North Eastern Regional Institute of Science & Technology

Arunachal Pradesh

(Computer Science & Engineering)

Project Report On**:**

CLASSIFICATION OF USER ACTIVITIES USING MACHINE LEARNING ON NETWORK TRAFFIC

Submitted By

Changnyei Phom and Rahul Prasad

Roll No. D/18/CS/113 and D/18/CS/121

Under the Supervision of

Dr. Amar Taggu

Department of Computer Science and Engineering

North Eastern Regional of Science and Technology

Arunachal Pradesh, India

CANDI DATE'S DECLARATION

We, **Changnyei Phom** and **Rahul Prasad** bearing registration number:218/037 and 218/061, students of **Degree** in **Computer Science & Engineering** Department at **North Eastern Regional Institute Of Science And Technology**, Arunachal Pradesh, hereby declare that we own full responsibility for the information, result, conclusion etc provided in this project report titled, "**CLASSIFICATION OF USER ACTIVITIES USING MACHINE LEARNING ON NETWORK TRAFFIC**" submitted to **NERIST, Itanagar, Arunachal Pradesh**, India. We have completely taken care in acknowledging the contribution of others in this academic work. We further declare that in case of any violation of intellectual property rights or copyrights foundat any stage, We, as candidates will be solely responsible for the same. Our supervisor or Head of department and institute should not be held for full or partial violation of copyright if found atany stage of our degree.

Date:16-06-22

Place: NERIST, Arunachal Pradesh

(**Rahul Prasad**)

And

(Changnyei Phom)

CERTIFICATE

It is to certify that Changnyei Phom (D/18/CS/113), Rahul Prasad (D/18/CS/121) had carried out the project work presented in this project entitled "**CLASSIFICATION OF USER ACTIVITIES USING MACHINE LEARNING ON NETWORK TRAFFIC**" for the award of Degree in Computer Science & Engineering to North Eastern Regional Institute of Science and Technology, Nirjuli, Itanagar, Arunachal Pradesh Deemed to be University, under my supervision. The contents of the project do not form the basis for the award of any other degree to the candidate or to anybody else.

Date: 16-06-2022

Project Supervisor:

(Ashwini Kumar Patra)
Asst. Professor Department of CSE.

ACKNOWLEDGEMENT

A successful and satisfactory completion of any project is the outcome of invaluable and aggregate contribution of different personal pulls in radial directions, explicitly or implicitly.

Like every big thing, this project is not only our effort. A lot of others who supported us in various ways, while some in technical way, and some, in moral way.

We humbly like to express our gratitude to our Project guide **Dr.Amar Taggu**, Assistant Professor, Dept. of Computer Science & Engineering, NERIST who has helped us a lot in providing all facilities needed and proper guidance and coordination in completing the project within specified time for the completion of this project. The suggestions and criticism of these people significantly improved the project.

ABSTRACT

In this project an attempt has been made to classify user activities using Machine Learning Algorithm such as K-Means and SVM. Python has been used for the implementation purpose. Each of multiple traffic traces from the dataset is arranged belonging to certain user and activity. The dataset has network traffic traces in CSV format.

TABLE OF CONTENT

1. INTRODUCTION

Analysis and monitoring of network has become more challenging because of a high demand in growth of traffic. Many applications thus causing the network management challenges as well as resulting the network by increasing the complexity in capturing of application to the network. That particular identity is very beneficial for making use of Quality of Service (QoS) rules and detecting intrusion amongst different tasks. Each utility has its very own community requirement therefore, it's miles critical to become aware of those necessities to control and offer the proper aid had to make certain an appropriate operation.

The conventional tactics for network traffic classification are in large part primarily based totally on popularity of famous ports and Data Packet Inspection (DPI).

However, there are numerous packages that makes use of dynamic ports, and lots of online offerings are then transported through HTTP making it extra tough to understand the ones offerings with a port-primarily based totally approach. The boost in users' private involved has accelerated using HTTPS digital personal community along with the encrypted tunnels. Implementing provider category primarily based totally on DPI unfeasible.

Due to this factor, new evaluation technique consisting of the ones primarily based totally on device mastering approach have acquired and growing recognition amongst educational researchers and commercial users.

The dataset consists of numerous site visitors lines every belonging to a positive consumer and activity. Every of the packet is properly characterised through the subsequent features like the timestamp of the packet, designation of being a TCP/UDP packet, the payload length in bytes, supply and destination IP addresses and transport port involved.

2. LITERATURE SURVEY

Machine Learning Algorithm for Automated Network Application Identification:

The identity of network software that makes site visitors flows is important to the network control and surveillance.

Current famous technique together with a port number and payload-primarily based totally identity are in good enough and show off some of shortfalls. A capacity answer is using machine mastering method to pick out community programs primarily based totally on payload impartial statistical features.
We evaluate and compare the efficiency and performance of different feature selection and machine learning technique. Based on the flow data obtained from the traffic traces. Classification refers to the identification of an application or group of application responsible for a traffic flow. Port based classification is still widely practiced despite being only moderately accurate at based. It I expected to become less effective in the near future due to an ever-increasing number of network applications, extensive use of network address translation (NAT), dynamic port allocation and end user deliberately choosing non-default ports.

Neural Network for Internet of Traffic Classification:

Internet Traffic Identification is an crucial device for community management, it lets in operation to higher expect destiny traffic matrices and needs protection to locate anomalous behavior, and researcher to expand more practical site visitors model.

Traffic classifier can achieve better accuracy throughout various software kinds with none supply or destination host-address or port information. We use supervised gadget gaining knowledge of primarily based totally on a Bayesian educated neural network.

By imparting class with out get right of entry to to the content material of the packet. This method gives wider software then approach that require complete packets/payloads for category.

Traffic Classification Using Semi Supervised Approach:

This traffic type method makes use of best waft statistic in classifying the traffic. Particularly,

semi-supervised technique which allows in classifier as to be designed from education records involving of only some categorised and plenty of unlabelled flows.

