Network Intrusion Detection

Dharu Piraba Muguntharaman: dm5596@nyu.edu
Mohanna R S: mr6552@nyu.edu
Shamyukta Rajagopal: sr6626@nyu.edu

Agenda

- Objective
- Problem Statement
- Dataset Description
- Technologies Used
- Architecture Design
- Exploratory Data Analysis
- Results
- Conclusion
- Challenges & Future Work

Problem Statement

- With the increase in online activities and the growing number of cyber threats, it is becoming increasingly important to have an effective Intrusion Detection System (IDS).

- Traditional IDS solutions struggle to keep up with the massive amounts of data generated by modern networks, making it difficult to detect and respond to potential threats in real-time.

- By leveraging the power of Spark, we aim to build IDS solution capable of processing large amounts of data in real-time, enabling organizations to detect and respond to potential threats more effectively.

Objectives

- The primary objective of this project is to develop a system that can effectively detect and prevent security breaches in a network environment.
- This involves analyzing large volumes of network traffic data in real-time to identify suspicious activity that may indicate a potential intrusion or attack.
- Utilized developing machine learning models and algorithms to improve the accuracy of intrusion detection and reduce false positives.
- Enhance the security of a network and protect against cyber threats that could compromise sensitive data and disrupt business operations.

Dataset Description

- Dataset: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
  The datasets contain a total of 24 attack types, with an additional 14 types in the test data.
  Each row of data contains 42 attributes.

