Interactive Lab for Data Analysis using Pandas

Submitted by: Eugene D.G. Dela Cruz

Submitted to: Engr. Roman Richard

Submitted on: 4/3/24

Section: CPE22S3

```
pip install ucimlrepo
```

```
Requirement already satisfied: ucimlrepo in /usr/local/lib/python3.10/dist-packages (0.0.6)
```

```
from ucimlrepo import fetch_ucirepo

# fetch dataset
rt_iot2022 = fetch_ucirepo(id=942)

# data (as pandas dataframes)
X = rt_iot2022.data.features
y = rt_iot2022.data.targets

# metadata
print(rt_iot2022.metadata)

# variable information
print(rt_iot2022.variables)
```

```
{'uci_id': 942, 'name': 'RT-IoT2022 ', 'repository_url': 'https://archive.ics.uci.edu/dataset/942/rt-iot2022', 'data_url': 'https://arch
                    name     role         type demographic description units  \
0              id.orig_p  Feature      Integer        None        None  None
1              id.resp_p  Feature      Integer        None        None  None
2                  proto  Feature  Categorical        None        None  None
3                service  Feature   Continuous        None        None  None
4          flow_duration  Feature   Continuous        None        None  None
..                   ...      ...          ...         ...         ...   ...
80  fwd_init_window_size  Feature      Integer        None        None  None
81  bwd_init_window_size  Feature      Integer        None        None  None
82  fwd_last_window_size  Feature      Integer        None        None  None
83           Attack_type   Target  Categorical        None        None  None
84                    id       ID      Integer        None        None  None

   missing_values
0              no
1              no
2              no
3              no
4              no
..            ...
80             no
81             no
82             no
83             no
84             no

[85 rows x 7 columns]
```

```
import pandas as pd
import numpy as np
```

```
X
```

	id.orig_p	id.resp_p	proto	service	flow_duration	fwd_pkts_tot	bwd_pkts_tot	fwd_data_pkts_tot	bwd_data_pkts_tot	fwd_pkts_
0	38667	1883	tcp	mqtt	32.011598	9	5	3	3	0
1	51143	1883	tcp	mqtt	31.883584	9	5	3	3	0
2	44761	1883	tcp	mqtt	32.124053	9	5	3	3	0
3	60893	1883	tcp	mqtt	31.961063	9	5	3	3	0
4	51087	1883	tcp	mqtt	31.902362	9	5	3	3	0
...	...	...	...	...	...	...	...	...	...	
123112	59247	63331	tcp	-	0.000006	1	1	0	0	167772
123113	59247	64623	tcp	-	0.000007	1	1	0	0	144631
123114	59247	64680	tcp	-	0.000006	1	1	0	0	167772
123115	59247	65000	tcp	-	0.000006	1	1	0	0	167772
123116	59247	65129	tcp	-	0.000006	1	1	0	0	167772

123117 rows × 83 columns

y

	Attack_type
0	MQTT_Publish
1	MQTT_Publish
2	MQTT_Publish
3	MQTT_Publish
4	MQTT_Publish
...	...
123112	NMAP_XMAS_TREE_SCAN
123113	NMAP_XMAS_TREE_SCAN
123114	NMAP_XMAS_TREE_SCAN
123115	NMAP_XMAS_TREE_SCAN
123116	NMAP_XMAS_TREE_SCAN

