
memory usage: 28.3+ MB
test_csv_raw:
 None
train_csv_raw describe:
                  id            dur          spkts          dpkts  \
count  175341.000000  175341.000000  175341.000000  175341.000000
mean    87671.000000       1.359389      20.298664      18.969591
std     50616.731112       6.480249     136.887597     110.258271
min         1.000000       0.000000       1.000000       0.000000
25%     43836.000000       0.000008       2.000000       0.000000
50%     87671.000000       0.001582       2.000000       2.000000
75%    131506.000000       0.668069      12.000000      10.000000
max    175341.000000      59.999989    9616.000000   10974.000000

             sbytes        dbytes          rate           sttl           dttl  \
count  1.753410e+05  1.753410e+05  1.753410e+05  175341.000000  175341.000000
mean   8.844844e+03  1.492892e+04  9.540619e+04     179.546997      79.609567
std    1.747656e+05  1.436542e+05  1.654010e+05     102.940011     110.506863
min    2.800000e+01  0.000000e+00  0.000000e+00       0.000000       0.000000
25%    1.140000e+02  0.000000e+00  3.278614e+01      62.000000       0.000000
50%    4.300000e+02  1.640000e+02  3.225807e+03     254.000000      29.000000
75%    1.418000e+03  1.102000e+03  1.250000e+05     254.000000     252.000000
max    1.296523e+07  1.465555e+07  1.000000e+06     255.000000     254.000000

              sload  …  ct_src_dport_ltm  ct_dst_sport_ltm  ct_dst_src_ltm  \
count  1.753410e+05  …     175341.000000     175341.000000   175341.000000
mean   7.345403e+07  …          5.383538          4.206255        8.729881
std    1.883574e+08  …          8.047104          5.783585       10.956186
min    0.000000e+00  …          1.000000          1.000000        1.000000
25%    1.305334e+04  …          1.000000          1.000000        1.000000
50%    8.796748e+05  …          1.000000          1.000000        3.000000
75%    8.888889e+07  …          5.000000          3.000000       12.000000
max    5.988000e+09  …         51.000000         46.000000       65.000000

        is_ftp_login     ct_ftp_cmd  ct_flw_http_mthd     ct_src_ltm  \
count  175341.000000  175341.000000     175341.000000  175341.000000
mean        0.014948       0.014948          0.133066       6.955789
std         0.126048       0.126048          0.701208       8.321493
min         0.000000       0.000000          0.000000       1.000000
25%         0.000000       0.000000          0.000000       2.000000
50%         0.000000       0.000000          0.000000       3.000000
75%         0.000000       0.000000          0.000000       9.000000
max         4.000000       4.000000         30.000000      60.000000

          ct_srv_dst  is_sm_ips_ports          label
count  175341.000000    175341.000000  175341.000000
mean        9.100758         0.015752       0.680622
```

```
std        10.756952         0.124516       0.466237
min         1.000000         0.000000       0.000000
25%         2.000000         0.000000       0.000000
50%         4.000000         0.000000       1.000000
75%        12.000000         0.000000       1.000000
max        62.000000         1.000000       1.000000

[8 rows x 41 columns]
test_csv_raw describe:
                 id           dur         spkts         dpkts        sbytes  \
count  82332.000000  82332.000000  82332.000000  82332.000000  8.233200e+04
mean   41166.500000      1.006756     18.666472     17.545936  7.993908e+03
std    23767.345519      4.710444    133.916353    115.574086  1.716423e+05
min        1.000000      0.000000      1.000000      0.000000  2.400000e+01
25%    20583.750000      0.000008      2.000000      0.000000  1.140000e+02
50%    41166.500000      0.014138      6.000000      2.000000  5.340000e+02
75%    61749.250000      0.719360     12.000000     10.000000  1.280000e+03
max    82332.000000     59.999989  10646.000000  11018.000000  1.435577e+07

             dbytes          rate          sttl          dttl         sload  \
count  8.233200e+04  8.233200e+04  82332.000000  82332.000000  8.233200e+04
mean   1.323379e+04  8.241089e+04    180.967667     95.713003  6.454902e+07
std    1.514715e+05  1.486204e+05    101.513358    116.667722  1.798618e+08
min    0.000000e+00  0.000000e+00      0.000000      0.000000  0.000000e+00
25%    0.000000e+00  2.860611e+01     62.000000      0.000000  1.120247e+04
50%    1.780000e+02  2.650177e+03    254.000000     29.000000  5.770032e+05
75%    9.560000e+02  1.111111e+05    254.000000    252.000000  6.514286e+07
max    1.465753e+07  1.000000e+06    255.000000    253.000000  5.268000e+09