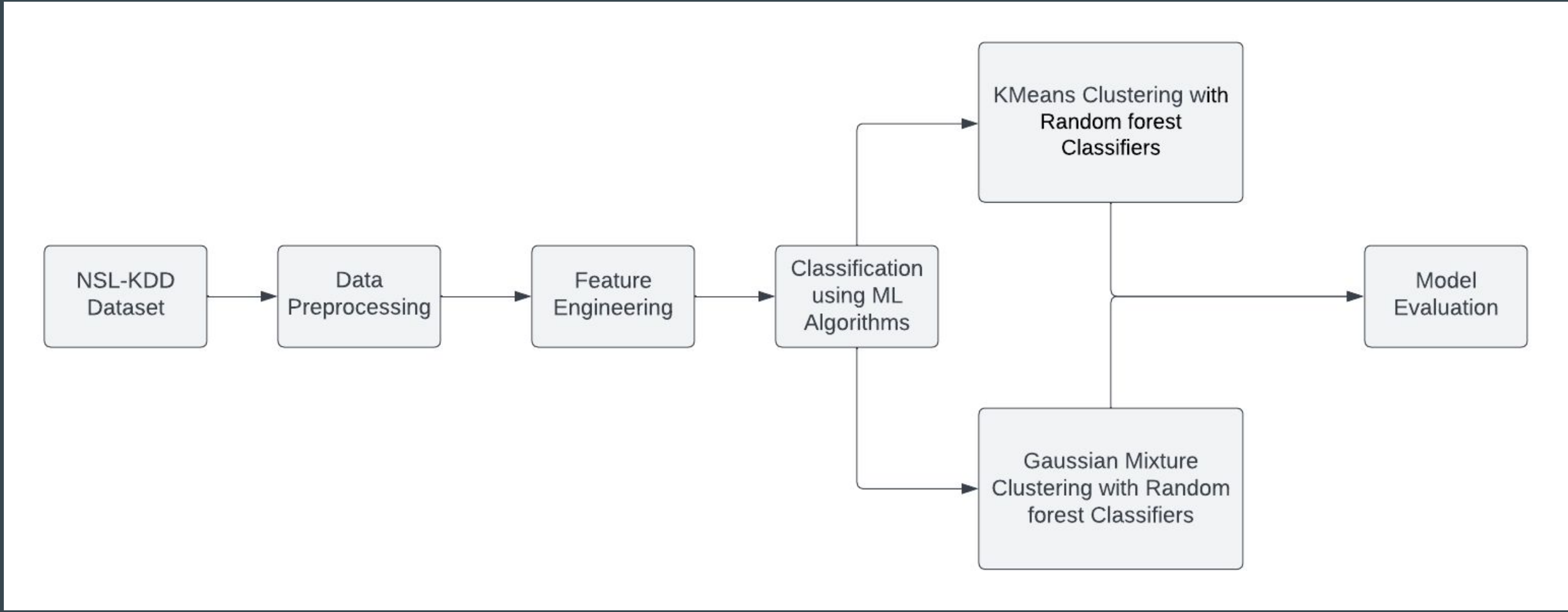
- Size: 4GB

- Relevant schema description:
  - protocol_type - type of the protocol, e.g. tcp, udp, icmp
  - Service - network service on the destination, e.g., http, telnet
  - src_bytes -number of data bytes from source to destination
  - dst_bytes - number of data bytes from destination to source
  - flag - normal or error status of the connection
  - Num_failed_logins- number of failed login attempts
  - logged_in - 1 if successfully logged in; 0 otherwise
  - root_shell - 1 if root shell is obtained; 0 otherwise
  - su_attempted - 1 if`su root" command attempted

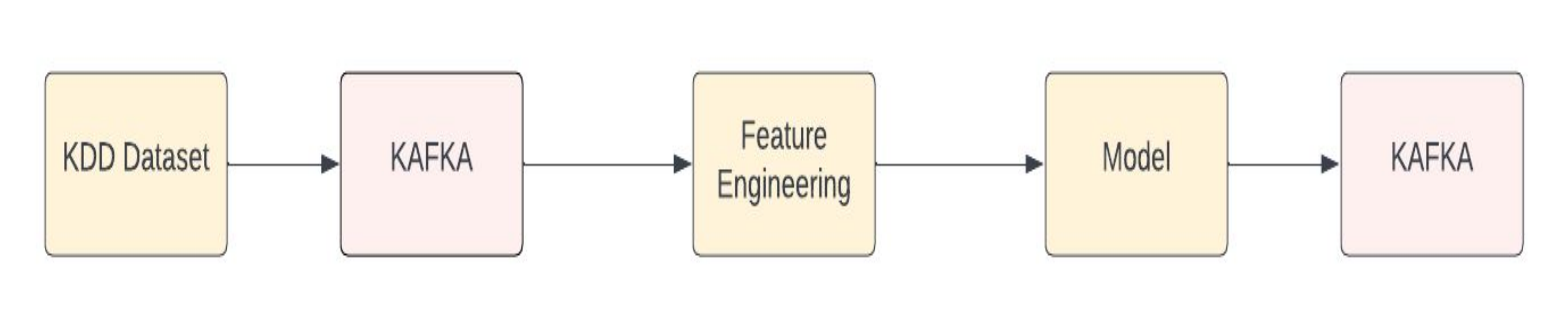
Technologies Used

PySpark	For exploratory data analysis
Pandas & Matplotlib	For visualizations
SparkML	Train and test the model on pyspark
Kafka	Data streaming