This method includes steps, clustering and class. Classification partition the schooling dataset into disjoin groups (clusters) after making cluster class is preformed wherein categorized information are used for assigning elegance label to the cluster. The checking out end result are then as compared with SVM primarily based totally classifiers.

Classification of Peer-to-Peer Traffic Using Incremental Neural Network (Fuzzy ARTMAP):

In Fuzzy ARTMAP Neural Network, class of peer-to-peer (P2P) visitors in IP networks. We seize net visitors at the primary gateway router, completed preprocessing at the data, pick the maximum giant attributes, and prepare a schooling dataset to attain the Fuzzy ARTMAP set of rules in which applied.

Fuzzy ARTMAP is really an incremental getting to know classifier appropriately in mining the movement of data. The technique is based only at the IP header of the packets.

Finger print user behind NAT from Net-flow records alone:

It is typically identified that the network site visitors generated through an person act as his biometric signature. The predominant contribution is for providing the primary answer for fingerprinting person hidden in the back of a NAT router while most effective Net--waft statistics are available.

The predominant declare on this paper is that there may be no powerful method to fingerprint character hidden from NAT while simplest Net-waft data are available. This method works simplest while the attacker is bodily near the goal and might eavesdrop the wi-fi traffic.

Detecting Anomalous Traffic in Backbone:

With steadily increasing of community scale and the complexity of the community topological shape tracking the site visitors behavior to make sure its availability and operability is turning into a difficult task.

In a while there are important two technique for anomaly detection. Signature primarily based totally and profile primarily based totally, in signature-primarily based totally technique via way of means of the use of the understanding of recognized community attacks, classification-primarily based totally version is then created to decide whether or not or now no longer the incoming occasion is anomalous.

The approached and sensible intrusion detection set of rules to enforce function choice and choice rule era via way of means of the usage of SVM, decision tree and simulated Annealing.

Online Classification of User Activities Using Machine Learning on Network Traffic:

A new hybrid gadget to categorise community site visitors right to an easy set of consumer roles, primarily it's based totally at the community conduct exhibited through the underlying application. These site visitors instructions are used in QoS with differentiation and consumer profiling.

Unlike in the more growth of preceding studies, that type isn't always accomplished for the unique visitors that flows individually; where instead all of the visitors that's generated through a person is then chosen for the consideration. In addition, the 3 window-primarily is based totally technique makes use of all packets, not like the maximum common strategies primarily based totally on simplest the primary packets in every flow.
The first-class mixture of models has been K-Means for the primary class layers, and a Random Forest in the ultimate layer.

3. PROPOSED METHODOLOGY

We have thus used variety machine learning technique to classify the users' activities over the network.
The different machine learning techniques which has also used in the classification are as follows:

1. **Logistic Regression Model**
   Logistic regression model is a statistical analysis method used for predicting a binary outcome, as yes or no, according to the point of observation of a dataset. Logistic Regression model, it predicts the dependent variable thus by analyzing the relationship in one or more existing independent variables.
   We have example, logistic regression can be used in predicting whether a political candidate will be able to win or lose an election or whether a student will be able to admit or not in a particular college and these binary outcomes allows a straight-forward decisions between the two alternatives.

The clarification of logistic regression is started with a proof of the same old logistic feature. The logistic feature is a sigmoid feature, which takes any actual enter, and outputs a fee among 0 and one. For the logic, that is translated by taking enter log-odds with having output probability. The popular logistic feature is described as follows:

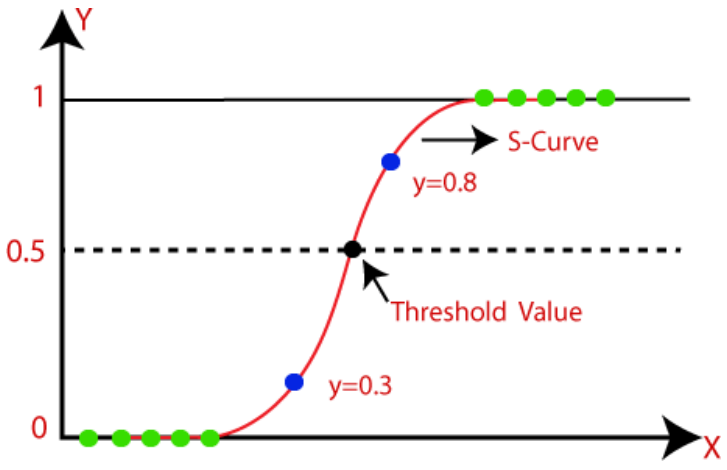


Figure: (i). Logistic Function

2. Random Forest

It is a famous machine learning set of rules and algorithm which belongs with the supervised learning technique. It may be implemented for each classification and in the Regression solving problems in ML. It is primarily based on totally at the idea of ensemble mastering, that's a method of mixing more than one classifier to clear up a complicated hassle as well as to enhance in overall performance within the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset

." Where instead of counting in one decision tree, the Random Forest insist in predicting from every tree and is primarily based totally on the bulk votes of the 12 predictions, and also it thus predicts the very last output. The extra wide variety of a trees in the wooded area results in better accuracy and forestalls the trouble of overfitting.

