Installation of UCI

```
pip install ucimlrepo
```

```
Collecting ucimlrepo
  Downloading ucimlrepo-0.0.6-py3-none-any.whl (8.0 kB)
Installing collected packages: ucimlrepo
Successfully installed ucimlrepo-0.0.6
```

Fetchin the dataset using the import ucimlrepo

```
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
                  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
```

```
3              no
4              no
..            ...
80             no
81             no
82             no
83             no
84             no

[85 rows x 7 columns]
```

The data (X) that we fetched

```
X
```

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

123117 rows × 83 columns

The data (y) that we fetched

```
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

Applying concat to join the two data which is the x data and y data

```
import pandas as pd
import numpy as np

df = pd.concat((X, y), axis = 1)
df
```

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

123117 rows × 84 columns

```
df.describe()
#Describing what's within the data
```

	id.orig_p	id.resp_p	flow_duration	fwd_pkts_tot	bwd_pkts_tot	fwd_d
count	123117.000000	123117.000000	123117.000000	123117.000000	123117.000000	
mean	34639.258738	1014.305092	3.809566	2.268826	1.909509	
std	19070.620354	5256.371994	130.005408	22.336565	33.018311	
min	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	17702.000000	21.000000	0.000001	1.000000	1.000000	
50%	37221.000000	21.000000	0.000004	1.000000	1.000000	
75%	50971.000000	21.000000	0.000005	1.000000	1.000000	
max	65535.000000	65389.000000	21728.335580	4345.000000	10112.000000	

8 rows × 81 columns

Identifying the column names

```
df.columns
#Getting each name of the 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',
       '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', 'Attack_type'],
      dtype='object')
```

```
df.head(20)
#first 20 of the data
```

	id.orig_p	id.resp_p	proto	service	flow_duration	fwd_pkts_tot	bwd_pkts_tot
0	38667	1883	tcp	mqtt	32.011598	9	5
1	51143	1883	tcp	mqtt	31.883584	9	5
2	44761	1883	tcp	mqtt	32.124053	9	5
3	60893	1883	tcp	mqtt	31.961063	9	5
4	51087	1883	tcp	mqtt	31.902362	9	5
5	48579	1883	tcp	mqtt	31.869686	9	5
6	54063	1883	tcp	mqtt	32.094711	9	5
7	33457	1883	tcp	mqtt	32.104011	9	5
8	52181	1883	tcp	mqtt	32.026967	9	5
9	53469	1883	tcp	mqtt	32.048637	9	5
10	54153	1883	tcp	mqtt	31.977057	9	5
11	39671	1883	tcp	mqtt	31.962308	9	5
12	44225	1883	tcp	mqtt	31.965302	9	5
13	51495	1883	tcp	mqtt	31.885127	9	5
14	42037	1883	tcp	mqtt	31.926578	9	5
15	36349	1883	tcp	mqtt	32.061416	9	5
16	39763	1883	tcp	mqtt	32.025109	9	5
17	57501	1883	tcp	mqtt	31.908247	9	5
18	56117	1883	tcp	mqtt	32.009238	9	5
19	44927	1883	tcp	mqtt	31.930485	9	5

20 rows × 84 columns

```
df.tail(20)
df.tail(20)
#Last 20 of the data
```

	id.orig_p	id.resp_p	proto	service	flow_duration	fwd_pkts_tot	bwd_pkts_t
123097	59247	50389	tcp	-	0.000005	1	
123098	59247	50500	tcp	-	0.000008	1	
123099	59247	50636	tcp	-	0.000006	1	
123100	59247	51103	tcp	-	0.000006	1	
123101	59247	51493	tcp	-	0.000008	1	
123102	59247	52848	tcp	-	0.000003	1	
123103	59247	54045	tcp	-	0.000007	1	
123104	59247	54328	tcp	-	0.000006	1	
123105	59247	55055	tcp	-	0.000006	1	
123106	59247	55056	tcp	-	0.000002	1	
123107	59247	55600	tcp	-	0.000007	1	
123108	59247	57797	tcp	-	0.000006	1	
123109	59247	60020	tcp	-	0.000007	1	
123110	59247	60443	tcp	-	0.000006	1	
123111	59247	61900	tcp	-	0.000007	1	
123112	59247	63331	tcp	-	0.000006	1	
123113	59247	64623	tcp	-	0.000007	1	
123114	59247	64680	tcp	-	0.000006	1	
123115	59247	65000	tcp	-	0.000006	1	
123116	59247	65129	tcp	-	0.000006	1	

20 rows × 84 columns

```
print("Total number of data: ", len(df))
#getting the number of datas
```

```
Total number of data:  123117
```

Identifying the data types of the data

```
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
```

```
df['proto'].unique()
#Using unique to identify the three data types of the proto
```

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

```
df['service'].unique()
#Using unique to identify the data types of service
```

```
array(['mqtt', '-', 'http', 'dns', 'ntp', 'ssl', 'dhcp', 'irc', 'ssh',
       'radius'], dtype=object)
```

```
df['Attack_type'].unique()
#Using unique to identify the data type of Attack_type
```

```
array(['MQTT_Publish', 'Thing_Speak', 'Wipro_bulb', 'ARP_poisioning',
       'DDOS_Slowloris', 'DOS_SYN_Hping', 'Metasploit_Brute_Force_SSH',
       'NMAP_FIN_SCAN', 'NMAP_OS_DETECTION', 'NMAP_TCP_scan',
       'NMAP_UDP_SCAN', 'NMAP_XMAS_TREE_SCAN'], dtype=object)
```

```
def preprocess(df, column_name):
    if df[column_name].dtype == 'object':
        df[column_name].replace(to_replace=df[column_name].unique(), value = range(len(df[
```

```
preprocess(df, 'proto')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-77-1a4a2f49df80> in <cell line: 1>()
----> 1 preprocess(df, 'proto')

<ipython-input-76-6b943bbf6b9d> in preprocess(df, column_name)
      1 def preprocess(df, column_name):
----> 2     if df[column_name].dtype == 'object':
      3         df[column_name].replace(to_replace=df[column_name].unique(), value =
range(len(df[column_name].unique())), inplace = True)
```

```
TypeError: 'NoneType' object is not subscriptable
```

```
df.dtypes
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-70-5cc0934cc03c> in <cell line: 1>()
----> 1 df.dtypes

AttributeError: 'NoneType' object has no attribute 'dtypes'
```

Conclusion we did a lot of new things like fetching the data set and concatenating it into one and making it into a one data frame which we applied cleaning data, and also making the objects in data types into int where the unique value of it will change into numeric.