       …  ct_src_dport_ltm  ct_dst_sport_ltm  ct_dst_src_ltm  is_ftp_login  \
count  …      82332.000000      82332.000000    82332.000000  82332.000000
mean   …          4.928898          3.663011        7.456360      0.008284
std    …          8.389545          5.915386       11.415191      0.091171
min    …          1.000000          1.000000        1.000000      0.000000
25%    …          1.000000          1.000000        1.000000      0.000000
50%    …          1.000000          1.000000        3.000000      0.000000
75%    …          4.000000          3.000000        6.000000      0.000000
max    …         59.000000         38.000000       63.000000      2.000000

         ct_ftp_cmd  ct_flw_http_mthd    ct_src_ltm    ct_srv_dst  \
count  82332.000000      82332.000000  82332.000000  82332.000000
mean       0.008381          0.129743      6.468360      9.164262
std        0.092485          0.638683      8.543927     11.121413
min        0.000000          0.000000      1.000000      1.000000
25%        0.000000          0.000000      1.000000      2.000000
50%        0.000000          0.000000      3.000000      5.000000
75%        0.000000          0.000000      7.000000     11.000000
max        2.000000         16.000000     60.000000     62.000000
```

```
       is_sm_ips_ports         label
count     82332.000000  82332.000000
mean          0.011126      0.550600
std           0.104891      0.497436
min           0.000000      0.000000
25%           0.000000      0.000000
50%           0.000000      1.000000
75%           0.000000      1.000000
max           1.000000      1.000000

[8 rows x 41 columns]
```

2.0.1 Added New Columns

- pkt_ratio = (spkts + 1)/(dpkts + 1)

packet ratio is balanced when send packets(spkts) and destination packets(dpkts) are mostly similar (Normal). However the attack typically shows imbalanced spkts and dpkts spkts » dpkts → Many requests, no responds (DoS, Port scan)

spkts dpkts → totally normal bi-directional communication

dpkts » spkts → server responds in huge file (Download, Streaming)

- ttl_gap = abs(sttl - dttl)

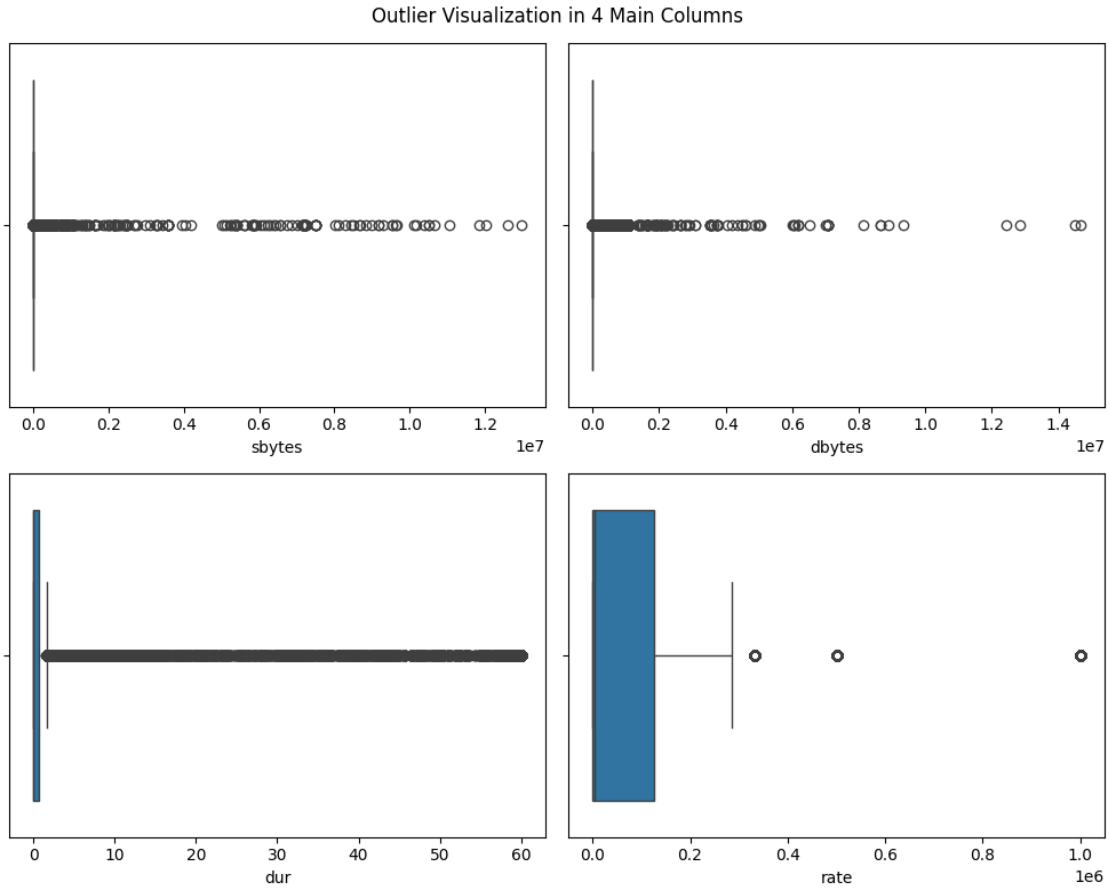
-> Bot Net/ Spoofing have greater ttl difference

-> Normal -> relatively even

3 Visualization

- plot
- Outlier display (But don't remove for model training)

[6]:
```
fig, axes = plt.subplots(2, 2, figsize=(10,8))
fig.suptitle('Outlier Visualization in 4 Main Columns')
sns.boxplot(x=train_csv_raw['sbytes'], ax=axes[0,0])
sns.boxplot(x=train_csv_raw['dbytes'], ax=axes[0,1])
sns.boxplot(x=train_csv_raw['dur'], ax=axes[1,0])
sns.boxplot(x=train_csv_raw['rate'], ax=axes[1,1])
plt.savefig("EDA_images/task1_outlier_visualization.png", dpi=300)