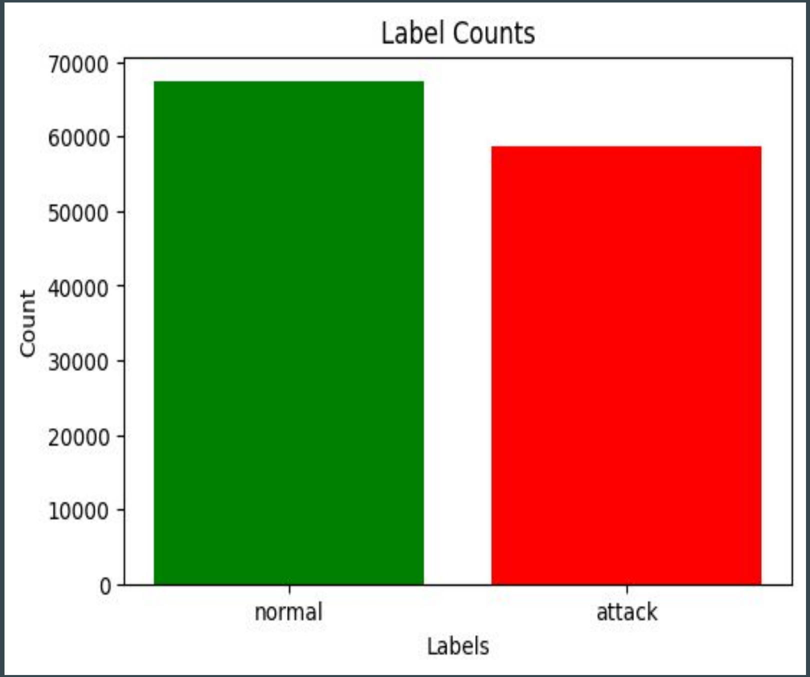
Architecture Design

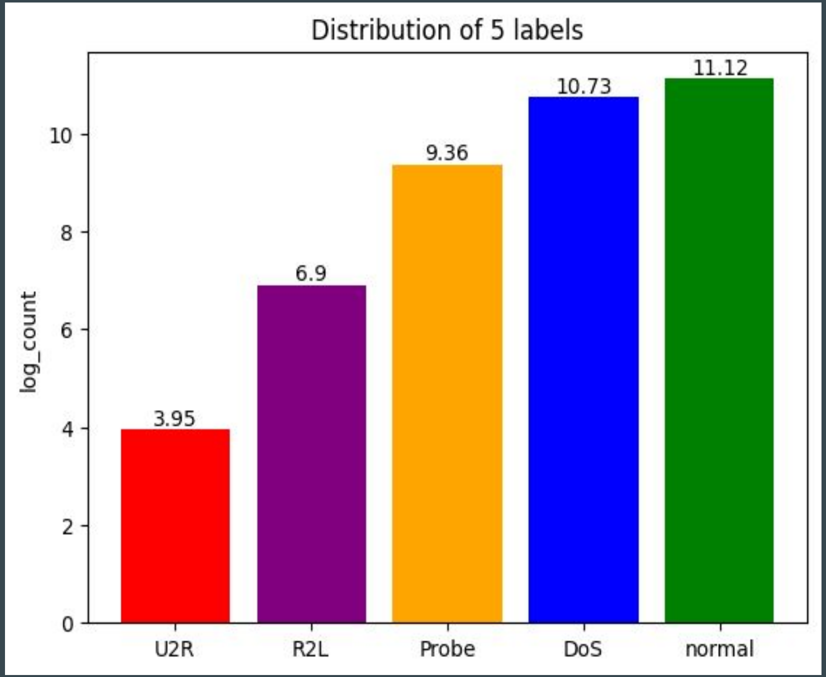


Prototype pipeline : Stream Processing and Transformation



Exploratory Data Analysis

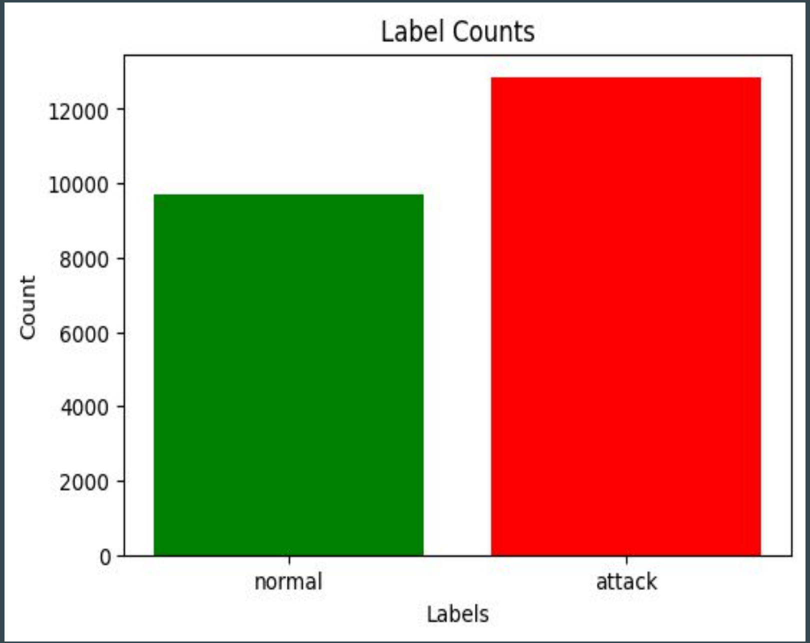