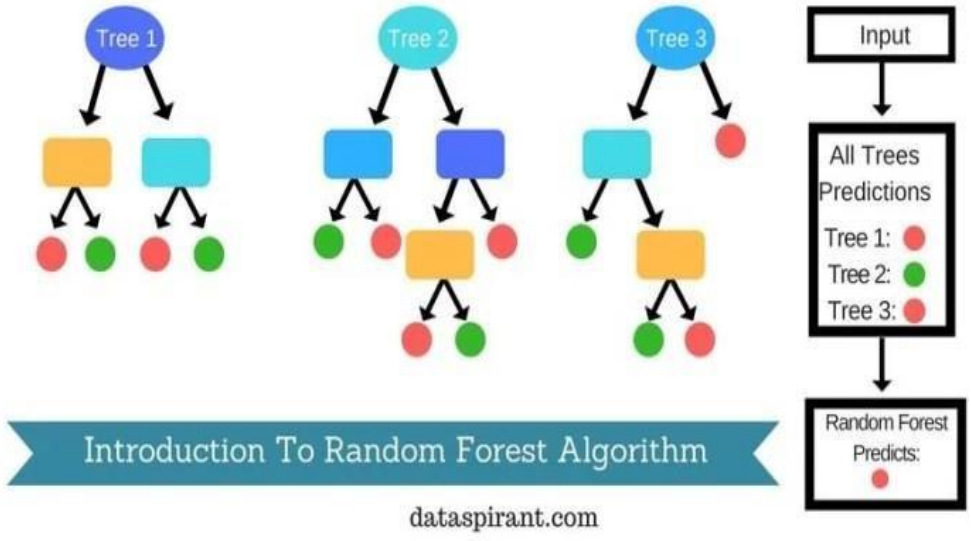


Figure (ii). Algorithm of Random Forest

3. K-Nearest Neighbour (K-NN)

- K-Nearest Neighbour is the only Machine Learning algorithms that's primarily referred totally with Supervised Learning technique
- K-NN algorithm, it takes in similarity among the brand new case/facts and to be had instances and positioned the brand new case into the class this is maximum just like the to be had categories.
- K-NN algorithm shops all of the to be had facts and classifies a brand new facts factor primarily based totally at the similarity. This approach whilst new facts seems then it is able to be without problems categorised right into a properly suite class with the aid of using the usage of K- NN algorithm.
- K-NN algorithm may be used for Regression in addition to for Classification however mainly it's for the used for the Classification problems.

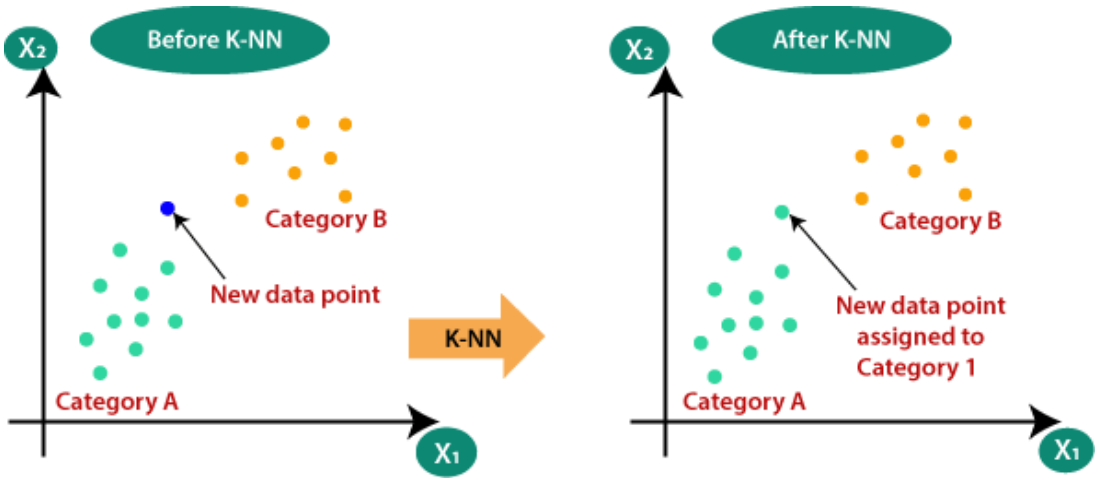


Figure (iii). Working of K-NN (K-Nearest Neighbour)

4. Confusion Matrix

Confusion matrix, it's an N x N matrix applied for solving the overall performance of a classification model, wherein N is the variety of goal classes.

In matrix it compares the real goal values with the ones predicted through the machine learning model where it offers a holistic view of how nicely the classification model is appearing and what forms of mistakes it is making.

In the binary type of classification problem, we'd take a 2 x 2 matrix for proven beneath with four values:

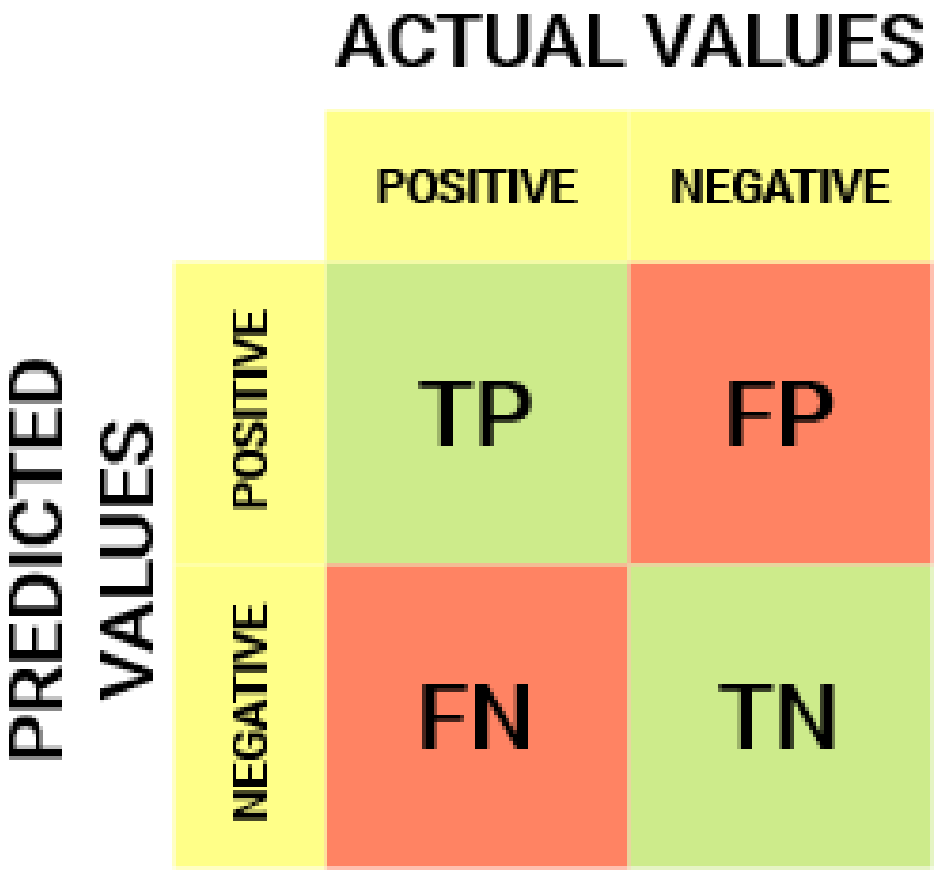


Figure (iv). Confusion Matrix of 2*2

4. DESIGN & Flowchart

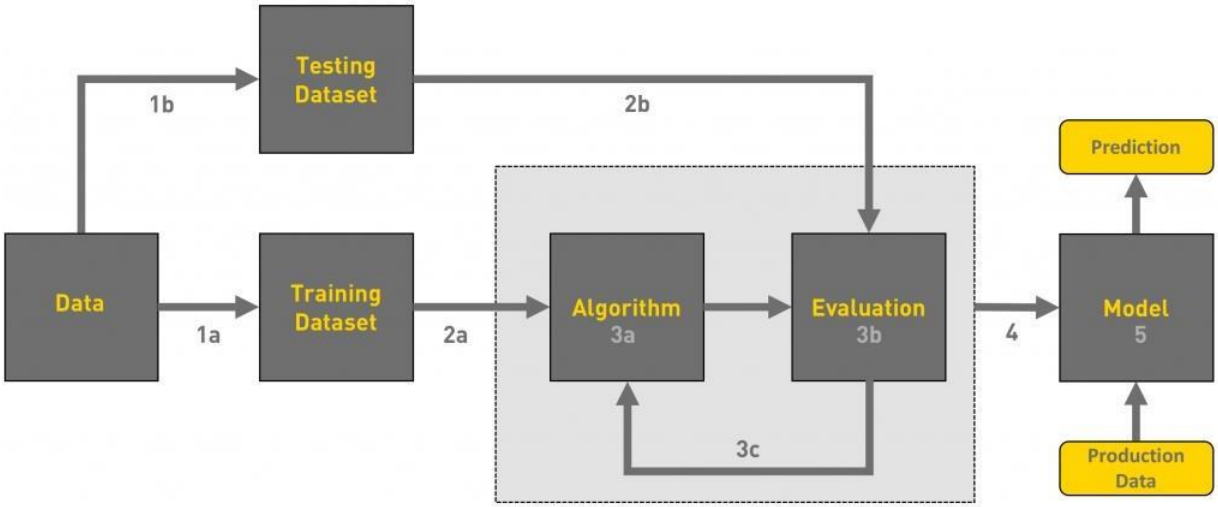


Figure (v): Diagram for classification phases and workflow

We will also go over data pre-processing, data cleaning, feature exploration and feature engineering and show the impact that it has on Machine Learning Model Performance. We will also cover a couple of the pre-modelling steps that can help to improve the model performance.
Python Libraries that would be need to achieve the task:

1. Numpy
2. Pandas
3. Sci-kit Learn
4. Matplotlib

5. IMPLEMENTATION

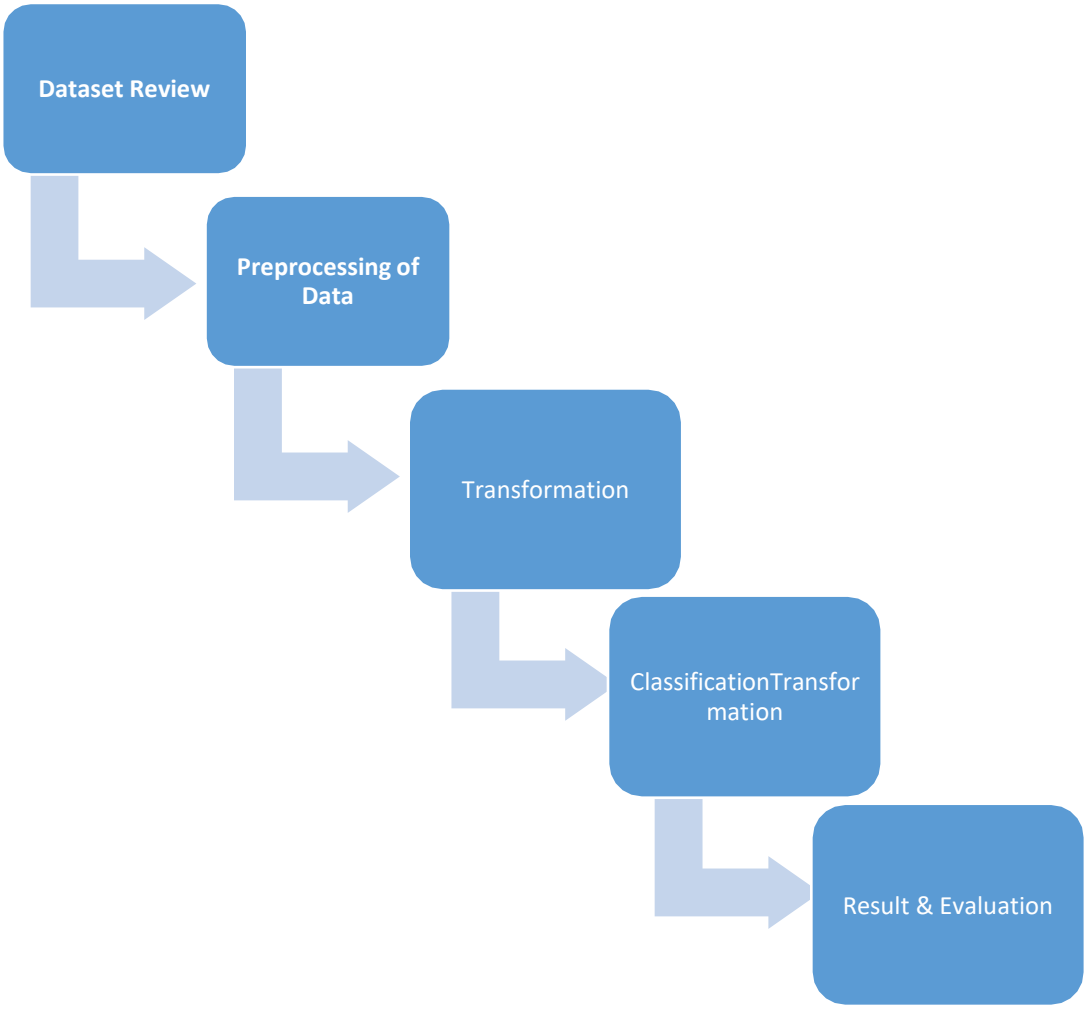
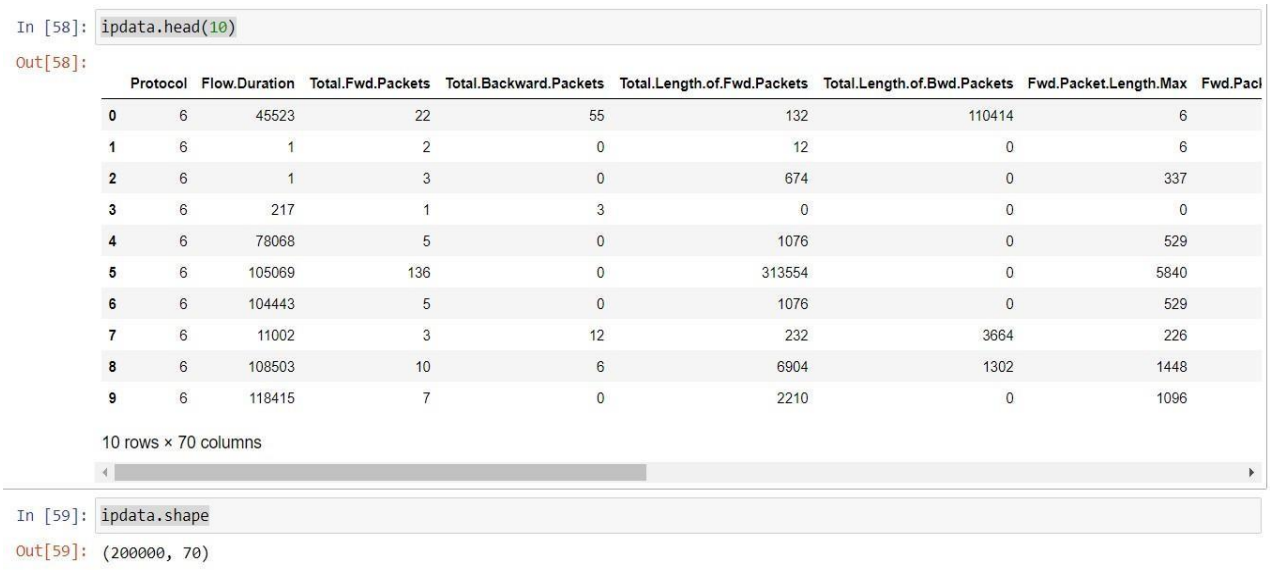


Figure (vi): Flowchart

Dataset Review



In [58]:
```
ipdata.head(10)
```

Out[58]:

	Protocol	Flow.Duration	Total.Fwd.Packets	Total.Backward.Packets	Total.Length.of.Fwd.Packets	Total.Length.of.Bwd.Packets	Fwd.Packet.Length.Max	Fwd.Pacl
0	6	45523	22	55	132	110414	6	
1	6	1	2	0	12	0	6	
2	6	1	3	0	674	0	337	
3	6	217	1	3	0	0	0	
4	6	78068	5	0	1076	0	529	
5	6	105069	136	0	313554	0	5840	
6	6	104443	5	0	1076	0	529	
7	6	11002	3	12	232	3664	226	
8	6	108503	10	6	6904	1302	1448	
9	6	118415	7	0	2210	0	1096	

10 rows × 70 columns

In [59]:
```
ipdata.shape
```

Out[59]:
```
(200000, 70)
```

Figure (vii): Snap shoot of a dataset

Our dataset consists of 70 features and 200000 rows. It has 6 classes.

Data Preprocessing

Data preprocessing is a step in a data analysis process that takes raw data and transform it into a format that can be understood and analyzed by computer and machine learning models.
Raw data is messy and it may contain error and inconsistencies, but it often incomplete and does not have a regular uniform design.
Through preprocessing of the dataset, we remove the nan values from the dataset. In order to ensure and enhance the performance data preprocessing is required.

Data pre-processing is needed to explain in the network traffic along with a fixed of consultant features.
Data is supplied with many strains to a single-person and an activity consistent with trace; to one flow of packet they're joined.

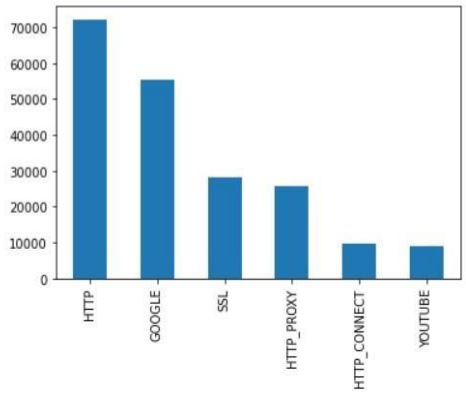
Graphical View of the Classes:

```
In [35]: print(sorted(Counter(df['ProtocolName']).items()))

[('GOOGLE', 55341), ('HTTP', 72199), ('HTTP_CONNECT', 9648), ('HTTP_PROXY', 25765), ('SSL', 28016), ('YOUTUBE', 9031)]
```

```
In [36]: top =df['ProtocolName'].value_counts()
top.sort_values(ascending = False)
top = top[ : 6]
top.plot(kind = "bar")

Out[36]: <AxesSubplot:>
```



Since the classes are not balance, we have to balance the dataset by applying Under-Sampling method or over sampling method.

