JSS Mahavidyapeetha

JSS SCIENCE & TECHNOLOGY UNIVERSITY

(Sri Jayachamarajendra College of Engineering)
MYSURU

INTERNET TRAFFIC CLASSIFICATION

PROJECT REPORT

Submitted To:

Prof. Manju N

Department of Information science
JSS S&TU, Mysuru

SUBMITTED BY

NAGENDRA M	**USN :01JST18IS027**
SHREYAS GS	**USN:01JST18IS044**
CHARAN S DEVARAJ	**USN:01JST18IS058**
SHARATH M	**USN:01JST18IS066**

3rd YEAR

1. ABSTRACT

Accurate traffic classification is of fundamental importance to numerous other network activities, from security monitoring to accounting, and from Quality of Service to providing operators with useful forecasts for long-term provisioning. We apply a KNN, Random forest, Decision tree and SVM to categorize traffic by application. Uniquely, our work capitalizes on pre-classified network data, using it as input to a supervised algorithms. In this report, we illustrate the high level of accuracy achievable with the KNN and random forest. We further illustrate the improved accuracy obtained from these algorithms where we have used feature importance property of a tree-based classifier and correlation-based selection to increase the speed and performance of a classifier.
We were able to achieve an accuracy of 99.04 percent by using random forest with feature selection and without any feature selection we got an accuracy of 98.37 percent.

2. INTRODUCTION

Network Traffic Classification is an automated process, which categorizes computer network traffic according to various parameters. It is an important part of the current network monitoring system, its task is to infer the network service (e.g. HTTP, SIP . . .) that is being used. NTC (Network Traffic Classifier) helps to collect information about a current network flow just by knowing its network service. This information is important for network management and Quality of Service (QoS) [1,2], as the service user has a direct relationship with QoS requirements and user expectations. NTC helps to detect dissimilar device with different user profile. Yet, classification schemes are difficult to operate correctly because the knowledge commonly available to the network, i.e. packet-headers, often does not contain sufficient information to allow for an accurate methodology.

Network traffic identification is crucial for implementing effective management of network policy and resources. There are several approaches to NTC: port-based, payload-based, and flow 2 statistics-based. Port-based methods make use of port information for service identification. Payload-based approaches the problem by Deep Packet Inspection (DPI) of the payload carried out by the communication flow. Flow statistics-based methods rely on information that can be obtained from packets

header (e.g. bytes transmitted, packets interarrival times, TCP window size) [12]. The first two methods have more drawbacks compared to the latter one.

CHALLENGES:

KNN:

- Always needs to determine the value of K which may be complex some time.
- The computation cost is high because of calculating the distance between the data points for all the training samples.

DECISION TREE:

- Over fitting the data: Definition: given a hypothesis space H, a hypothesis is said to overfit the training data if there exists some alternative hypothesis.
- Guarding against bad attribute choices.
- Handling continuous valued attributes.
- Handling missing attributes values.
- Handling attributes with differing costs.

RANDOM FOREST:

- Model interpretability: **Random forest** models are not all that interpretable; they are like black boxes.
- For very large data sets, the size of the trees can take up a lot of memory.
- It can tend to overfit, so you should tune the hyper parameters.

SVM:

- SVM algorithm is not suitable for large data sets.
- SVM does not perform very well when the data set has more noise i.e. target classes are overlapping.
- In cases where the number of features for each data point exceeds the number of training data samples, the SVM will underperform.

We applied Machine Learning (ML) algorithms, utilising the statistical properties of the network traffic flow. We try to compare the performance in generalized manner

by applying algorithms like KNN, DECISIONTREE AND RANDOM FOREST [3,5,6].

3. LITERATURE SURVEY

In this paper we highlighting the previous and current state of art work in the domain of internet traffic classification

Due to its fundamental nature and its underpinning of many other techniques, the field of traffic classification has maintained continuous interest. For example, the most common technique for the identification of Internet network applications through traffic monitoring relies on the use of well known ports: an analysis of the headers of packets is used to identify traffic associated with a particular port and thus of a particular application [7, 8]. It is well known that such a process is likely to lead to inaccurate estimates of the amount of traffic carried by different applications given that specific protocols, such as HTTP, are frequently used to relay other types of traffic, e.g. a VLAN over HTTP. In addition, emerging services avoid the use of well-known ports altogether, e.g. some peer-to-peer applications. Our work is presented in the light of these traditional classification techniques diminishing in effectiveness. This report is not the forum for a survey of the entire Machine Learning field. However, our approach may be contrasted with previous work which attempts the classification of network traffic [9][10][11]. Even today the use of hierarchical model represented as a dendrogram with each object representing the transfer of communications data between applications is used. A classification scheme based upon this representation allows them to describe representative traffic patterns for input to the simulation work but does not intend to allow identification of the traffic itself. The field of security also uses ML techniques. However, they are attempting to do something fundamentally different from us where we are attempting to categories all traffic, intrusion-detection and related elds attempt to perform signature matching in order to identify intrusion or malicious software by it's peculiar behaviour among the continuum of otherwise legitimate activity(basically detecting outliers). But here in this report we are trying to categorize all the labels for future application purpose in the field of IOT (Internet Of Things), hence the overhead to correctly label the test data is huge.

Here our main aim is to use KNN [6], Decision tree [3, 4], Random Forest [5] and SVM algorithms of Machine Learning Techniques for Internet Traffic Classification.