plt.tight_layout()
plt.show()
```



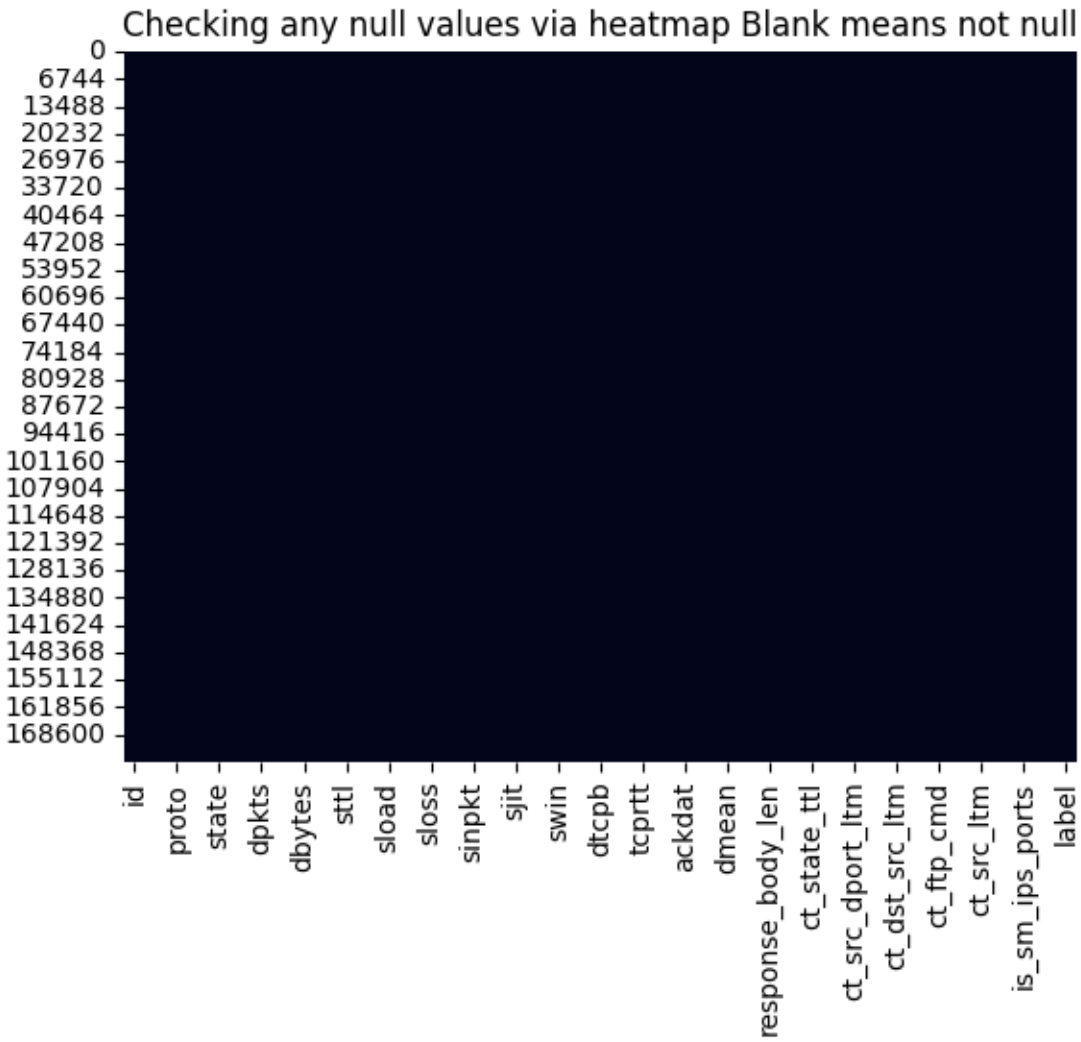
[7]:
```
print("="*40)
print("Testing Graphs")

import numpy as np

sns.heatmap(train_csv_raw.isnull(), cbar=False)
plt.title("Checking any null values via heatmap Blank means not null")
plt.savefig("EDA_images/task1_null_value_check.png", dpi=300)

plt.show()