Under- Sampling:

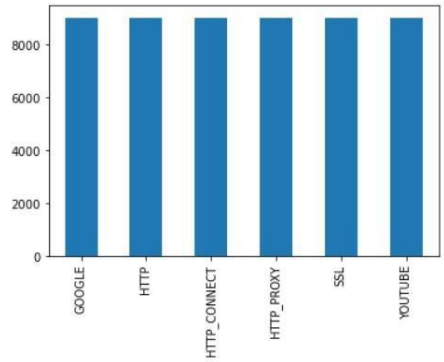
```
In [37]: X = df
         Y = df['ProtocolName']

In [38]: from imblearn.under_sampling import RandomUnderSampler
         rus = RandomUnderSampler(random_state = 0)
         X_resampled,y_resampled = rus.fit_resample(X, Y)
         print(sorted(Counter(y_resampled).items()), y_resampled.shape)

         [('GOOGLE', 9031), ('HTTP', 9031), ('HTTP_CONNECT', 9031), ('HTTP_PROXY', 9031), ('SSL', 9031), ('YOUTUBE', 9031)] (54186,)

In [39]: top = y_resampled.value_counts()
         top.sort_values(ascending = False)
         top = top[ : 6]
         top.plot(kind = "bar")

Out[39]: <AxesSubplot:>
```



If merging is required, all traces are converted to a familiar starting time, enabling captures taken on different days to also be merged.

A window filtering operation is carried out to remove from the dataset any window with no massive quantity of traffic.

In data preprocessing we have preprocessed the data sample, we have removed the Nan values present in the dataset, we have also removed the redundant data present in a dataset.

REPLACE THE MISSING VALUES

```
In [16]: from sklearn.impute import SimpleImputer
         imputer = SimpleImputer(missing_values='NAN',strategy= 'constant', fill_value="dataset")
         imputer.fit(dataset)

Out[16]: SimpleImputer(fill_value='dataset', missing_values='NAN', strategy='constant')
```

Simple imputer, it's a *sklearn* class which is needed for handling the missing data with a predicted model dataset. Imputer replaces the NaN value with a specific place folder *fill_value*. The constant value to be given to the NaN data using constant strategy.

In imputation, it preserves all the cases through replacing the missing data along with how much it's estimated value which done to different needed information at once when the missing value's being imputed so the dataset could be solved by using a standard technique for the rest of the data.

Data Capturing

The seized process is then solved by using T-shark, Linux CLI Tool, the capturing is thus performed by utilizing a network probe that's attached with the router that which forwards the network traffic to the user.
With a Switched Port Analyzer session need should have to be enabled within the router in order to gain a duplicate of a traffic on the network probe. All of non-TCP or UDP packets are filtered out, in every packet with no payload, on account that they're now no longer significant for the said classification.

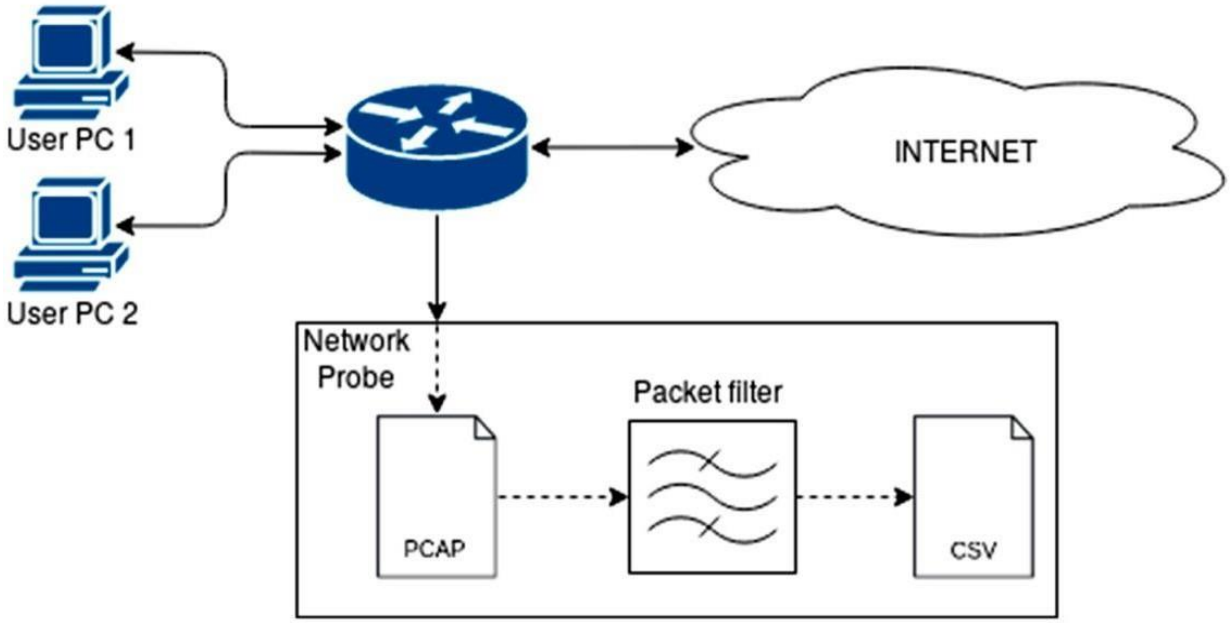


Figure (viii): Capture diagram for the main dataset

6. RESULT AND ANALYSIS

The Data Is Then Splitted For Training and Testing:

```
In [8]: from sklearn.model_selection import train_test_split
        x_train, x_test = train_test_split(dataset, test_size=0.2, random_state=0)
        print(f"Rows in train set: {len(x_train)}\nRows in test set:{len(x_test)}\n")

        Rows in train set: 838860
        Rows in test set:209715
```

We have imported the "*train_test_split"* function through the sklearn library.