4. IMPLEMENTATION

4.1 DATASET:

We have used data collected from Kaggle platform Network Traffic Flows Labelled with 75 Apps [13]. This dataset contains 83 features. Each instance holds the information of an IP flow generated by a network device i.e., source and destination IP addresses, ports, interarrival times, layer 7 protocol (application) used on that flow as the class, among others. Most of the attributes are numeric type but there are also nominal types and a date type due to the Timestamp. The application layer protocol was obtained by performing a DPI (Deep Packet Inspection) processing on the flows with ntopng. The flow statistics (IP addresses, ports, inter-arrival times, etc) were obtained using CICFlowmeter.

We created these data sets as training and testing sets to allow assessment of different ml classification techniques. We have tested classifiers on balanced dataset after converting it from the unbalanced dataset which is shown in the [fig .1].

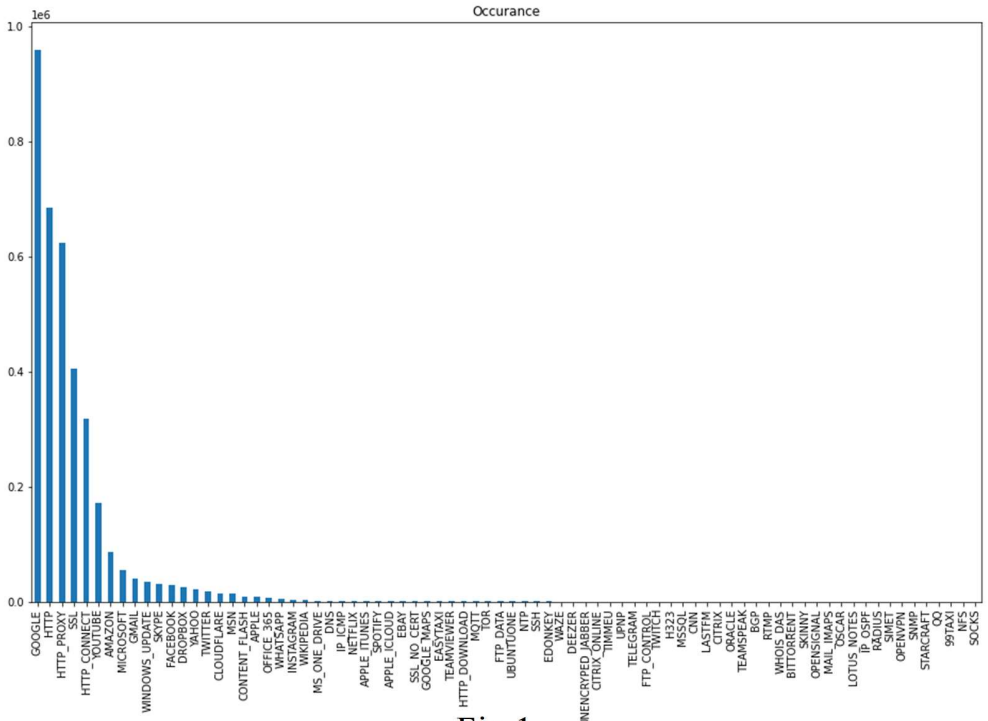


Fig.1

Balanced dataset: Balanced data refers to a situation where the number of observations is the same for all the classes in a classification dataset. We converted the unbalanced dataset to balanced dataset through SMOTE and NEARMISS algorithm. Balanced dataset collected has 3577082 instances which are evenly distributed among 42 classes [fig .2].

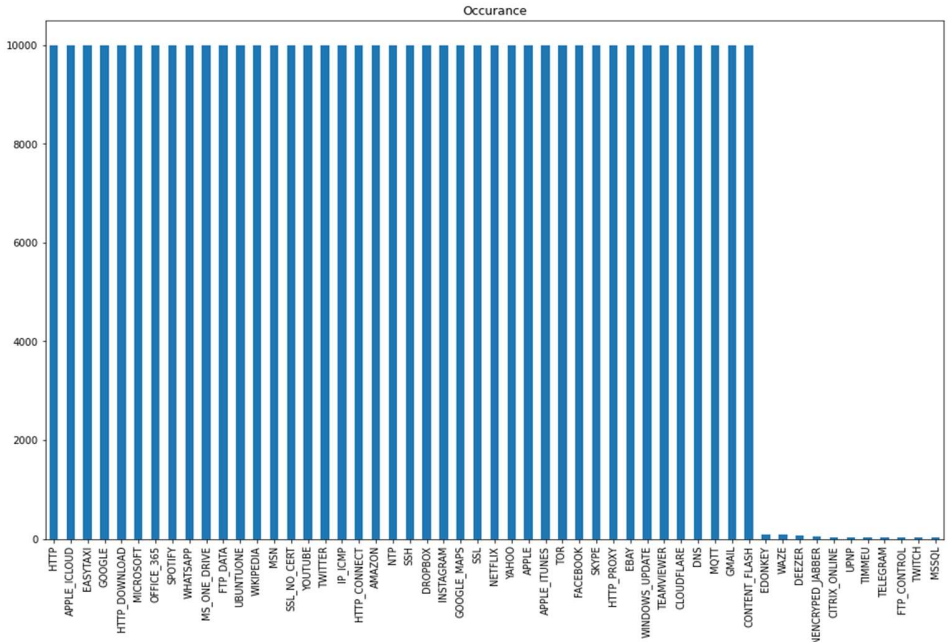


Fig .2

4.2 PRE-PROCESSING:

One of the important stages of machine learning is pre-processing, Imbalance dataset can cause many problems. From fig.1 it can be noted that class GOOGLE, HTTP have a huge instances compared to other classes. This can lead to bias while training the model and leads to incorrect prediction which indicates that our model has been trained more to predict GOOGLE class. So, the dataset should be balanced before training the model. We have used statically methods called Under sampling and Over sampling.