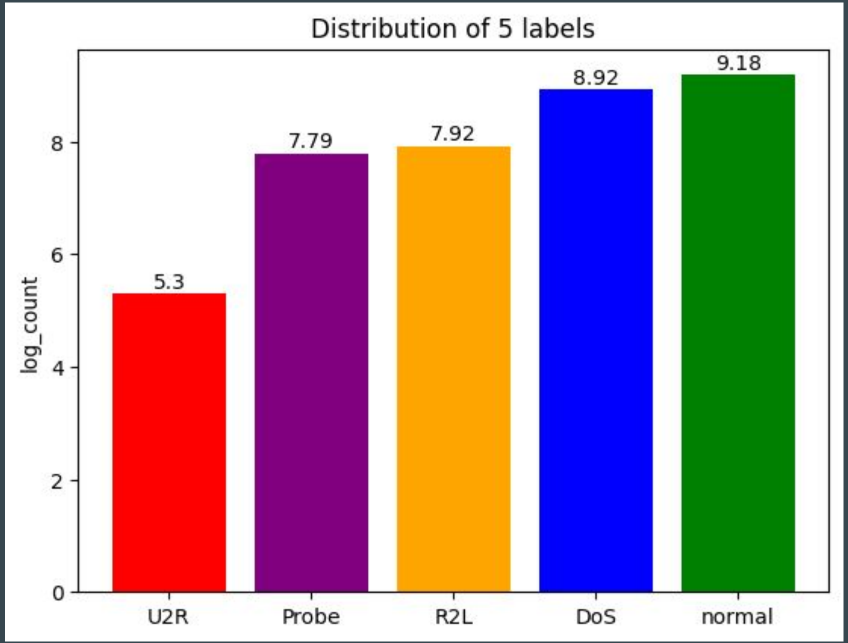


TRAINING DATA :

Labels2Convertor : Classifies the dataset into normal and attacks
Labels5Convertor : Classification of dataset into normal vs four types of attacks