The "*train_test split"* is thus a function in a "sklearn model_selection" to splitt data array into two subsets in training the data and the testing data. From this function we do not split the dataset manually by default "*sklearn train_test_split"* will be making a random partition for two subsets.

i). Then data has now been splitted for training and testing the data.
ii). The test data consists of 20 precent and the remaining 80 percent is for training part
iii). The training data has been kept high to so that we can train the model effectively.

Logistic Regression

```
In [65]: from sklearn.linear_model import LogisticRegression
         classifier = LogisticRegression(random_state = 0)
         classifier.fit(X_train, y_train)
```

In [67]:
```
from sklearn.metrics import confusion_matrix, accuracy_score, classification_report
cm = confusion_matrix(y_test, y_pred)
print("confusion Matrix : \n", cm)
```

```
confusion Matrix :
 [[10789     0   166    75     0   109]
 [    0 14612     0     0     1     0]
 [  162     0   973   732     0    10]
 [   79     0   171  4902     1    13]
 [    1     1     0     0  5403     1]
 [ 1429     0    30     2     2   336]]
```

In [68]:
```
from sklearn.metrics import accuracy_score
print("Accuracy : ", accuracy_score( y_pred,y_test))
print(classification_report(y_test,y_pred))
```

```
Accuracy :  0.925375
              precision    recall  f1-score   support

           0       0.87      0.97      0.91     11139
           1       1.00      1.00      1.00     14613
           2       0.73      0.52      0.60      1877
           3       0.86      0.95      0.90      5166
           4       1.00      1.00      1.00      5406
           5       0.72      0.19      0.30      1799

    accuracy                           0.93     40000
   macro avg       0.86      0.77      0.79     40000
weighted avg       0.92      0.93      0.91     40000
```

Random Forest

In [70]:
```
from sklearn.decomposition import PCA
pca = PCA(n_components = 1 )
X_train = pca.fit_transform(X_train)
X_test = pca.transform(X_test)
```

In [71]:
```
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators = 10, criterion= 'entropy', random_state= 0)
classifier.fit(X_train, y_train)
```

In [73]:
```
from sklearn.metrics import confusion_matrix, accuracy_score, classification_report
cm = confusion_matrix(y_test, y_pred)
print("confusion Matrix : \n", cm)
```

```
confusion Matrix :
 [[ 5905  1283   499  1508  1369   575]
 [ 1403 11207   308   671   801   223]
 [  558   302   375   251   303    88]
 [ 1742   659   281  1585   669   230]
 [ 1495   819   258   664  1909   261]
 [  655   291   103   260   293   197]]
```

In [74]:
```
from sklearn.metrics import accuracy_score
print("Accuracy : ", accuracy_score( y_pred,y_test))
print(classification_report(y_test,y_pred))
```

```
Accuracy :  0.52945
              precision    recall  f1-score   support

           0       0.50      0.53      0.52     11139
           1       0.77      0.77      0.77     14613
           2       0.21      0.20      0.20      1877
           3       0.32      0.31      0.31      5166
           4       0.36      0.35      0.36      5406
           5       0.13      0.11      0.12      1799

    accuracy                           0.53     40000
   macro avg       0.38      0.38      0.38     40000
weighted avg       0.53      0.53      0.53     40000
```

K-Nearest Neighbour (K-NN)

In [76]:
```
from sklearn.decomposition import PCA
pca = PCA(n_components = 1 )
X_train = pca.fit_transform(X_train)
X_test = pca.transform(X_test)
```

In [77]:
```
from sklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p = 2)
classifier.fit(X_train, y_train)
```

In [79]:
```
cm=confusion_matrix(y_test,y_pred)
print("confusion Matrix : \n", cm)
```

```
confusion Matrix :
 [[ 7512  1154   261  1168   868   176]
 [ 1923 11334   169   491   604    92]
 [  817   322   328   188   194    28]
 [ 2383   577   171  1540   419    76]
 [ 2171   769   171   493  1715    87]
 [  924   294    61   190   220   110]]
```

In [80]:
```
from sklearn.metrics import accuracy_score
print("Accuracy : ", accuracy_score( y_pred,y_test))
print(classification_report(y_test,y_pred))
```

```
Accuracy :  0.563475
              precision    recall  f1-score   support

           0       0.48      0.67      0.56     11139
           1       0.78      0.78      0.78     14613
           2       0.28      0.17      0.22      1877
           3       0.38      0.30      0.33      5166
           4       0.43      0.32      0.36      5406
           5       0.19      0.06      0.09      1799

    accuracy                           0.56     40000
   macro avg       0.42      0.38      0.39     40000
weighted avg       0.55      0.56      0.55     40000
```