Near Miss Under sampling:

Near Miss refers to a collection of under sampling methods that select examples based on the distance of majority class examples to minority class examples.

Smote Oversampling:

It oversamples the data to desired value by duplicating the values based on the current data instances.

We used above both these techniques to balance our dataset but still you can see in the
fig .2 at the tail of the graph some classes have not been balanced, we kept that intentionally because it didn't have enough features to oversample.

Missing values: This is the step where we prepare the data for training. Data tends to be incomplete, noisy, and inconsistent. Such datasets however are incompatible. A basic strategy to use incomplete datasets is to discard entire rows or columns containing missing values. However, this comes at the price of losing data which may be valuable. However, our dataset had no missing values.
The source IP and destination IP has been converted into number format using the python library (socket & struct).
Label Encoding has been done to our dataset and then converted whole data into floating point integers.
Deleted the columns which had no unique values in the dataset.

Feature scaling: It is a step of Data Pre-processing which is applied to independent variables or features of data. It basically helps to normalise the data within a particular range. Sometimes, it also helps in speeding up the calculations in an algorithm. So, we transformed our data using Standard Scalar method. The transformed data is then used to train algorithms and a clear difference in predicting accuracies is observed wherein the
dataset which is scaled vastly outperforms the unscaled version.
Standardizing the features of dataset means by removing the mean and scaling it into unit variance.
The standard score of a sample x is calculated as:

$z = (x - u) / s$

where u is the mean of the training samples or zero if with_mean=False and s is the standard deviation of the training samples or one if with_std=False.

4.3 FEATURE SELECTION:

It is a most important part in Machine learning process.
From the feature selection we get benefits like:

- increasing the interpretability of the model.
- reducing the complexity of the model.
- reducing the training time of the model.

Feature Importance:

Feature importance gives a score for each feature of the data, the higher the score more important or relevant is the feature towards our output variable which is protocol names. Feature importance is an inbuilt class that comes with Tree Based Classifiers, we used Random Forest Classifier for extracting the top 55 features for this dataset.

Correlation Matrix with Heatmap:

Correlation states how the features are related to each other or the target variable. Correlation can be positive (increase in one value of feature increases the value of the target variable) or negative (increase in one value of feature decreases the value of the target variable). Heatmap makes it easy to identify which features are most related to the target variable, we will plot heatmap of correlated features using the seaborn library.

From these two techniques we found out that top 55 features are enough to achieve accuracy.
Below fig .3 clearly shows that L7protocol,destination IP, source port have high importance among 83 features

