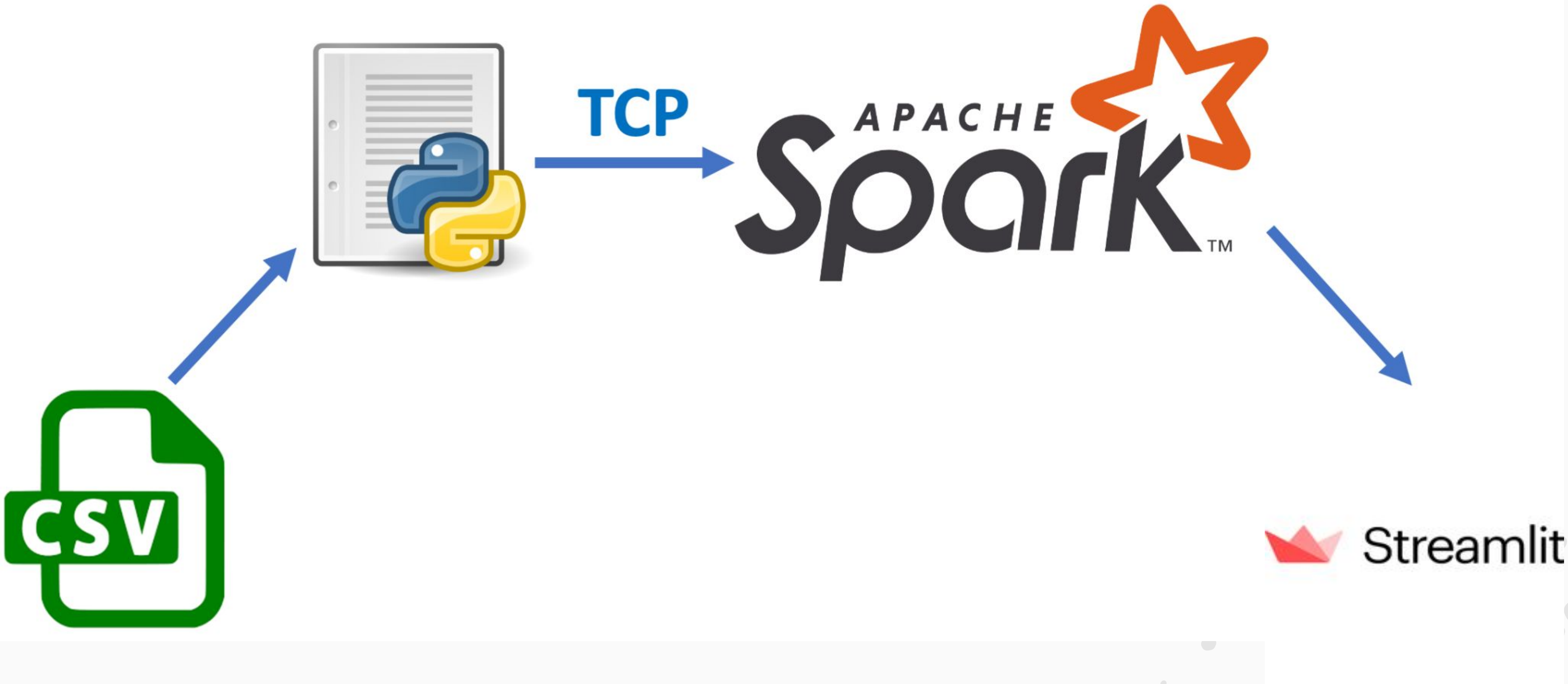
Problem Definition

- Network monitoring is important to identify and avoid potential issues before they occur.
- It is also important for network security and threat analysis.
- Increase in number of devices could affect performance of devices connected to network
- Pinpointing resource intensive devices and the degree of utilization as compared to other devices is vital in ensuring that the network continues to function as expected

Solution:

Query-Based Network Monitoring System

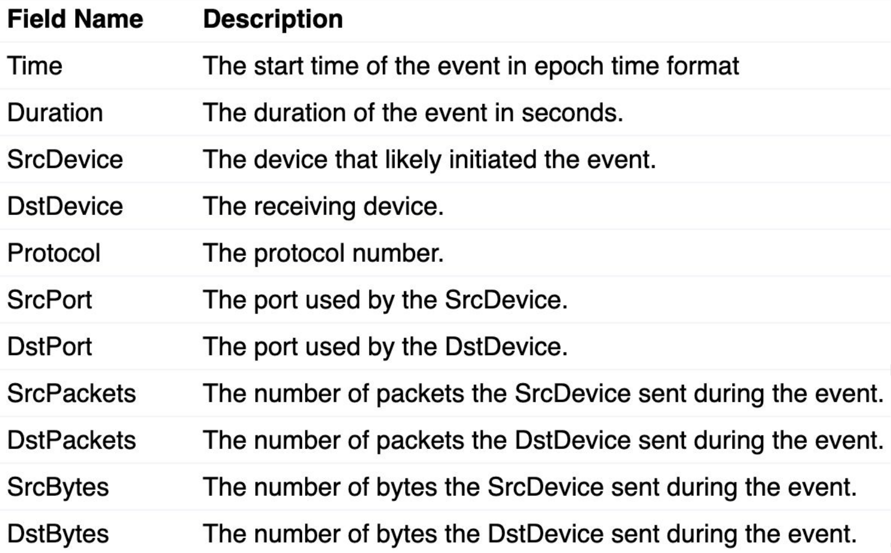
System Architecture



Data Input

- **Unified Host & Network** data set: network event data

- Python script reads data from csv and sends to Spark script via TCP row by row



Field Name	Description
Time	The start time of the event in epoch time format
Duration	The duration of the event in seconds.
SrcDevice	The device that likely initiated the event.
DstDevice	The receiving device.
Protocol	The protocol number.
SrcPort	The port used by the SrcDevice.
DstPort	The port used by the DstDevice.
SrcPackets	The number of packets the SrcDevice sent during the event.
DstPackets	The number of packets the DstDevice sent during the event.
SrcBytes	The number of bytes the SrcDevice sent during the event.
DstBytes	The number of bytes the DstDevice sent during the event.

Spark

Queries

1. List the devices that are consuming more than H percent of the total external bandwidth over the last T time units.
2. List the top-k most resource intensive devices over the last T time units.
3. List all the devices that are consuming more than X times the standard deviation of the average traffic consumption of all the devices over the last T time units.

Implementation

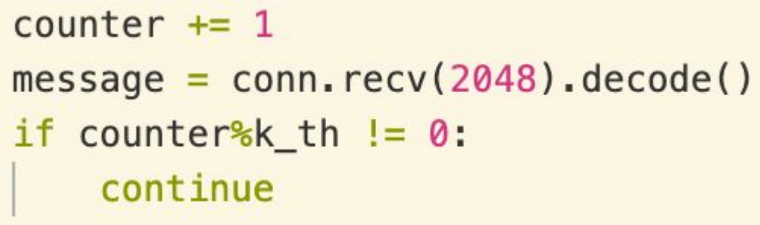
Window - Sliding windows: slide = 10
Tuples are appended to a list until list size reaches window size.
Earliest 10 tuples removed and new tuples are appended again.

Overall - since the beginning
Tuples are continuously appended to a list.

1. List is converted to dataframe
2. Temporary dataframe containing sum of total packets from distinct source devices and source device name was created for queries

Streaming Algorithm: Sampling

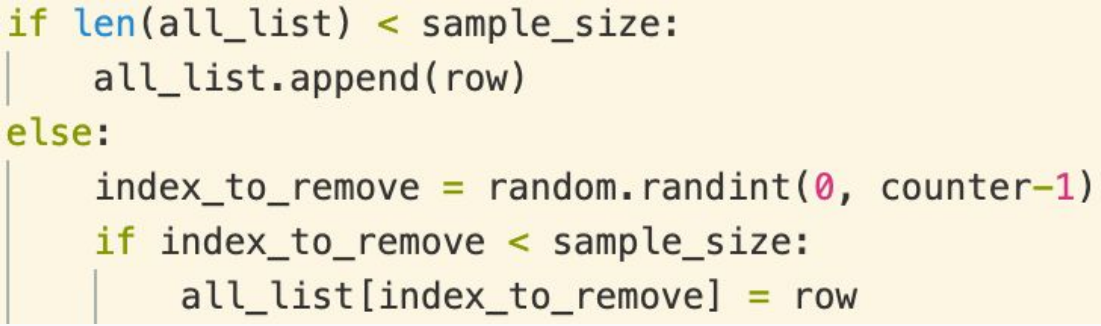
Uniform Sampling



```
counter += 1
message = conn.recv(2048).decode()
if counter%k_th != 0:
    continue
```