# Distribution & Outliers in major columns (log scaling if necessary)
fit, axes = plt.subplots(3, 2, figsize=(10,8))
sns.histplot(np.log1p(train_csv_raw['spkts']), bins=50, ax=axes[0,0], kde=True)␣
 ↪#log scaled since the values are highly left-skewed
axes[0,0].set_title('Distribution of spkts')
axes[0,0].set_xlabel("LOG SCALED spkts (log(spkts + 1))")

sns.histplot(np.log1p(train_csv_raw['sbytes']), bins=50, ax=axes[0,1], kde=True)
```

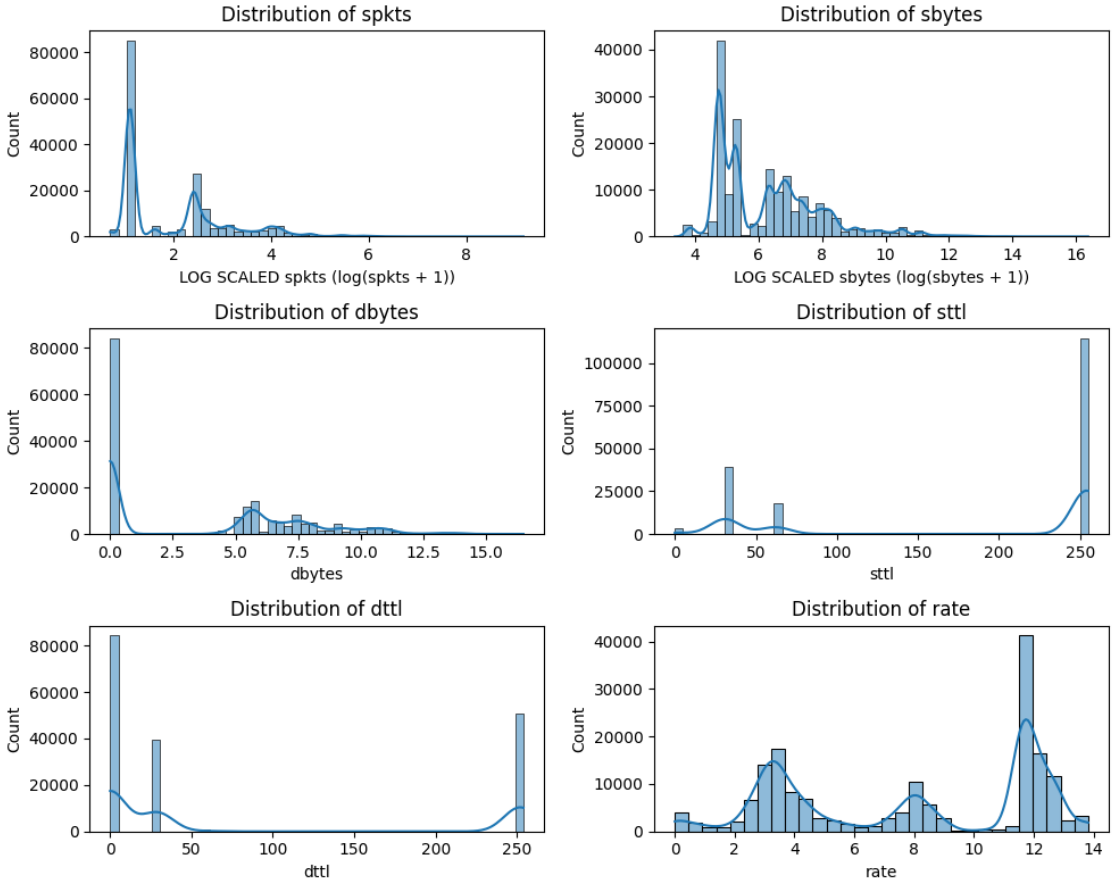
```
axes[0,1].set_title('Distribution of sbytes')
axes[0,1].set_xlabel("LOG SCALED sbytes (log(sbytes + 1))")

sns.histplot(np.log1p(train_csv_raw['dbytes']), bins=50, ax=axes[1,0], kde=True)
axes[1,0].set_title('Distribution of dbytes')
axes[1,0].set_xlabel("dbytes")

sns.histplot(train_csv_raw['sttl'], bins=50, ax=axes[1,1], kde=True)
axes[1,1].set_title('Distribution of sttl')
axes[1,1].set_xlabel("sttl")

sns.histplot(train_csv_raw['dttl'], bins=50, ax=axes[2,0], kde=True)
axes[2,0].set_title('Distribution of dttl')
axes[2,0].set_xlabel("dttl")

sns.histplot(np.log1p(train_csv_raw['rate']), bins=30, ax=axes[2,1], kde=True)
axes[2,1].set_title('Distribution of rate')
axes[2,1].set_xlabel("rate")

plt.savefig("EDA_images/task1_six_cols_distribution.png", dpi=300)

plt.tight_layout()
plt.show()

print("="*40)


fig, axes = plt.subplots(2, 2, figsize=(10,8))
fig.suptitle('Checking outlieres')

sns.countplot(x='label', data=train_csv_raw, ax=axes[0,0])
axes[0,0].set_title('Count on \'Labels\'')

#sbyte is the data send
sns.histplot(np.log1p(train_csv_raw['sbytes']), bins=50, ax=axes[0,1], kde=True)
axes[0,1].set_title('sbytes Distribution')