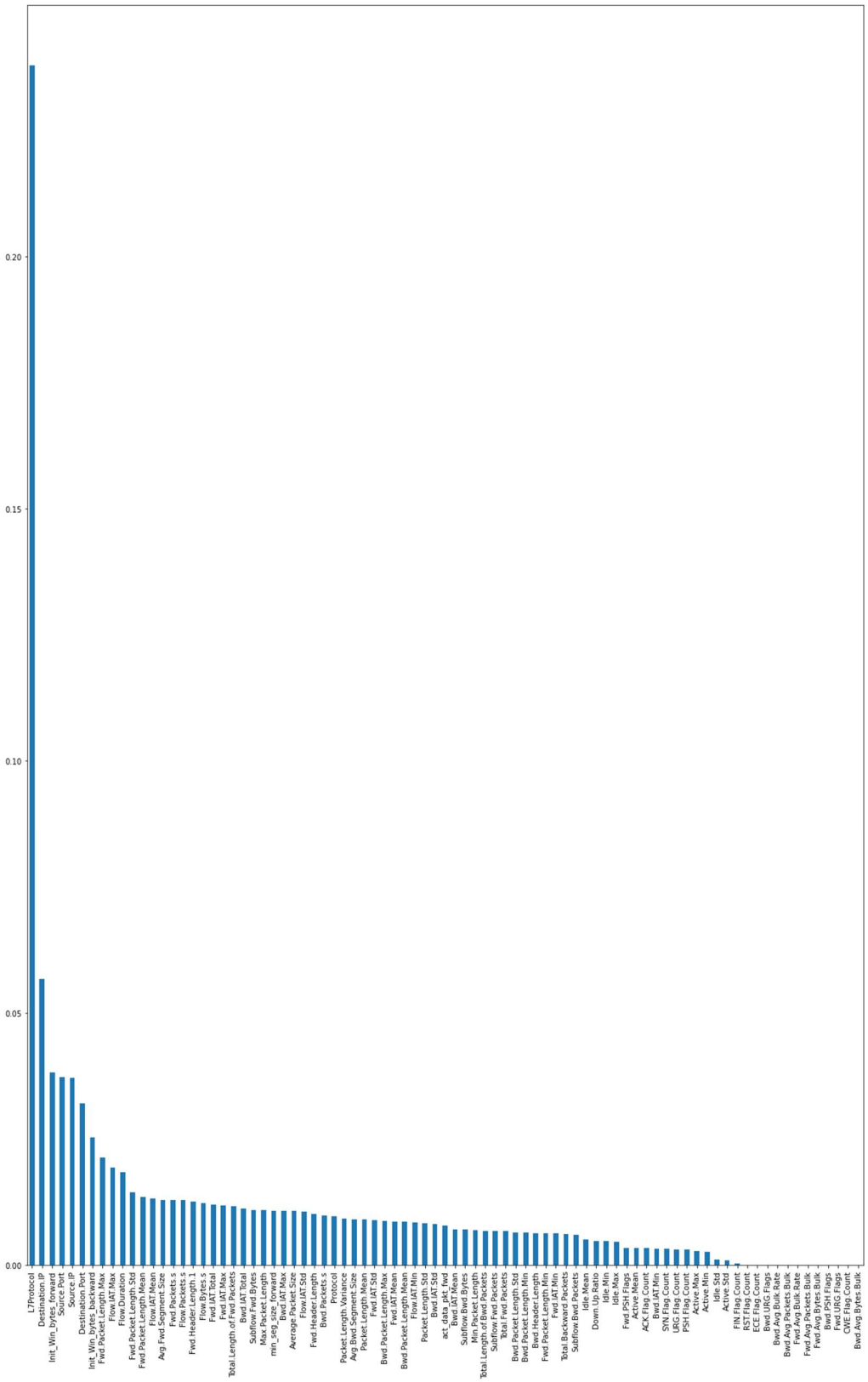
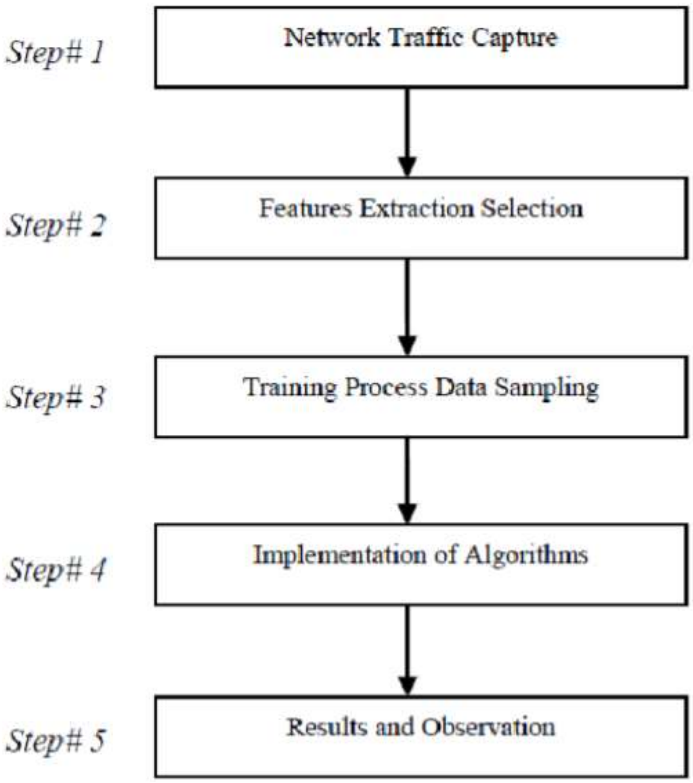

0.20
0.15
0.10
0.05
0.00
L7Protocol
Destination IP
Init_Win_bytes_forward
Source Port
Source IP
Destination Port
Init_Win_bytes_backward
Fwd Packet Length Max
Flow IAT Max
Flow Duration
Fwd Packet Length Std
Fwd Packet Length Mean
Flow IAT Mean
Avg Fwd Segment Size
Fwd Packets/s
Flow Packets/s
Fwd Header Length.1
Flow Bytes/s
Fwd IAT Total
Fwd IAT Max
Total Length of Fwd Packets
Bwd IAT Total
Subflow Fwd Bytes
Max Packet Length
min_seg_size_forward
Bwd IAT Max
Average Packet Size
Flow IAT Std
Fwd Header Length
Bwd Packets/s
Protocol
Packet Length Variance
Avg Bwd Segment Size
Packet Length Mean
Fwd IAT Std
Bwd Packet Length Max
Fwd IAT Mean
Bwd Packet Length Mean
Flow IAT Min
Packet Length Std
Bwd IAT Std
act_data_pkt_fwd
Bwd IAT Mean
Subflow Bwd Bytes
Min Packet Length
Total Length of Bwd Packets
Subflow Fwd Packets
Total Fwd Packets
Bwd Packet Length Std
Bwd Packet Length Min
Bwd Header Length
Fwd Packet Length Min
Fwd IAT Min
Total Backward Packets
Subflow Bwd Packets
Idle Mean
Down Up Ratio
Idle Min
Idle Max
Fwd PSH Flags
Active Mean
ACK Flag Count
Bwd IAT Min
SYN Flag Count
URG Flag Count
PSH Flag Count
Active Max
Active Min
Idle Std
Active Std
FIN Flag Count
RST Flag Count
ECE Flag Count
Bwd URG Flags
Bwd Avg Bulk Rate
Bwd Avg Packets Bulk
Fwd Avg Bulk Rate
Fwd Avg Packets Bulk
Fwd Avg Bytes Bulk
Bwd PSH Flags
Fwd URG Flags
CWE Flag Count
Bwd Avg Bytes Bulk

Fig.3

4.4 PROPOSED MODEL:

Details: We apply different algorithms like KNN, decision tree, Random forest and SVM to categorize traffic by application. Uniquely, our work capitalizes on pre-classified network data, using it as input to a supervised algorithm. In this paper we illustrate the high level of accuracy achievable with the KNN and Random forest. We further illustrate the improved accuracy obtained from these algorithms where we have used feature selection to increase the speed and performance of a classifier. Two algorithms with the best results has been selected and they are KNN, and RANDOM FOREST.