Exploratory Data Analysis

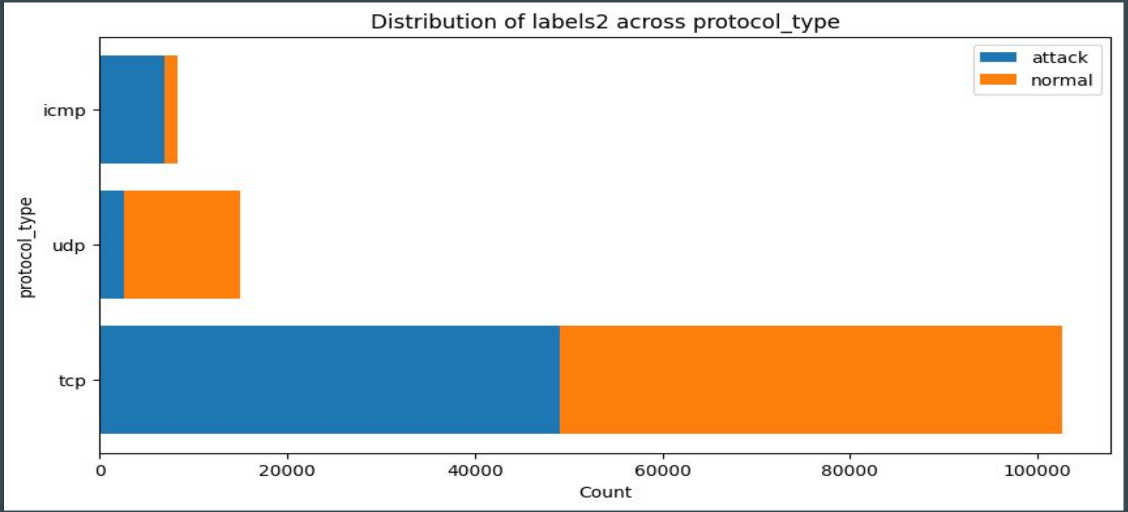




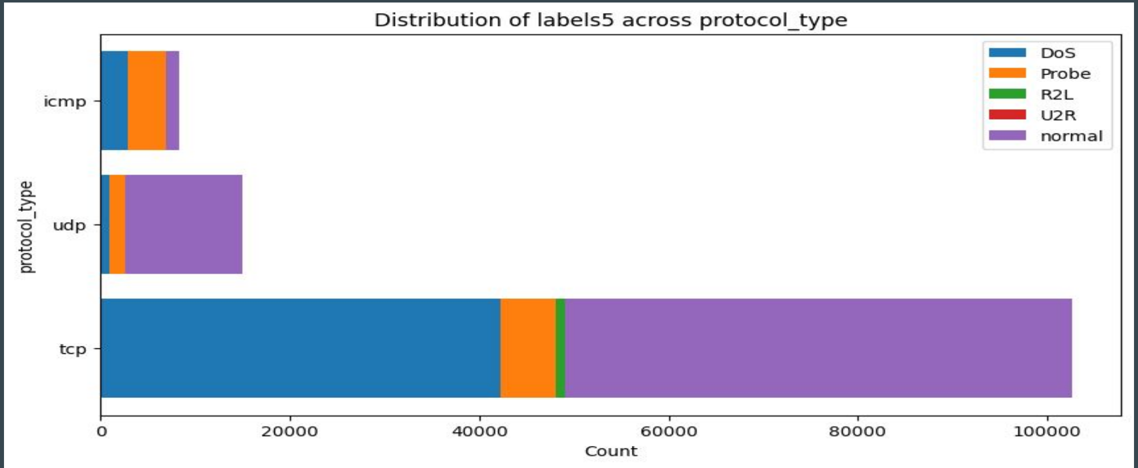
TEST DATA :

Labels2Convertor : Classifies the dataset into normal and attacks
Labels5Convertor : Classification of dataset into normal vs four types of attacks

Exploratory Data Analysis



Distribution of normal connections and attacks in different types of protocols such as TCP , UDP , ICMP.



Distribution of normal connections vs four types of attacks in different types of protocols such as TCP , UDP , ICMP.

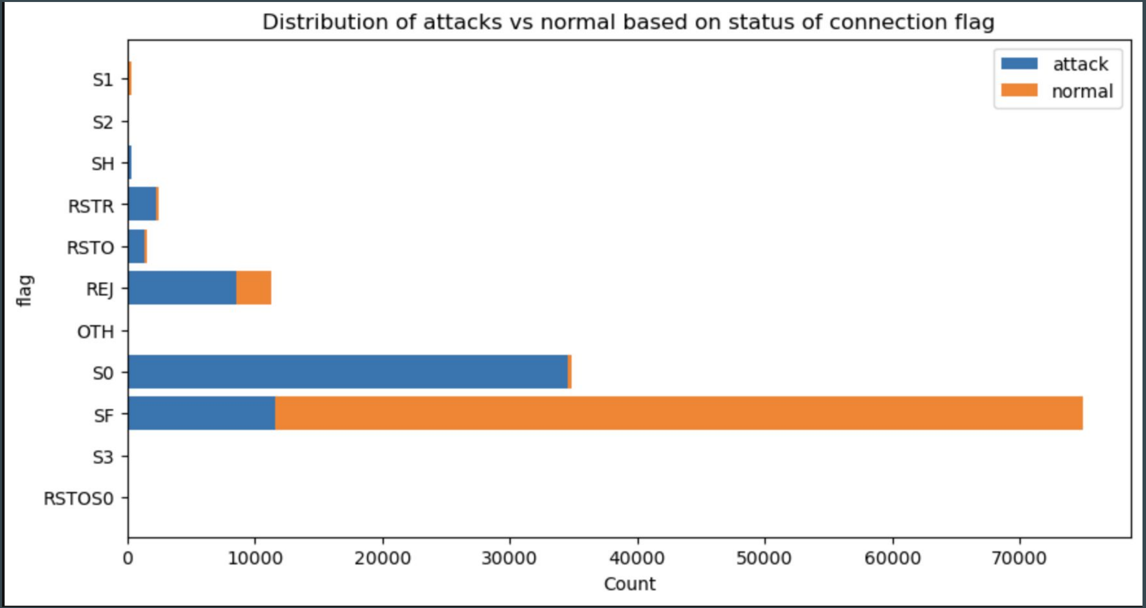
Inference : TCP is more vulnerable to attacks and DoS attack is predominant in all protocols.