## checking outliers (all numeric columns)
selected_num_cols = ['spkts', 'sbytes', 'dbytes', 'sttl', 'dttl', 'rate']
np.log1p(train_csv_raw[selected_num_cols]).boxplot(ax=axes[1,0])
axes[1,0].set_title('Checking Outliers in major numerical columns')
axes[1,0].tick_params(axis='x', rotation=90)

import matplotlib.pyplot as plt
# Reducing the number of sample for faster view
sample_df = train_csv_raw.sample(20000, random_state=42)
```

```
sns.heatmap(sample_df.corr(numeric_only=True), ax=axes[1,1])
axes[1,1].set_title('HeatMap with 20000 samples')

plt.savefig("EDA_images/task1_Outlier_heatmap.png", dpi=300)


plt.tight_layout()
plt.show()
```

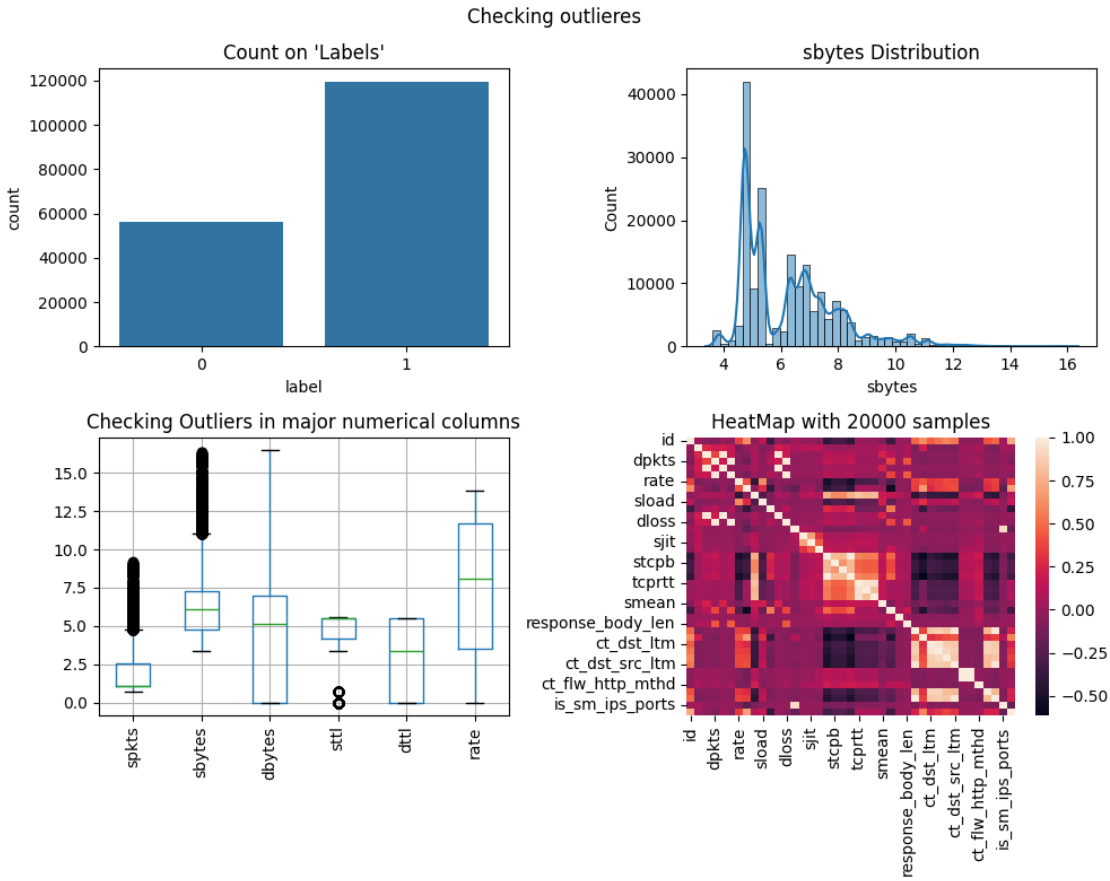
==========

Testing Graphs




Distribution of spkts
Count
LOG SCALED spkts (log(spkts + 1))
Distribution of sbytes
Count
LOG SCALED sbytes (log(sbytes + 1))
Distribution of dbytes
Count
dbytes
Distribution of sttl
Count
sttl
Distribution of dttl
Count
dttl
Distribution of rate
Count
rate

==========



3.0.1 Conclusion

Outliers may exist in sload, stcpp, dtcpp, implying unusual activity.

3.1 Define Customer transformer

[8]:
```
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.base import BaseEstimator, TransformerMixin
from sklearn.model_selection import train_test_split
from tqdm.auto import tqdm

# using tqdm status printer
tqdm.pandas()

class CustomFeatureTransformer(BaseEstimator, TransformerMixin):
    """
    Class used to encode data into model trainable format.
    Based on the model it can distinguish catetorical and numerical columns and
    Conducts proper encoding.
    + also it can do outliers/missing values/ noise(pollishing) detection.
```

