**INTRODUCTION**

Since the evolution of modern technology and with the drastic increase in the scale of network communication more and more network disruptions in traffic and private protocols have been taking place. Identifying and classifying the unknown network disruptions can provide support and even help to maintain the backup systems. Furthermore, Research on Identifying and classifying the unknown network disruptions can help us overcome the problem of detecting an illegal network monitoring, intrusion detection, analysis of the network, and providing day-to-day analysis of the network can eventually help us to ensure the network behavior. This Network Disruptions can be identified in many ways such as:

The traditional method using fixed port numbers can be easily cheated by changing the port numbers in the system. Deep Packet Inspection is a widely used protocol identification technique that is been used at present, although it is widely used by organizations around the world, this has its limitations such as resource consumption might be very high when we deal with its feature database. It cannot accurately identify the protocol types. The other method which uses association rule mining for unknown protocol identification also has limitations, in the case of real-time large-scale network protocol analysis, the computational complicity might go very high. So alternatively, we can use Machine Learning to Identify network Disruptions, due to the fact of its powerful adaptive and learning capability, researchers from all around the world have been started using Machine Learning in the file of network disruption and on protocol analysis. Generally speaking, machine learning is divided into various branches, and depending on our needs we can work on a specific branch. (Supervised, Unsupervised, Semi-Supervised, Reinforcement learning.)

**PROBLEM STATEMENT**

The main objective of our problem is to predict the network fault severity at a particular location based on the log data available. The project has been done by the data collected from the Kaggle data repositories, consisting of various features which help us determine the network fault severity in the network. The datasets/log files which were used here are event\_type.csv, log\_feature.csv, resource\_type.csv, severity\_type.csv.

The target class variable Severity type has 3 classes such as 0,1,2, representing the fault severity of the network.

“Fault severity” is a measurement of actually reported faults from users of the network and is the target variable.

**RELATED WORKS**

Hong et al. proposed an application layer protocol that combines the traditional Deep packet Inspection and clustering methods which can effectively classify and identify the unknown application layer protocols which can intern help to protect from network disruptions.

Peng et al. proposed a way of classifying and identifying the network disruptions using mathematical statistics to calculate the k value, the cluster initial center of the K-Means Clustering Algorithm.

Similarly, Zhang et. Al. proposed a way of identifying and classifying the network by combining the traditional AGNES Hierarchical clustering algorithm with the features of bitstream data frames. This method has been proven for automatically identifying the number of clusters and classifying the unknown bitstream data frames.

However, most protocol identification methods that are based on using traditional machine learning algorithms require manual feature selection as inputs to train the model, in advance to further classify and identify the protocols.

Supervised Learning is a method of training the model with the input features and expecting the model to predict identification or classification results. Whereas deep learning is an advanced neural network that probably has more hidden layers which help to convert the data into data that can be learned by machines and helps us to identify and classify the network disruptions.

It learns the inherent rules and representation levels of sample data by transforming low-level features into sophisticated high-level features for expressing the qualities of input photos. This end-to-end learning strategy eliminates the time-consuming process of extracting features in advance, allowing the protocol classification and identification to become more automated. This method has shown good results in real-time analysis of online network traffic and massive data volume analysis, such as picture and video classification and identification analysis. Wang et al. first proposed this idea of treating bits of data traffic as pixels of an image and then applying deep learning techniques for identification and classifying the network traffics/disruptions.

**CONTRIBUTION OF OBJECTIVE**

As the world is dynamically evolving towards the new age of technology at the users using different networks increasing minute by minute, more and more network disruptions emerge and can pose a very serious threat to the organizations.

An artificial intelligence method was used to explore autonomous classification and identification of unknown network protocols in this paper to reduce the time and labor cost of network disruption classification and identification. In this paper, firstly, we are taking a dataset having each row corresponding to a location and a time point. This data is pre-processed and modeled using three Machine learning algorithms. As a result, we see which algorithm gives the best accuracy among the three that we have used.

**BLOCK DIAGRAM**

**Prediction**

**Testing Dataset**

**Model**

**Data**

**Evaluation**

**Algorithm**

**Training Dataset**

**Production data**

**Machine Learning Workflow Undertaken:**

We can define the machine learning workflow in 5 stages.