4.5 THEORY:

In this section of the report, we are going to remind the reader about the fundamental concepts of KNN, Decision Tree, Random Forest and SVM.

KNN:

The k-nearest neighbours (KNN) algorithm is a supervised machine learning algorithm that can be used to solve both classification and regression problems.
The KNN algorithm assumes that similar things exist in close proximity. In other words, similar things are near to each other.

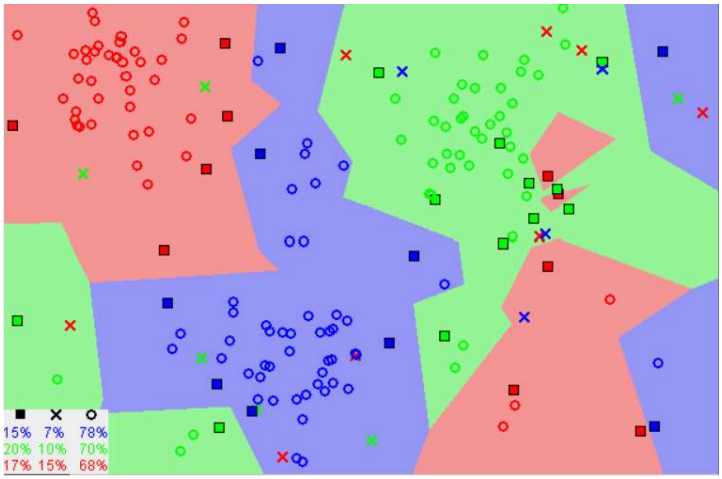


Image showing how similar data points typically exist close to each other

Notice in the image above that most of the time, similar data points are close to each other. The KNN algorithm hinges on this assumption being true enough for the algorithm to be useful. KNN captures the idea of similarity (sometimes called distance, proximity, or closeness) with some mathematics calculating the distance between points on a graph.

There are many methods for calculating distance. In this session we will consider two methods

1. **Euclidean Distance**
2. **Manhattan Distance**

Euclidean Distance:

In mathematics, the **Euclidean distance** between two points in Euclidean space is the length of a line segment between the two points. It can be calculated from the Cartesian coordinates of the points using the Pythagorean theorem, therefore occasionally being called the **Pythagorean distance**.
In the Euclidean plane, let point p have Cartesian coordinates (p1,p2) and let point q have coordinates (q1,q2). Then the distance between p and q is given by:

$$d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2}.$$

Manhattan Distance:

The distance between two points measured along axes at right angles is called Manhattan distance.
In a plane with p_1 at (x_1, y_1) and p_2 at (x_2, y_2)

$d(p_1, p_2) = |x_1 - x_2| + |y_1 - y_2|$

DECISION TREE:

Decision Tree is a type of Supervised Machine Learning algorithm where the data is continuously split according to a certain parameters. The tree can be explained by two entities, namely decision nodes and leaves. The leaves are the decisions or the final outcomes. And the decision nodes are where the data is split.

There are some hyperparameters we need to give correctly to get best accuracy. Some are listed below

1. ***criterion: string, optional (default = " gini"):***

   The function to measure the quality of a split. Supported criteria are "Gini" for the Gini impurity and "entropy" for the information gain.

   There are many ways to implement the impurity measure, two of which scikit-learn has implemented is the Information gain and Gini Impurity or Gini Index. Information gain uses the entropy measure as the impurity measure and splits a node such that it gives the most amount of information gain. Whereas Gini Impurity measures the divergences between the probability distributions of the target attribute's values and splits a node such that it gives the least amount of impurity.

$$Gini: Gini(E) = 1 - \sum_{j=1}^{c} p_j^2$$

$$Entropy: H(E) = -\sum_{j=1}^{c} p_j \log p_j$$

Many of the researchers point out that in most of the cases, the choice of splitting criteria will not make much difference in the tree performance. Each criterion is superior in some cases and inferior in others.

2. ***splitter: string, optional (default="best")***

   It is the strategy used to choose the split at each node. Supported strategies are "best" to choose the best split and "random" to choose the best random split.