```
    """
    def __init__(self, use_extra_features=True):
        self.use_extra_features = use_extra_features

    def fit(self, X, y=None):
        return self

    # Juwon Custom
    def cat_num_encoding(self, X_train, X_test, needCat=True) -> pd.DataFrame:
        # Distinguish categorical numerical columns
        cat_cols = X_train.select_dtypes(include=['object', 'category']).
↪columns.tolist()
        num_cols = X_train.select_dtypes(include=['int64', 'float64']).columns.
↪tolist()

        # =========================
        # Print Summary
        # =========================
        print("\n" + "=" * 60)
        print("Encoding & Scaling Summary")
        print("=" * 60)

        print(f"Train shape : {X_train.shape}")
        print(f"Test  shape : {X_test.shape}\n")

        print(f"Numerical columns ({len(num_cols)}):")
        print("  - " + ", ".join(num_cols) if num_cols else "  - None")

        print(f"\nCategorical columns ({len(cat_cols)}):")
        print("  - " + ", ".join(cat_cols) if cat_cols else "  - None")

        print("=" * 60)

        # =========================
        # Numerical Scaling
        # =========================
        scaler = StandardScaler()

        if num_cols:
            X_train[num_cols] = scaler.fit_transform(X_train[num_cols])
            X_test[num_cols] = scaler.transform(X_test[num_cols])
            print("\nNumerical Scaling Completed\n")
            print("StandardScaler applied to numerical columns")
            print("=" * 60)

        # =========================
        # Categorical Label Encoding
```

```
        # ========================
        encoders = {}
        if needCat:
          print("\nCategorical Label Encoding Started\n")
          for col in tqdm(cat_cols, desc="Label Encoding"):
              le = LabelEncoder()

              X_train[col] = le.fit_transform(X_train[col].astype(str))

              X_test[col] = X_test[col].astype(str).map(
                  lambda x: le.transform([x])[0] if x in le.classes_ else -1
              )

              encoders[col] = le

              # print output
              print(f"Encoded column: {col}")
              print(f"  Classes: {list(le.classes_)}")
          print("\n\n**LabelEncoder** applied to categorical columns")
          print("=" * 60)

          print("\nEncoding complete")
          print("=" * 60)
        else:
          print("\nCategorical columns removed")
          print("=" * 60)

        return X_train, X_test, encoders


    # Transform X(train or test) and add new features
    def transform(self, X):
        X_transformed = X.copy()
        if self.use_extra_features:
            # --- JUWON'S LOGIC GOES HERE LATER ---
            # For now, we create a dummy feature so the logic holds
            # Example: Interaction between Duration and Source Bytes
            if 'spkts' in X.columns and 'dpkts' in X.columns:
                X_transformed['pkt_ratio'] = (X_transformed['spkts'] + 1)/
↪(X_transformed['dpkts'] + 1)
            if 'sttl' in X.columns and 'dttl' in X.columns:
              X_transformed['ttl_gap'] = abs(X_transformed['sttl'] -␣
↪X_transformed['dttl'])
        return X_transformed
```

```
# Generate new_columns
transformer = CustomFeatureTransformer()
# New features added (pkt_ratio, ttl_gap)
train_transformed = transformer.transform(train_csv_raw)
test_transformed = transformer.transform(test_csv_raw)

print(train_transformed.head())


# cat_num_encoding# Unpack the validation set
# X_train, y_train, X_val, y_val, X_test, y_test already defined on top␣
 ↪(LabelEncoded)
```

```
   id       dur proto service state  spkts  dpkts  sbytes  dbytes       rate  \
0   1  0.121478   tcp       -   FIN      6      4     258     172  74.087490
1   2  0.649902   tcp       -   FIN     14     38     734   42014  78.473372
2   3  1.623129   tcp       -   FIN      8     16     364   13186  14.170161
3   4  1.681642   tcp     ftp   FIN     12     12     628     770  13.677108
4   5  0.449454   tcp       -   FIN     10      6     534     268  33.373826

   …  is_ftp_login  ct_ftp_cmd  ct_flw_http_mthd  ct_src_ltm  ct_srv_dst  \
0  …             0           0                 0           1           1
1  …             0           0                 0           1           6
2  …             0           0                 0           2           6
3  …             1           1                 0           2           1
4  …             0           0                 0           2          39

   is_sm_ips_ports  attack_cat  label  pkt_ratio  ttl_gap
0                0      Normal      0   1.400000        2
1                0      Normal      0   0.384615      190
2                0      Normal      0   0.529412      190
3                0      Normal      0   1.000000      190
4                0      Normal      0   1.571429        2

[5 rows x 47 columns]
```

```
/Users/jju/Documents/SIM/Semester 1, 2026/CSCI316 Big Data
Mining/Assignments/.venv/lib/python3.9/site-packages/tqdm/auto.py:21:
TqdmWarning: IProgress not found. Please update jupyter and ipywidgets. See
https://ipywidgets.readthedocs.io/en/stable/user_install.html
  from .autonotebook import tqdm as notebook_tqdm
```

4 Feature engineering & Data preprocessing

- split train/test (Already done)
- StandardScaler(K-NN), Random Forest(LabelEncoding(only categorical))
- num of features

[9]:
```
# Remove less important columns (Feature Engineering)
"""
Considered Important
> dur, sbytes, dbytes, rate,
spkts, dpkts,
sttl, dttl,
sinpkt, dinpkt,
sjit, djit,
ct_srv_src, ct_srv_dst, ct_dst_src_ltm
proto/service/state (label-encoding)
is_ftp_login, ct_ftp_cmd