123117 rows × 1 columns

X.columns

```
Index(['id.orig_p', 'id.resp_p', 'proto', 'service', 'flow_duration',
       'fwd_pkts_tot', 'bwd_pkts_tot', 'fwd_data_pkts_tot',
       'bwd_data_pkts_tot', 'fwd_pkts_per_sec', 'bwd_pkts_per_sec',
       'flow_pkts_per_sec', 'down_up_ratio', 'fwd_header_size_tot',
       'fwd_header_size_min', 'fwd_header_size_max', 'bwd_header_size_tot',
       'bwd_header_size_min', 'bwd_header_size_max', 'flow_FIN_flag_count',
       'flow_SYN_flag_count', 'flow_RST_flag_count', 'fwd_PSH_flag_count',
       'bwd_PSH_flag_count', 'flow_ACK_flag_count', 'fwd_URG_flag_count',
       'bwd_URG_flag_count', 'flow_CWR_flag_count', 'flow_ECE_flag_count',
       'fwd_pkts_payload.min', 'fwd_pkts_payload.max', 'fwd_pkts_payload.tot',
       'fwd_pkts_payload.avg', 'fwd_pkts_payload.std', 'bwd_pkts_payload.min',
       'bwd_pkts_payload.max', 'bwd_pkts_payload.tot', 'bwd_pkts_payload.avg',
       'bwd_pkts_payload.std', 'flow_pkts_payload.min',
       'flow_pkts_payload.max', 'flow_pkts_payload.tot',
       'flow_pkts_payload.avg', 'flow_pkts_payload.std', 'fwd_iat.min',
       'fwd_iat.max', 'fwd_iat.tot', 'fwd_iat.avg', 'fwd_iat.std',
       'bwd_iat.min', 'bwd_iat.max', 'bwd_iat.tot', 'bwd_iat.avg',
       'bwd_iat.std', 'flow_iat.min', 'flow_iat.max', 'flow_iat.tot',
       'flow_iat.avg', 'flow_iat.std', 'payload_bytes_per_second',
       'fwd_subflow_pkts', 'bwd_subflow_pkts', 'fwd_subflow_bytes',
       'bwd_subflow_bytes', 'fwd_bulk_bytes', 'bwd_bulk_bytes',
       'fwd_bulk_packets', 'bwd_bulk_packets', 'fwd_bulk_rate',
       'bwd_bulk_rate', 'active.min', 'active.max', 'active.tot', 'active.avg',
       'active.std', 'idle.min', 'idle.max', 'idle.tot', 'idle.avg',
       'idle.std', 'fwd_init_window_size', 'bwd_init_window_size',
       'fwd_last_window_size'],
      dtype='object')
```

```
df = pd.concat([X,y], axis=1) # concat the two data and put it in a data frame
df
```

	id.orig_p	id.resp_p	proto	service	flow_duration	fwd_pkts_tot	bwd_pkts_tot	fwd_data_pkts_tot	bwd_data_pkts_tot	fwd_pkts_
0	38667	1883	tcp	mqtt	32.011598	9	5	3	3	0
1	51143	1883	tcp	mqtt	31.883584	9	5	3	3	0
2	44761	1883	tcp	mqtt	32.124053	9	5	3	3	0
3	60893	1883	tcp	mqtt	31.961063	9	5	3	3	0
4	51087	1883	tcp	mqtt	31.902362	9	5	3	3	0
...	...	...	...	...	...	...	...	...	...	
123112	59247	63331	tcp	-	0.000006	1	1	0	0	167772
123113	59247	64623	tcp	-	0.000007	1	1	0	0	144631
123114	59247	64680	tcp	-	0.000006	1	1	0	0	167772
123115	59247	65000	tcp	-	0.000006	1	1	0	0	167772
123116	59247	65129	tcp	-	0.000006	1	1	0	0	167772

123117 rows × 84 columns

```
df.head() #read the concatenated data frame which are the X and y
```

	id.orig_p	id.resp_p	proto	service	flow_duration	fwd_pkts_tot	bwd_pkts_tot	fwd_data_pkts_tot	bwd_data_pkts_tot	fwd_pkts_per_se
0	38667	1883	tcp	mqtt	32.011598	9	5	3	3	0.2811
1	51143	1883	tcp	mqtt	31.883584	9	5	3	3	0.2822
2	44761	1883	tcp	mqtt	32.124053	9	5	3	3	0.2801
3	60893	1883	tcp	mqtt	31.961063	9	5	3	3	0.2815
4	51087	1883	tcp	mqtt	31.902362	9	5	3	3	0.2821

5 rows × 84 columns

```
df.dtypes
```

```
id.orig_p                int64
id.resp_p                int64
proto                   object
service                 object
flow_duration          float64
                         ...
idle.std               float64
fwd_init_window_size     int64
bwd_init_window_size     int64
fwd_last_window_size     int64
Attack_type             object
Length: 84, dtype: object
```