3. ***max_depth: int or None, optional (default=None)***

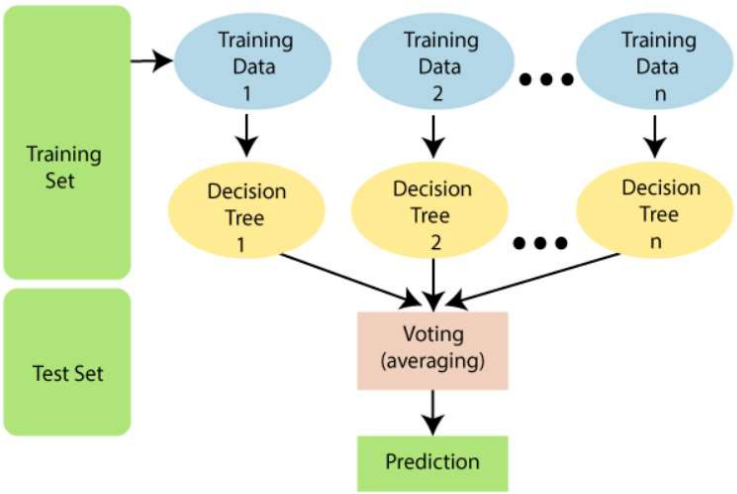
   It is the maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

RANDOM FOREST:

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of 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." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.



The above diagram explains the working of the Random Forest.

Hyperparameters of random forest is similar to that of decision tree.

SVM:

In machine learning, **support-vector machine** is supervised learning models with associated learning algorithms that analyse data for classification and regression analysis.
The objective of the support vector machine algorithm is to find a hyperplane in an N-dimensional space (N — the number of features) that distinctly classifies the data points as shown in the fig 4.

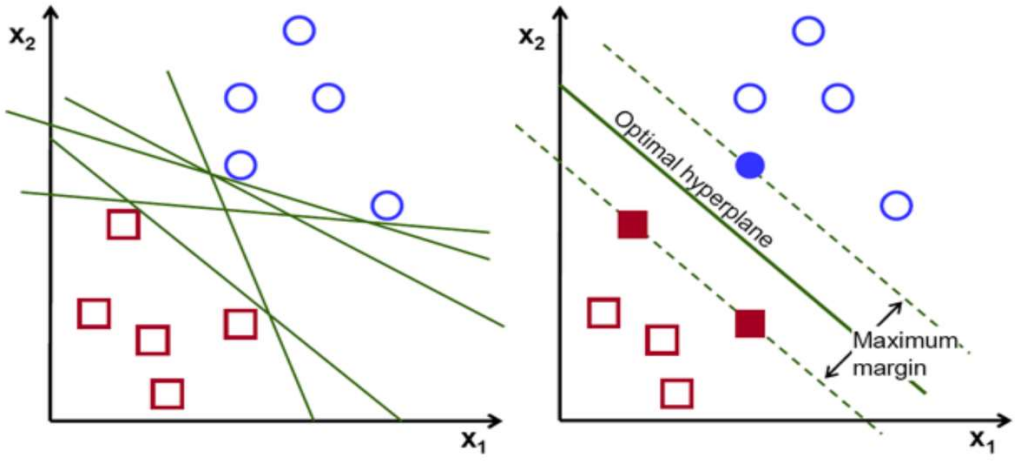


Fig4

RESULT AND DISCUSSION

We applied KNN, Decision Tree, Random Forest and SVM algorithms to our dataset.

The results are as follows.

1. **KNN**:

   We used GridSearchCV for hyperparameter tuning.
   As a result of GridSearchCV hyperparameters we used are:
   - n_neighbours: 3

- metric: Manhattan
- weights: distance

	BEFORE FEATURE SELECTION	AFTER FEATURE SELECTION
Accuracy	94.23%	94.80%
Confusion matrix	[[2361 19 5 ... 22 4 0] [10 2633 107 ... 0 1 0] [1 25 2872 ... 6 0 1] ... [3 3 0 ... 3035 2 0] [6 3 0 ... 3 2592 1] [0 2 0 ... 0 0 2838]]	[[2391 15 3 ... 17 1 0] [7 2651 99 ... 1 1 0] [0 28 2884 ... 1 0 1] ... [1 2 0 ... 3053 1 1] [6 3 1 ... 2 2638 0] [0 0 0 ... 0 0 2881]]

2. **Decision Tree**:

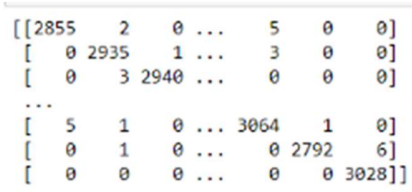
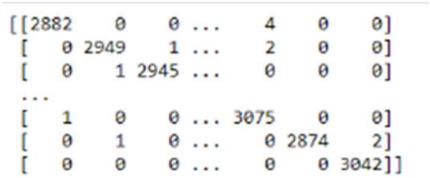
   As a result of GridSearchCV hyperparameters we used are:
   - Criterion: entropy
   - Splitter: best
   - max_depth: 10

   As we apply the algorithm, we got 99.99% accuracy, so it may be the cause of overfitting.
   Overfitting is a significant practical difficulty for decision tree model. Overfitting happens when the learning algorithm continues to develop hypotheses that reduce training set error at the cost of an increased test set error.

   So, we used **Random Forest** over Decision Tree.

3. **Random Forest:**

   As a result of GridSearchCV hyperparameters we used are:
   - Criterion: entropy
   - max_depth: 40
   - n_estimators: 50

	BEFORE FEATURE SELECTION	AFTER FEATURE SELECTION
Accuracy	98.37%	99.04%
Confusion matrix		

Further inspection of the Confusion Matrix will reveal that the model is not biased as the number of False Positive and False Negative is small.
Accuracy calculated using confusion matrix gives idea about the number of correctly predicted classes. It can tell us immediately how well the model has been trained and how it performs generally. But it cannot be considered as our main performance metrics since it does not do well when we have imbalance in the dataset. So, three other metrics Precision, Recall and F1 score is used in order to properly validate the model. It is there in the below classification report.