Decrease in value of k_th,
Increase in accuracy

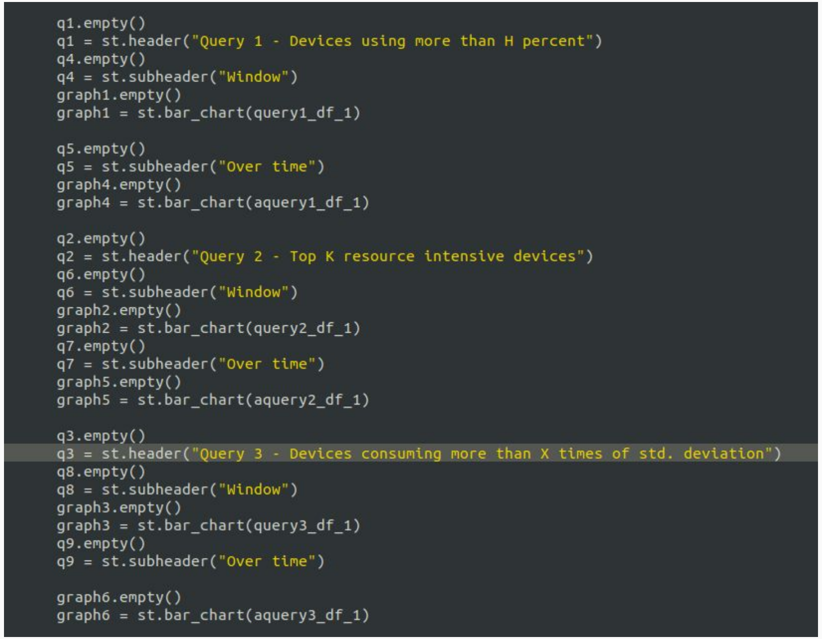
Random Sampling: Reservoir Sampling



```
if len(all_list) < sample_size:
    all_list.append(row)
else:
    index_to_remove = random.randint(0, counter-1)
    if index_to_remove < sample_size:
        all_list[index_to_remove] = row
```

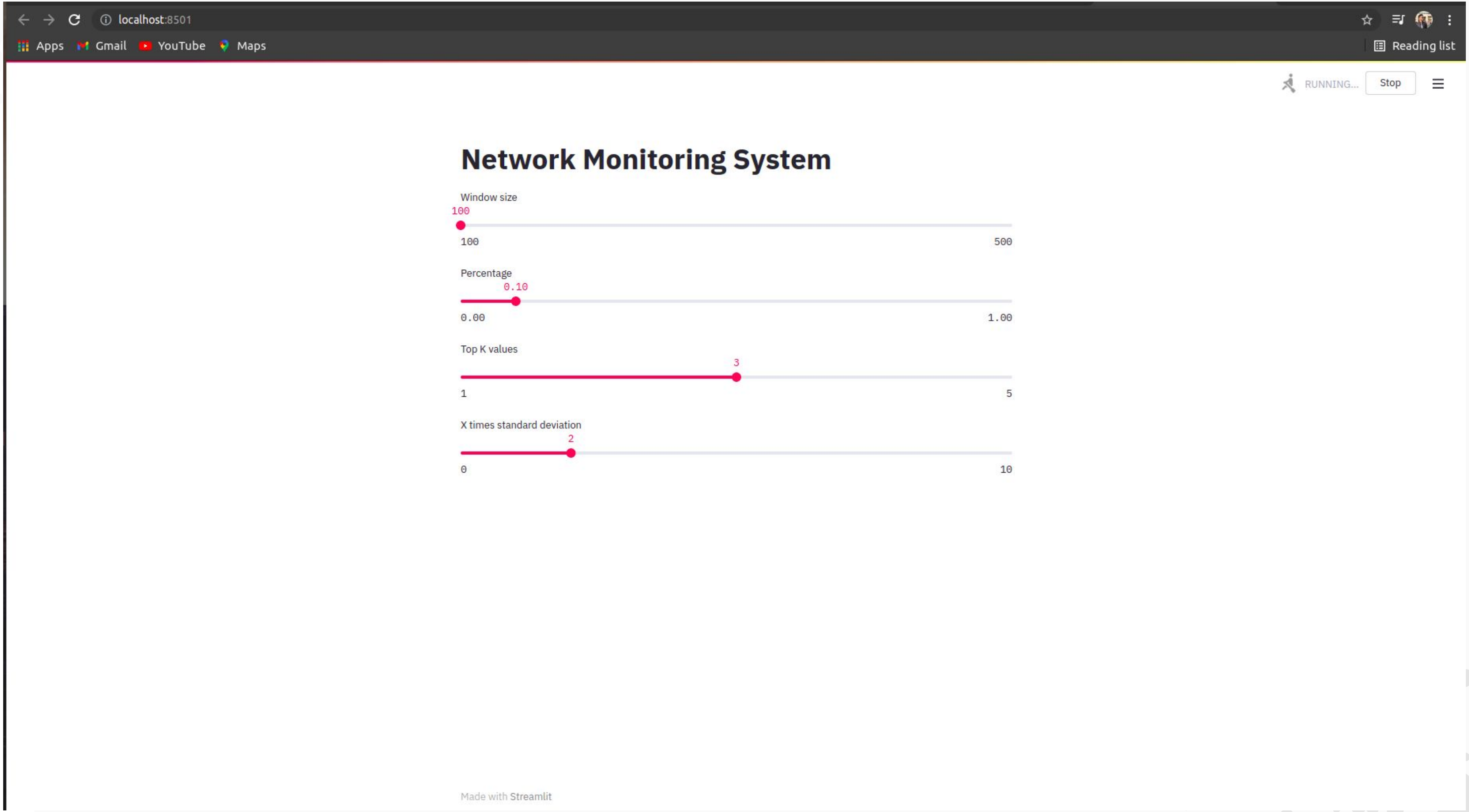
Increase in sample size,
Increase in accuracy