Less Important
- id, label, attack_cat(NEED to DELETE in X)
"""
def report_feature_dropping(
    X_train,
    X_test,
    drop_cols,
    title="Feature Dropping Report (Just for Visualization Purpose, Already␣
 ↪done before)"
):
    print("\n" + "=" * 80)
    print(title)
    print("=" * 80)

    # ---------- BEFORE ----------
    print("\n[BEFORE DROP, New Features already added]")
    print(f"Train shape : {X_train.shape}")
    print(f"Test  shape : {X_test.shape}")
    print(f"Train columns ({len(X_train.columns)}):")
    print(list(X_train.columns))

    # ---------- DROP ----------
    filtered_X_train = X_train.drop(drop_cols, axis=1, errors="ignore")
    filtered_X_test  = X_test.drop(drop_cols, axis=1, errors="ignore")

    dropped_cols = set(X_train.columns) - set(filtered_X_train.columns)

    # ---------- ALIGN ----------
    filtered_X_train, filtered_X_test = filtered_X_train.align(
        filtered_X_test, join="left", axis=1, fill_value=0
    )

    # ---------- AFTER ----------
    print("\n" + "-" * 80)
```

```
    print("[AFTER DROP & ALIGN]")
    print(f"Train shape : {filtered_X_train.shape}")
    print(f"Test  shape : {filtered_X_test.shape}")
    print(f"Remaining columns ({len(filtered_X_train.columns)}):")
    print(list(filtered_X_train.columns))

    # ---------- SUMMARY ----------
    print("\n" + "-" * 80)
    print("[SUMMARY]")
    print(f"Dropped columns ({len(dropped_cols)}): {sorted(dropped_cols)}")
    print("Train/Test columns aligned  ")
    print("=" * 80)

    return filtered_X_train, filtered_X_test

drop_cols = ['label', 'id', 'attack_cat']

# Drop unnecessary columns
filtered_X_train = X_train.copy()
filtered_X_test = X_test.copy()

# Matching train and test columns
filtered_X_train, filtered_X_test = report_feature_dropping(filtered_X_train,␣
 ↪filtered_X_test, drop_cols)

# Save column names
X_train_cols = X_train.columns
filtered_train_cols = filtered_X_train.columns
```

```
================================================================================
Feature Dropping Report (Just for Visualization Purpose, Already done before)
================================================================================

[BEFORE DROP, New Features already added]
Train shape : (180371, 42)
Test  shape : (38651, 42)
Train columns (42):
['dur', 'proto', 'service', 'state', 'spkts', 'dpkts', 'sbytes', 'dbytes',
'rate', 'sttl', 'dttl', 'sload', 'dload', 'sloss', 'dloss', 'sinpkt', 'dinpkt',
'sjit', 'djit', 'swin', 'stcpb', 'dtcpb', 'dwin', 'tcprtt', 'synack', 'ackdat',
'smean', 'dmean', 'trans_depth', 'response_body_len', 'ct_srv_src',
'ct_state_ttl', 'ct_dst_ltm', 'ct_src_dport_ltm', 'ct_dst_sport_ltm',
'ct_dst_src_ltm', 'is_ftp_login', 'ct_ftp_cmd', 'ct_flw_http_mthd',
'ct_src_ltm', 'ct_srv_dst', 'is_sm_ips_ports']

--------------------------------------------------------------------------------
[AFTER DROP & ALIGN]
```

```
Train shape : (180371, 42)
Test  shape : (38651, 42)
Remaining columns (42):
['dur', 'proto', 'service', 'state', 'spkts', 'dpkts', 'sbytes', 'dbytes',
'rate', 'sttl', 'dttl', 'sload', 'dload', 'sloss', 'dloss', 'sinpkt', 'dinpkt',
'sjit', 'djit', 'swin', 'stcpb', 'dtcpb', 'dwin', 'tcprtt', 'synack', 'ackdat',
'smean', 'dmean', 'trans_depth', 'response_body_len', 'ct_srv_src',
'ct_state_ttl', 'ct_dst_ltm', 'ct_src_dport_ltm', 'ct_dst_sport_ltm',
'ct_dst_src_ltm', 'is_ftp_login', 'ct_ftp_cmd', 'ct_flw_http_mthd',
'ct_src_ltm', 'ct_srv_dst', 'is_sm_ips_ports']

--------------------------------------------------------------------------------
[SUMMARY]
Dropped columns (0): []
Train/Test columns aligned
================================================================================
```

[10]:
```
# Print the values
print('\nX_train')
print(X_train[:10])
print('\nX_test')
print(X_test[:10])

print("First 5 feature means:")
print(X_train.mean(axis=0)[:5])

print("\nFirst 5 feature std:")
print(X_train.std(axis=0)[:5])

print('\nX_train dtype: ', type(X_train))
print('\nColumn length: ', len(X_train_cols))
print('\nTrain X shape: ', X_train.shape)
print('\nTest X shape: ', X_test.shape)