Precision: Precision is the ratio of correctly predicted positive observations to the total predicted positive observations.
Recall (Sensitivity) - Recall is the ratio of correctly predicted positive observations to the all observations in actual class.
F1 Score: F1 score is the weighted average of Precision and Recall.

Classification Reports:

1. KNN:

	precision	recall	f1-score	support
AMAZON	0.92	0.81	0.86	2958
APPLE	0.94	0.89	0.91	2977
APPLE_ICLOUD	0.93	0.98	0.95	2950
APPLE_ITUNES	0.93	0.96	0.95	3062
CITRIX_ONLINE	1.00	1.00	1.00	11
CLOUDFLARE	0.98	0.97	0.98	3046
CONTENT_FLASH	1.00	1.00	1.00	2950
DEEZER	1.00	0.33	0.50	18
DNS	1.00	0.99	1.00	2990
DROPBOX	0.92	0.89	0.91	3028
EASYTAXI	0.96	0.98	0.97	3083
EBAY	0.93	0.96	0.95	3025
EDONKEY	0.88	0.56	0.68	27
FACEBOOK	0.95	0.95	0.95	2932
FTP_CONTROL	1.00	1.00	1.00	9
FTP_DATA	0.97	0.99	0.98	3008
GMAIL	0.81	0.79	0.80	2966
GOOGLE	0.93	0.98	0.95	2948
GOOGLE_MAPS	0.92	0.97	0.94	3014
HTTP	1.00	1.00	1.00	3012
HTTP_CONNECT	0.96	0.98	0.97	3065
HTTP_DOWNLOAD	0.98	1.00	0.99	3010
HTTP_PROXY	0.96	0.95	0.95	2955
INSTAGRAM	0.97	0.95	0.96	3162
IP_ICMP	1.00	1.00	1.00	3012
MICROSOFT	0.97	0.97	0.97	2926
MQTT	0.99	1.00	0.99	3047
MSN	0.92	0.87	0.89	3015
MSSQL	0.00	0.00	0.00	5
MS_ONE_DRIVE	0.97	0.97	0.97	2979
NETFLIX	0.93	0.95	0.94	2967
NTP	1.00	1.00	1.00	2977
OFFICE_365	0.98	0.98	0.98	2879
SKYPE	0.78	0.80	0.79	2913
SPOTIFY	0.97	0.99	0.98	3025
SSH	0.99	1.00	1.00	2990
SSL	0.96	0.93	0.95	2990
SSL_NO_CERT	0.98	0.98	0.98	2988
TEAMVIEWER	1.00	1.00	1.00	3039
TELEGRAM	1.00	0.73	0.84	11
TIMMEU	1.00	0.09	0.17	11
TOR	0.97	0.99	0.98	3002
TWITCH	0.50	0.14	0.22	7
TWITTER	0.78	0.70	0.74	2994
UBUNTUONE	0.99	1.00	1.00	2937
UNENCRYPED_JABBER	0.92	0.92	0.92	12
UPNP	0.75	0.75	0.75	12
WAZE	0.78	0.78	0.78	23
WHATSAPP	0.94	0.92	0.93	3038
WIKIPEDIA	0.93	0.97	0.95	3087
WINDOWS_UPDATE	0.96	0.99	0.98	3080
YAHOO	0.90	0.90	0.90	2931
YOUTUBE	0.94	0.95	0.94	3048
accuracy			0.95	126151
macro avg	0.92	0.87	0.88	126151
weighted avg	0.95	0.95	0.95	126151