let's see which Dtype of column i should convert into numerical by using .info()

```
df.info()
```

```
 43  flow_pkts_payload.std     123117 non-null  float64
 44  fwd_iat.min               123117 non-null  float64
 45  fwd_iat.max               123117 non-null  float64
 46  fwd_iat.tot               123117 non-null  float64
 47  fwd_iat.avg               123117 non-null  float64
 48  fwd_iat.std               123117 non-null  float64
 49  bwd_iat.min               123117 non-null  float64
 50  bwd_iat.max               123117 non-null  float64
 51  bwd_iat.tot               123117 non-null  float64
 52  bwd_iat.avg               123117 non-null  float64
 53  bwd_iat.std               123117 non-null  float64
 54  flow_iat.min              123117 non-null  float64
 55  flow_iat.max              123117 non-null  float64
 56  flow_iat.tot              123117 non-null  float64
 57  flow_iat.avg              123117 non-null  float64
 58  flow_iat.std              123117 non-null  float64
 59  payload_bytes_per_second  123117 non-null  float64
 60  fwd_subflow_pkts          123117 non-null  float64
 61  bwd_subflow_pkts          123117 non-null  float64
 62  fwd_subflow_bytes         123117 non-null  float64
 63  bwd_subflow_bytes         123117 non-null  float64
 64  fwd_bulk_bytes            123117 non-null  float64
 65  bwd_bulk_bytes            123117 non-null  float64
 66  fwd_bulk_packets          123117 non-null  float64
 67  bwd_bulk_packets          123117 non-null  float64
 68  fwd_bulk_rate             123117 non-null  float64
 69  bwd_bulk_rate             123117 non-null  float64
 70  active.min                123117 non-null  float64
 71  active.max                123117 non-null  float64
 72  active.tot                123117 non-null  float64
 73  active.avg                123117 non-null  float64
 74  active.std                123117 non-null  float64
 75  idle.min                  123117 non-null  float64
 76  idle.max                  123117 non-null  float64
 77  idle.tot                  123117 non-null  float64
 78  idle.avg                  123117 non-null  float64
 79  idle.std                  123117 non-null  float64
 80  fwd_init_window_size      123117 non-null  int64
 81  bwd_init_window_size      123117 non-null  int64
 82  fwd_last_window_size      123117 non-null  int64
 83  Attack_type               123117 non-null  object
dtypes: float64(47), int64(34), object(3)
memory usage: 78.9+ MB
```

since the proto, service, and Attack_type has the only object Dtypes, im converting it into numerical

```
df.proto.describe()
```

```
count     123117
unique         3
top          tcp
freq      110427
Name: proto, dtype: object
```

```
df['proto']
```

```
0         tcp
1         tcp
2         tcp
3         tcp
4         tcp
          ...
123112    tcp
123113    tcp
123114    tcp
123115    tcp
123116    tcp
Name: proto, Length: 123117, dtype: object
```

we want to know what's inside the proto column

```
df['proto'].unique()
```

```
array(['tcp', 'udp', 'icmp'], dtype=object)
```

assign the unique data of proto in a new variable

```
protocol = df['proto'].unique()

protocol_dict = {value: x for x, value in enumerate(protocol)}
df['proto'] = df['proto'].apply(lambda x:protocol_dict[x])

df
#convert the proto into numerical using lambda
```

	id.orig_p	id.resp_p	proto	service	flow_duration	fwd_pkts_tot	bwd_pkts_tot	fwd_data_pkts_tot	bwd_data_pkts_tot	fwd_pkts_
0	38667	1883	0	mqtt	32.011598	9	5	3	3	0
1	51143	1883	0	mqtt	31.883584	9	5	3	3	0
2	44761	1883	0	mqtt	32.124053	9	5	3	3	0
3	60893	1883	0	mqtt	31.961063	9	5	3	3	0
4	51087	1883	0	mqtt	31.902362	9	5	3	3	0
...	...	...	...	...	...	...	...	...	...	
123112	59247	63331	0	-	0.000006	1	1	0	0	167772
123113	59247	64623	0	-	0.000007	1	1	0	0	144631
123114	59247	64680	0	-	0.000006	1	1	0	0	167772
123115	59247	65000	0	-	0.000006	1	1	0	0	167772
123116	59247	65129	0	-	0.000006	1	1	0	0	167772

123117 rows × 84 columns

```
df.describe() #show the EDA
```

	id.orig_p	id.resp_p	proto	flow_duration	fwd_pkts_tot	bwd_pkts_tot	fwd_data_pkts_tot	bwd_data_pkts_tot	fwd
count	123117.000000	123117.000000	123117.000000	123117.000000	123117.000000	123117.000000	123117.000000	123117.000000	
mean	34639.258738	1014.305092	0.103536	3.809566	2.268826	1.909509	1.471218	0.820260	
std	19070.620354	5256.371994	0.306174	130.005408	22.336565	33.018311	19.635196	32.293948	
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	17702.000000	21.000000	0.000000	0.000001	1.000000	1.000000	1.000000	0.000000	
50%	37221.000000	21.000000	0.000000	0.000004	1.000000	1.000000	1.000000	0.000000	
75%	50971.000000	21.000000	0.000000	0.000005	1.000000	1.000000	1.000000	0.000000	
max	65535.000000	65389.000000	2.000000	21728.335580	4345.000000	10112.000000	4345.000000	10105.000000	

8 rows × 82 columns