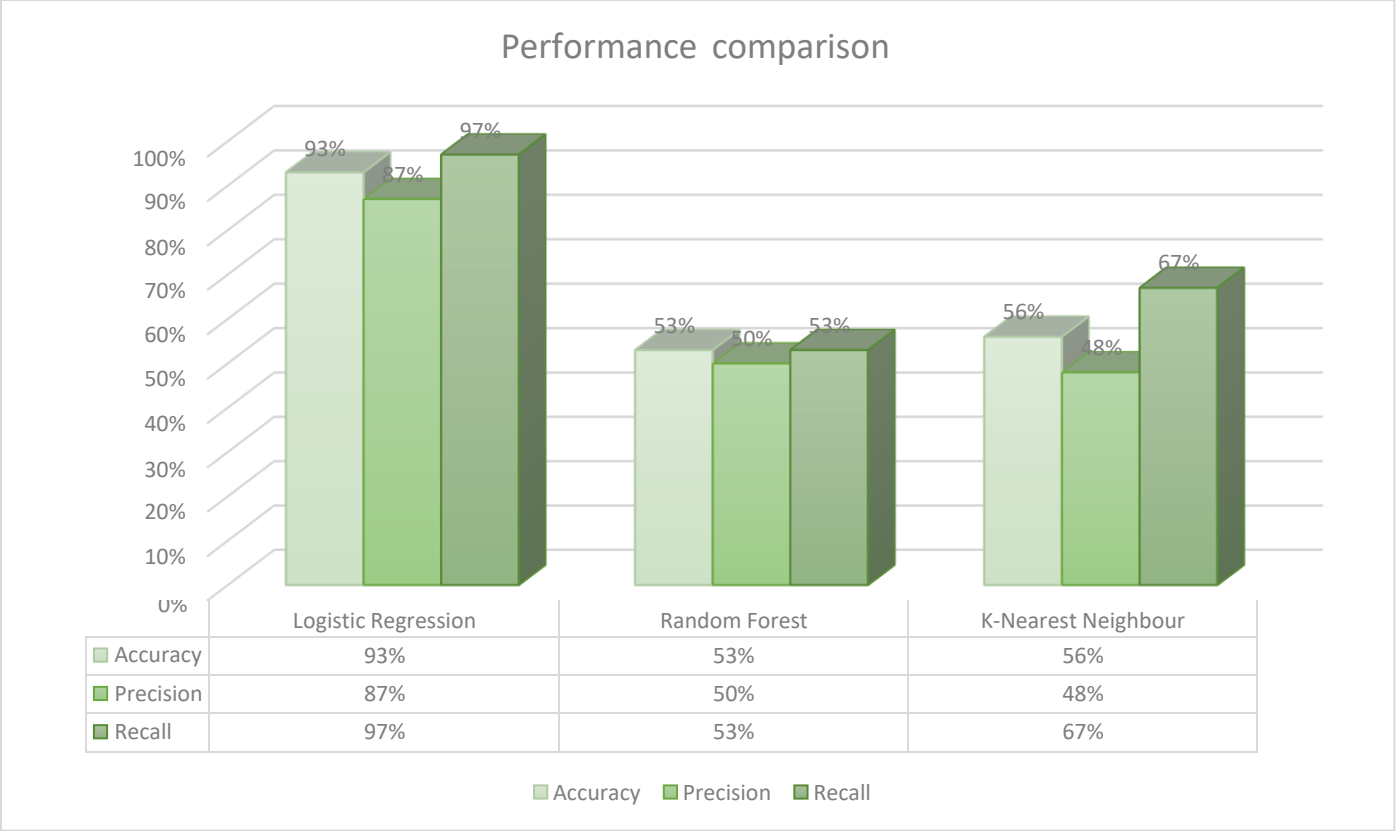


Figure: Comparison for 3 Models (i.e., Logistic Regression, Random Forest, K-Nearest Neighbour (K-NN))

7. SCOPE OF THE PROJECT

People use a variety of application while browsing the page of internet. It is very crucial for internet service provider (ISP) to keep an eye on the network traffic.

It will help the Internet Service Provider (ISP) to provide the bandwidth to the user according to their requirement dynamically. For example, if a user is doing a bulk data transfer and the other user is just browsing the webpage, the user who is doing the bulk data transfer will need a higher bandwidth as compared to the other user who is just browsing the page. So, the ISP should be able to provide the bulk data transfer user with a higher bandwidth as compared to the web browsing user dynamically.

It will enhance user's experience and provide the Quality of Services (QoS) to a user on a network.

It will help to maintain network security in a network since, the ISP will be able to track and examine every user over the network so they can easily distinguish the users who perform malicious activities over a network.

8. CONCLUSION

The review dataset is trained using three different models and while evaluating the performance based on the confusion matrix. Logistic Regression has the highest accuracy of 92% while K-NN and Random Forest have the same accuracy level.

Therefore, based on all the existing parameters from the confusion matrix Logistic Regression is the best suited for our review dataset.

We took the existing algorithm and tried to enhance the accuracy with different pre-processing techniques especially using the K-Fold Cross validation it greatly optimised the performance of the model.

9. REFFERENCE

- http://localhost:8888/notebooks/Untitled%20Folder/Untitled.ipynb
- https://data.mendeley.com/datasets/5pmnkshffm/3
- https://doi.org/10.1016/j.comnet.2020.107557
- https://scikit-learn.org/stable/modules/generated/sklearn.impute.SimpleImputer.html?highlight=imputer#sklearn.impute.SimpleImputer

10. BIBLIOGRAPHY

- https://www.semanticscholar.org/paper/Literature-Review-of-Network-Traffic-Classification-Bin-Ru/1cc4a62e143c0275f3a95d8a913fd8f6dda1cdaa
- https://www.researchgate.net/publication/316900016_Network_Traffic_Classification_techniques_and_comparative_analysis_using_Machine_Learning_algorithms

Final_Year_Project_report2.pdf

ORIGINALITY REPORT

8%	%	8%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Víctor Labayen, Eduardo Magaña, Daniel Morató, Mikel Izal. "Online classification of user activities using machine learning on network traffic", Computer Networks, 2020 Publication	5%
2	Marchang, N.. "Collaborative techniques for intrusion detection in mobile ad-hoc networks", Ad Hoc Networks, 200806 Publication	2%
3	Pramod Gupta, Naresh K. Sehgal. "Chapter 2 Machine Learning Algorithms", Springer Science and Business Media LLC, 2021 Publication	1%
4	Bijan Raahemi, Alexandre Kouznetsov, Ahmad Hayajneh, Peter Rabinovitch. "Classification of Peer-to-Peer traffic using incremental neural networks (Fuzzy ARTMAP)", 2008 Canadian Conference on Electrical and Computer Engineering, 2008 Publication	1%