2. Random Forest:

	precision	recall	f1-score	support
AMAZON	0.97	0.97	0.97	2958
APPLE	0.99	0.99	0.99	2977
APPLE_ICLOUD	1.00	1.00	1.00	2950
APPLE_ITUNES	1.00	1.00	1.00	3062
CITRIX_ONLINE	1.00	1.00	1.00	11
CLOUDFLARE	1.00	1.00	1.00	3046
CONTENT_FLASH	1.00	1.00	1.00	2950
DEEZER	1.00	0.44	0.62	18
DNS	1.00	1.00	1.00	2990
DROPBOX	0.99	0.95	0.97	3028
EASYTAXI	0.99	1.00	1.00	3083
EBAY	0.99	0.99	0.99	3025
EDONKEY	1.00	0.78	0.88	27
FACEBOOK	0.98	0.98	0.98	2932
FTP_CONTROL	1.00	1.00	1.00	9
FTP_DATA	0.99	1.00	1.00	3008
GMAIL	0.95	0.98	0.96	2966
GOOGLE	0.99	1.00	1.00	2948
GOOGLE_MAPS	1.00	0.99	1.00	3014
HTTP	0.97	1.00	0.98	3012
HTTP_CONNECT	1.00	1.00	1.00	3065
HTTP_DOWNLOAD	1.00	0.99	1.00	3010
HTTP_PROXY	0.98	1.00	0.99	2955
INSTAGRAM	0.99	0.98	0.99	3162
IP_ICMP	1.00	1.00	1.00	3012
MICROSOFT	0.98	0.99	0.98	2926
MQTT	1.00	1.00	1.00	3047
MSN	0.99	0.98	0.98	3015
MSSQL	1.00	0.20	0.33	5
MS_ONE_DRIVE	1.00	0.99	0.99	2979
NETFLIX	0.99	0.99	0.99	2967
NTP	1.00	1.00	1.00	2977
OFFICE_365	1.00	0.99	1.00	2879
SKYPE	0.99	0.99	0.99	2913
SPOTIFY	1.00	1.00	1.00	3025
SSH	1.00	1.00	1.00	2990
SSL	0.99	0.95	0.97	2990
SSL_NO_CERT	1.00	1.00	1.00	2988
TEAMVIEWER	1.00	1.00	1.00	3039
TELEGRAM	1.00	0.91	0.95	11
TIMMEU	1.00	0.27	0.43	11
TOR	1.00	1.00	1.00	3002
TWITCH	0.00	0.00	0.00	7
TWITTER	0.96	0.96	0.96	2994
UBUNTUONE	1.00	1.00	1.00	2937
UNENCRYPED_JABBER	1.00	0.92	0.96	12
UPNP	1.00	0.92	0.96	12
WAZE	1.00	0.83	0.90	23
WHATSAPP	0.99	0.98	0.99	3038
WIKIPEDIA	0.99	0.99	0.99	3087
WINDOWS_UPDATE	0.99	1.00	1.00	3080
YAHOO	0.96	0.98	0.97	2931
YOUTUBE	0.98	1.00	0.99	3048
accuracy			0.99	126151
macro avg	0.97	0.92	0.94	126151
weighted avg	0.99	0.99	0.99	126151

CONCLUSION:

After applying four algorithms and finding out their accuracies we finally conclude that Random Forest Classifier is best suited for internet traffic classification. Though our model gives **99**% accuracy we can't say it is the ultimate model as in the future many features can be added or deleted, So as learning is the continuous process for human brain similarly learning is the continuous process for ML model. Further in this report we, tried to experiment with the balanced dataset by applying feature importance, which decreased the number of features from 83 to 55. Hence on further looking the Confusion matrix for reference we are able to conclude that our prediction model is not biased as the number of correctly predicted values is larger than that of falsely predicted numbers .To the best of our knowledge the work reported in this paper is the study of how different classification algorithms of the ml classify the network traffic.

FUTURE WORK:

With the work discussed above, this report is one of several that have applied machine-learning techniques to the problem of network-traffic classification. In future work we plan to apply cross validation technique to confirm whether our decision tree model is overfitting or not, in case of overfitting we can apply post pruning methods to overcome this problem. We also want to explore some more techniques of feature selection and apply different ml algorithms (like gradient boosting trees). At last our future work will depend on the contributions and limitations of other researchers work.

REFERENCES:

[1] F. Pacheco, E. Exposito, M. Gineste, C. Baudoin, and J. Aguilar, ‘‘Towards the deployment of machine learning solutions in network traffic classification: A systematic survey,’’ IEEE Common. Surveys Tuts., vol. 21, no. 2, pp. 1988–2014, 2nd Quart., 2019.

[2] Recent Advancement in Machine Learning Based Internet Traffic Classification

[3] LiTing Hu and LiJun Zhang. Real-time Internet Traffic Identification Based on Decision Tree; World Automation Congress (WAC); 2012; IEEE; p. 1-3.

[4] Hu LT, Zhang LJ (2012) Real-time internet traffic identification based on decision tree. In: World Automation Congress 2012, pp 1–3. IEEE

[5] C. Wang, T. Xu and X. Qin, "Network Traffic Classification with Improved Random Forest," 2015 11th International Conference on Computational Intelligence and Security (CIS), Shenzhen, 2015, pp. 78-81, doi: 10.1109/CIS.2015.27.

[6] M. Dixit, R. Sharma, S. Shaikh and K. Muley, "Internet Traffic Detection using Naïve Bayes and K-Nearest Neighbors (KNN) algorithm," 2019 International Conference on Intelligent Computing and Control Systems (ICCS), Madurai, India, 2019, pp. 1153-1157, doi: 10.1109/ICCS45141.2019.9065655.

[7] D. Moore, K. Keys, R. Koga, E. Lagache, and K. C. Claffy. Coral Reef software suite as a tool for system and network administrators. In Proceedings of the LISA 2001 15th Systems Administration Conference, December 2001.

[8] C. Logg and L. Cottrell. Characterization of the Traffic between SLAC and the Internet, July 2003. http://www.slac.stanford.edu/comp/net/slacnet

[9] T. Karagiannis, A. Broido, M. Faloutsos, and k c claffy. Transport layer identification of P2P traffic. In Proceedings of Internet Measurement Conference, Taormina, Sicily, Italy, October 2004.

[10] A. W. Moore. Discrete content-based classication a data set. Technical report, Intel Research, Cambridge, 2005.

[11] K. C. Claffy. Internet traffic characterization. PhD thesis, University of California, San Diego, 1994.

[12] 4. Nguyen, T.T.T., Armitage, G.: A survey of techniques for internet traffic classification using machine learning. IEEE Common. Surv. Tutorials 10 (4), 56–76 (2008)

[13] https://www.kaggle.com/jsrojas/ip-network-traffic-flows-labeled-with-87-apps