GUI



```
q1.empty()
q1 = st.header("Query 1 - Devices using more than H percent")
q4.empty()
q4 = st.subheader("Window")
graph1.empty()
graph1 = st.bar_chart(query1_df_1)

q5.empty()
q5 = st.subheader("Over time")
graph4.empty()
graph4 = st.bar_chart(aquery1_df_1)

q2.empty()
q2 = st.header("Query 2 - Top K resource intensive devices")
q6.empty()
q6 = st.subheader("Window")
graph2.empty()
graph2 = st.bar_chart(query2_df_1)
q7.empty()
q7 = st.subheader("Over time")
graph5.empty()
graph5 = st.bar_chart(aquery2_df_1)

q3.empty()
q3 = st.header("Query 3 - Devices consuming more than X times of std. deviation")
q8.empty()
q8 = st.subheader("Window")
graph3.empty()
graph3 = st.bar_chart(query3_df_1)
q9.empty()
q9 = st.subheader("Over time")

graph6.empty()
graph6 = st.bar_chart(aquery3_df_1)
```

Streamlit



Network Monitoring System

Window size

100

100 500

Percentage

0.10

0.00 1.00

Top K values

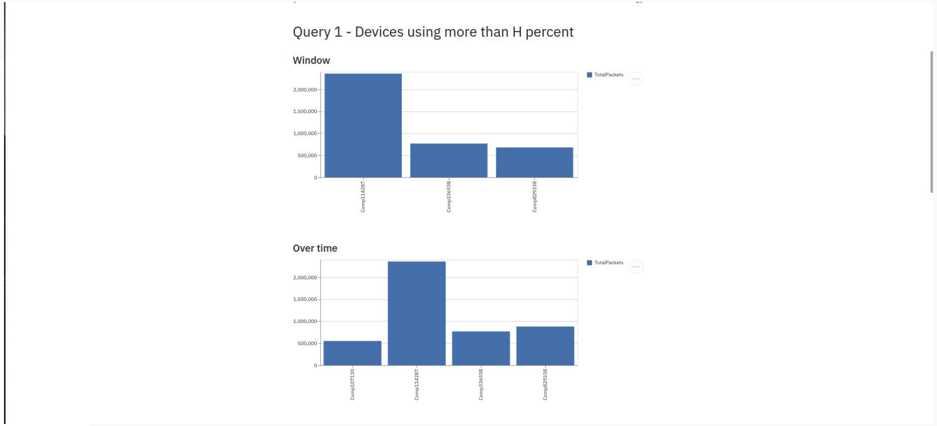
3

1 5

X times standard deviation

2

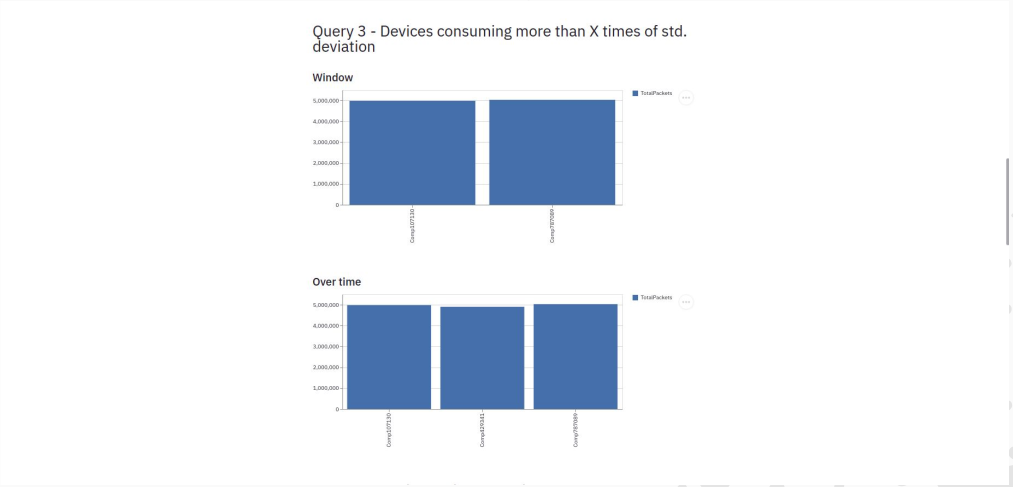
0 10



Query 1 - Devices using more than H percent

Window

Over time



Query 3 - Devices consuming more than X times of std. deviation

Window

Over time

Demo