Exploratory Data Analysis

```
# 'flag' column
print(train_df.select(nominal_cols[2]).distinct().count())
(train_df.crosstab(nominal_cols[2], 'labels2').sort(sql.asc
(train_df.crosstab(nominal_cols[2], 'labels5').sort(sql.asc
```

```
11
+------------+------+------+
|flag_labels2|attack|normal|
+------------+------+------+
|         OTH|    35|    11|
|         REJ|  8540|  2693|
|        RSTO|  1343|   219|
|      RSTOS0|   103|     0|
|        RSTR|  2275|   146|
|          S0| 34497|   354|
|          S1|     4|   361|
|          S2|     8|   119|
|          S3|     4|    45|
|          SF| 11552| 63393|
|          SH|   269|     2|
+------------+------+------+

+------------+-----+-----+---+---+------+
|flag_labels5|  DoS|Probe|R2L|U2R|normal|
+------------+-----+-----+---+---+------+
|         OTH|    0|   35|  0|  0|    11|
|         REJ| 5671| 2869|  0|  0|  2693|
|        RSTO| 1216|   80| 46|  1|   219|
|      RSTOS0|    0|  103|  0|  0|     0|
|        RSTR|   90| 2180|  5|  0|   146|
|          S0|34344|  153|  0|  0|   354|
|          S1|    2|    1|  1|  0|   361|
|          S2|    5|    2|  1|  0|   119|
|          S3|    0|    1|  3|  0|    45|
|          SF| 4599| 5967|935| 51| 63393|
|          SH|    0|  265|  4|  0|     2|
+------------+-----+-----+---+---+------+
```



Distribution of normal connections and attacks based on status of the connection.

Exploratory Data Analysis

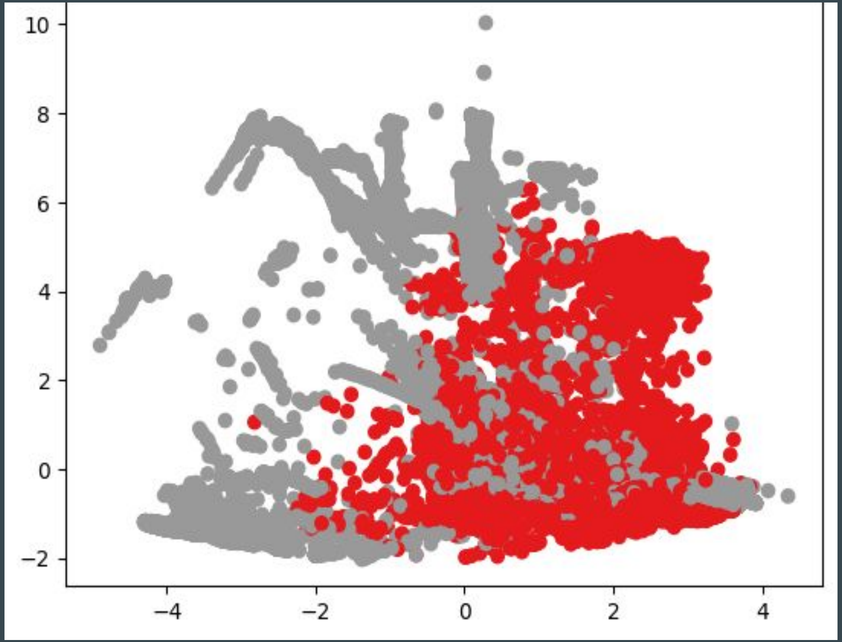
70

service_labels2	attack	normal
IRC	1	186
X11	6	67
Z39_50	862	0
aol	2	0
auth	719	236
bgp	710	0
courier	734	0
csnet_ns	545	0
ctf	563	0
daytime	521	0
discard	538	0
domain	531	38
domain_u	9	9034

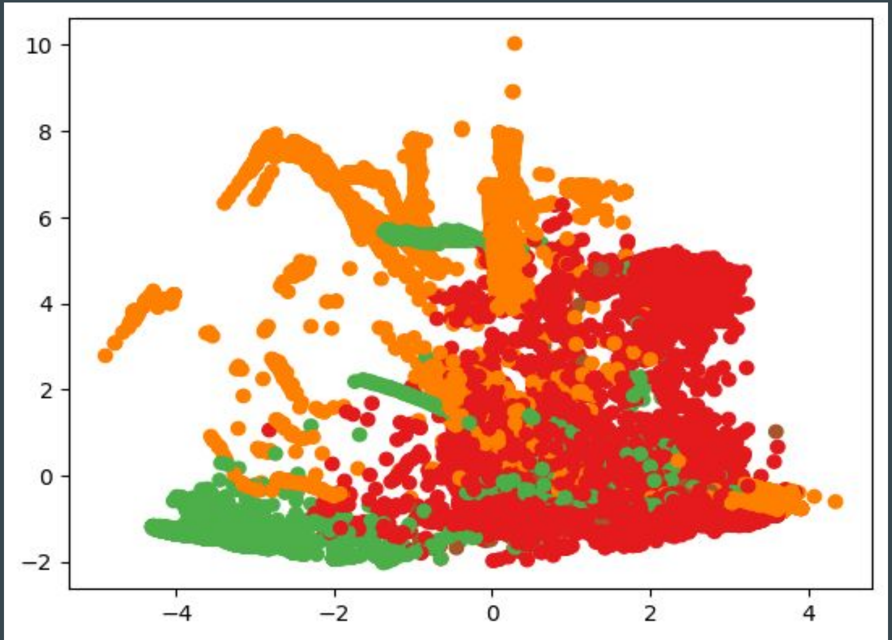
service_labels5	DoS	Probe	R2L	U2R	normal
IRC	0	1	0	0	186
X11	0	6	0	0	67
Z39_50	851	11	0	0	0
aol	0	2	0	0	0
auth	703	16	0	0	236
bgp	699	11	0	0	0
courier	726	8	0	0	0
csnet_ns	533	12	0	0	0
ctf	538	25	0	0	0
daytime	503	18	0	0	0
discard	520	18	0	0	0
domain	508	23	0	0	38
domain_u	0	9	0	0	9034
echo	416	18	0	0	0
eco_i	0	4089	0	0	497
ecr_i	2844	43	0	0	190