# Back to DataFrame with Column Name
X_train = pd.DataFrame(X_train, columns=X_train_cols)
X_test = pd.DataFrame(X_test, columns=X_train_cols)
```

```
X_train
              dur  proto  service  state  spkts  dpkts  sbytes  dbytes  \
102341   0.117938    112        0      2      6      2     986      86
99037   19.888327     78        0      5     20      0     900       0
241927   0.438336    112        0      4     10      6     642     268
129056   0.202640    112        5      4     10      6     790     268
```

```
67302    0.927228    112        0      4     12      8    1010     724
17559    0.001041    118        2      2      2      2     146     178
8654    50.004379     78        0      7      6      0     384       0
202925   0.009831    112        0      4     36     38    2334   16290
14843    2.456484    112        3      4     14     12     820     682
240134   2.264078    112        0      4     24     24    1264    1724

                rate  sttl  …  ct_dst_ltm  ct_src_dport_ltm  \
102341     59.353221    62  …           6                 6
99037       0.955334   254  …           1                 1
241927     34.220323   254  …           2                 1
129056     74.022899   254  …           1                 1
67302      20.491185   254  …           1                 1
17559    2881.844351    31  …           2                 1
8654        0.099991     1  …           2                 1
202925   7425.490704    31  …           3                 1
14843      10.177147   254  …           1                 1
240134     20.759003    62  …           2                 1

        ct_dst_sport_ltm  ct_dst_src_ltm  is_ftp_login  ct_ftp_cmd  \
102341                 1               8             0           0
99037                  1               1             0           0
241927                 1               3             0           0
129056                 1               1             0           0
67302                  1               1             0           0
17559                  1               2             0           0
8654                   2               1             0           0
202925                 1               1             0           0
14843                  1               1             0           0
240134                 1               2             0           0

        ct_flw_http_mthd  ct_src_ltm  ct_srv_dst  is_sm_ips_ports
102341                 0           6           8                0
99037                  0           1           1                0
241927                 0           2           5                0
129056                 1           1           1                0
67302                  0           1           1                0
17559                  0           2           4                0
8654                   0           1           2                0
202925                 0           4          16                0
14843                  0           2           1                0
240134                 0           2           3                0

[10 rows x 42 columns]

X_test
             dur  proto  service  state  spkts  dpkts  sbytes  dbytes  \
19937   0.754462    112       11      3    126    124   12648   13202
```

```
214767  0.456132    112        9      3     52     40   37228    3276
72073   0.000006     44        0      4      2      0     200       0
238372  3.074767    112        0      3     82    452    4326  610033
119651  0.000005    119        0      4      2      0     200       0
29347   0.001033    118        2      1      2      2     146     178
129745  0.000001    118        2      4      2      0     114       0
217168  0.000000      6        0      4      1      0      46       0
248610  1.602756    112        0      3     10      8     818     354
54770   0.000001     72        0      4      2      0     200       0

                rate  sttl  …  ct_dst_ltm  ct_src_dport_ltm  \
19937   3.300365e+02    31  …           2                 1
214767  1.995037e+02    31  …           4                 1
72073   1.666667e+05   254  …           1                 1
238372  1.733465e+02    62  …           1                 1
119651  2.000000e+05   254  …           2                 2
29347   2.904163e+03    31  …           2                 1
129745  1.000000e+06   254  …          17                16
217168  0.000000e+00     0  …           1                 1
248610  1.060673e+01   254  …           2                 2
54770   1.000000e+06   254  …           1                 1

        ct_dst_sport_ltm  ct_dst_src_ltm  is_ftp_login  ct_ftp_cmd  \
19937                  1               2             0           0
214767                 1               1             0           0
72073                  1               8             0           0
238372                 1               1             0           0
119651                 2               4             0           0
29347                  1               1             0           0
129745                16              33             0           0
217168                 1               1             0           0
248610                 2               2             0           0
54770                  1               6             0           0

        ct_flw_http_mthd  ct_src_ltm  ct_srv_dst  is_sm_ips_ports
19937                  0           3           1                0
214767                 0           1           1                0
72073                  0           2           8                0
238372                 0           8           1                0
119651                 0           2           4                0
29347                  0           1           3                0
129745                 0          16          33                0
217168                 0           1           1                1
248610                 0           3           3                0
54770                  0           1           6                0

[10 rows x 42 columns]
First 5 feature means:
```

```
dur          1.241866
proto      109.154337
service      1.558715
state        4.340737
spkts       20.096956
dtype: float64

First 5 feature std:
dur          5.950455
proto       21.205595
service      2.248903
state        0.886456
spkts      140.978561
dtype: float64

X_train dtype:  <class 'pandas.core.frame.DataFrame'>

Column length:  42

Train X shape:  (180371, 42)

Test X shape:  (38651, 42)
```