Distribution of normal connections and attacks in 70 different types of services like bgp, ssh, telnet.

Exploratory Data Analysis

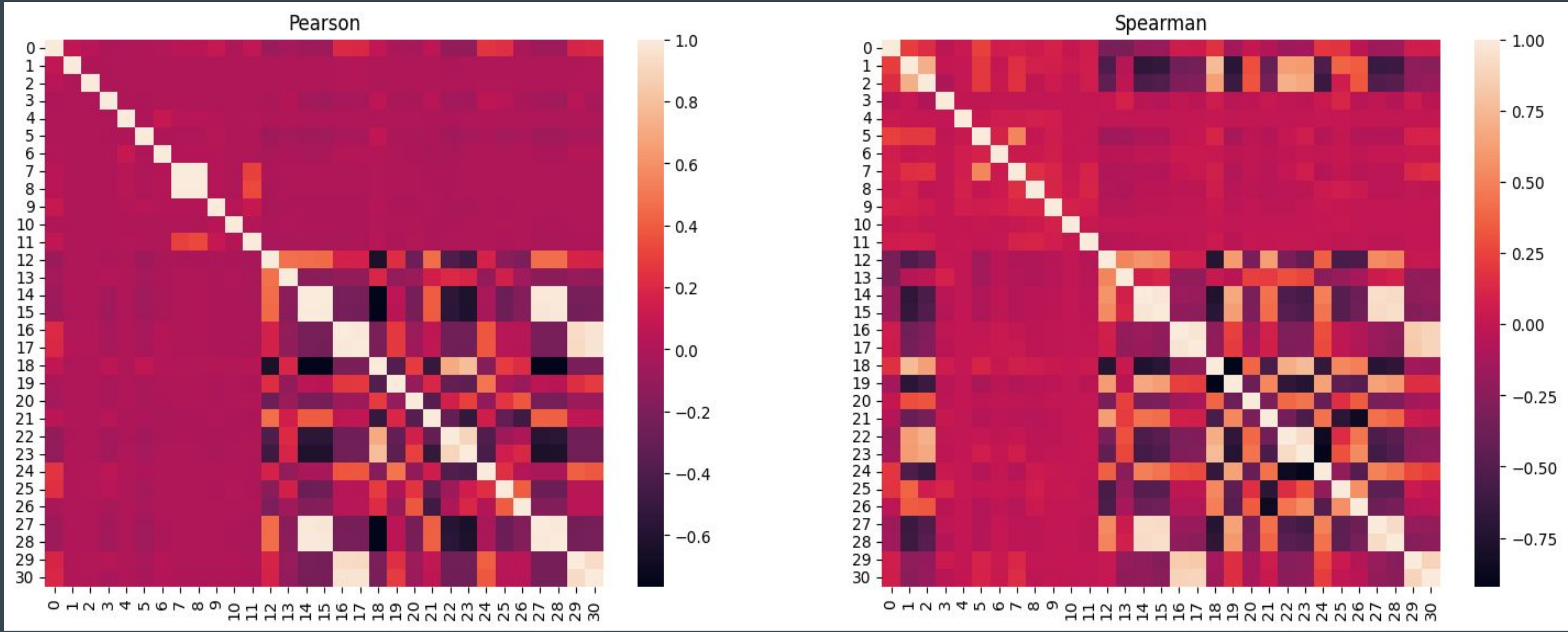


'Attack' Vs 'Normal'



'Four different Types of Attack' vs 'Normal'

Exploratory Data Analysis



PEARSON & SPEARMAN CORRELATION COEFFICIENT

Feature Selection

1. **Preprocessing:**
   The NSL-KDD dataset is preprocessed to handle missing values, normalize numeric features, and encode categorical features.

2. **AR Computation:**
   Attribute Ratio value is calculated for the Numeric cols and binary cols based on the AR metric formula.

3. **Ranking and Selection:**
   Employed feature ranking methods to assess the relevance of each feature based on their ability to discriminate between normal and attack instances.

Results - Model Performances

K-Means Clustering with Random Forest Classifiers :

```
        normal  attack
normal    8325    1386
attack     645   12188

Accuracy = 0.90991
AUC = 0.903507

False Alarm Rate = 0.142725
Detection Rate = 0.949739
F1 score = 0.923089

              precision    recall  f1-score   support

         0.0       0.93      0.86      0.89      9711
         1.0       0.90      0.95      0.92     12833

    accuracy                           0.91     22544
   macro avg       0.91      0.90      0.91     22544
weighted avg       0.91      0.91      0.91     22544
```

Cross Validation Data

```
        normal  attack
normal   13391      13
attack      29   11712

Accuracy = 0.99833
AUC = 0.99828

False Alarm Rate = 0.00096986
Detection Rate = 0.99753
F1 score = 0.99821

              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00     13404
         1.0       1.00      1.00      1.00     11741

    accuracy                           1.00     25145
   macro avg       1.00      1.00      1.00     25145
weighted avg       1.00      1.00      1.00     25145
```

Test Data

Results - Model Performances

Gaussian Mixture Clustering with Random Forest Classifiers :

	normal	attack
normal	13396	8
attack	28	11713

```
Accuracy = 0.998568
AUC = 0.998509

False Alarm Rate = 0.000596837
Detection Rate = 0.997615
F1 score = 0.998466
```

	precision	recall	f1-score	support
0.0	1.00	1.00	1.00	13404
1.0	1.00	1.00	1.00	11741
accuracy			1.00	25145
macro avg	1.00	1.00	1.00	25145
weighted avg	1.00	1.00	1.00	25145

Cross Validation Data

	normal	attack
normal	8510	1201
attack	1237	11596

```
Accuracy = 0.891856
AUC = 0.889967

False Alarm Rate = 0.123674
Detection Rate = 0.903608
F1 score = 0.904877
```

	precision	recall	f1-score	support
0.0	0.87	0.88	0.87	9711
1.0	0.91	0.90	0.90	12833
accuracy			0.89	22544
macro avg	0.89	0.89	0.89	22544
weighted avg	0.89	0.89	0.89	22544

Test Data

Conclusion

- Upon performing EDA we discovered a lot of patterns and insights from the dataset, based on protocol types, flags, service types.

- Data streaming through Kafka was efficient than storing it in a PySpark dataframe and using it for model training/testing.

- The best result from a single approach was achieved by K-Means Clustering with Random Forest Classifiers. It gives around ~98-99% of detection rate with F1 score of 0.99 and weighted avg is 1.

Challenges and Future Work

1. Spark Streaming does not allow multi-stream joins when aggregate functions are used.
2. External DB like Apache Ignite to store the processed and transformed data.
3. Ensembling approaches can be used for improving the detection rate.
4. Test different intrusion detection algorithms/approaches on a test dataset of a SIMILAR schema to our data like CSE-CIC-IDS2018.
5. Develop real-time end to end pipeline that sends stream queries to Kafka and expands the features of Zeek monitor logs through a data engineering pipeline to achieve higher resolution.

References

- https://www.naun.org/main/UPress/cc/2014/a102019-106.pdf
- 'Feature selection using attribute ratio in NSL-KDD Data' (2014) *International Conference Data Mining, Civil and Mechanical Engineering (ICDMCME'2014), Feb 4-5, 2014 Bali (Indonesia)* [Preprint]. doi:10.15242/iie.e0214081.

Thank You