1. Gathering data
2. Data pre-processing
3. Researching the model that will be best for the type of data
4. Training and testing the model
5. Evaluation

**What is the machine learning model?**

The machine learning model is nothing but a piece of code; which an engineer or data scientist models by training it with the data according to the need of the project and making the model learn through the data and allowing it to predict or give the solution that we want whenever we ask it to give. So, whenever we give our model the new data which we want it to predict, we will get the predicted value according to the model training, the trained model might or might not perform well on the test data that we want it to predict, due to various reasons, so before trying to train any model we need to make sure that the algorithm that is going to use is appropriate for the desired class that we want to predict and based on the data that we are using.

**1. Gathering Data**

The process of gathering data depends on the type of project we desire to make, if we want to make an ML project that uses real-time data, then we can build an IoT system that uses different sensors data. Then the sensor data can be connected to the database where we want to store it. But the collected data cannot be used directly for performing the analysis, Since the collected data might be very irrelevant, extremely large values, unorganized text data, or noisy data to the project that we are working on. Therefore, to solve this problem Data Preparation is done meaning data cleaning.

# 2. Data pre-processing

Data pre-processing is one of the most important steps in machine learning. It is the most important step that helps us to provide accurate and cleaned data for training so that we can get accurate results. In machine learning, there is an 80/20 rule, where, every data scientist should spend his/her 80% time for data pre-processing and 20% time to perform the analysis and build the actual machine learning model.

**What is data pre-processing?**

Data pre-processing is a process of cleaning the raw data and making it a meaningful and understandable format. i.e., the data which is collected in the real world is not clean and consists of a lot of irrelevant data and inconsistent data, so we first convert our raw into a meaningful way to generate an efficient machine learning model pipeline. In other words, whenever the data is gathered from different sources it is collected in a raw format and this data isn’t feasible for the analysis.  
Therefore, certain steps are executed to convert the data into a small clean, and meaningful way, where our model can understand. This part of the process is called data pre-processing.

## **Why do we need it?**

As we know that data pre-processing is a process of cleaning the raw data into clean data so that it can be used to train the model. So, data preprocessing is an essential step to generate a machine learning model to help us predict, classify, forecast the data when we pass unknown data.

Below mentioned are some of the examples which mostly occur in raw data, when the data is collected.

1. **Missing data:** Missing data/ missing values are the empty spaces in our data that might have occurred due to various reasons, such as Structured missing values, missing completely at random, Missing at Random, missing not at random, each has its method of treating missing values and in the case of Sensor data, missing values might be high due to technical issues, electricity or other environmental factors.

2. **Noisy data:** This type of data is also called outliers; this can occur due to human errors (humans manually gathering the data) or some technical problem of the device at the time of collection of data. Outliers can be easily termed as extremely low values or extremely high values which are not related to the particular data variable.

3. **Inconsistent data:** This type of data error might mostly happen due to human errors (mistakes with the name or values) or duplication of data. Inconsistent in numbering formats or if the data variable is not consistent through all the rows, then we can term it as an inconsistent data variable, making a data variable consistent is very important since it might result in a certain type of bias when the model is trained and it might also result in inaccurate analysis results when we visualize them onto plots.

## **Three Types of Data**

1. Numeric e.g., Income, Age

2. Categorical e.g., Gender, Nationality

3. Ordinal e.g., Low/Medium/High, Education, Ranking

## How can data pre-processing be performed?

These are a lot of techniques that one can incur to pre-process the data to convert it from raw to a meaningful and understandable format.

1. **Conversion of data:** As we know Machine Learning models can only handle numeric features, hence categorical and ordinal data must be somehow converted into numeric features, we have a lot of transformation functions that we can use on our categorical/ string data type to convert it to numerical formats, such as 1. One Hot Encoding, 2. get\_dummies(), map().

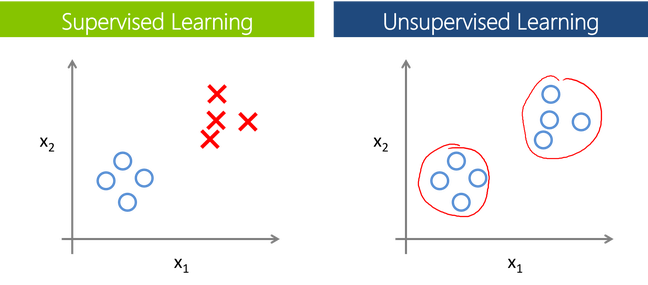
2. **Ignoring the missing values:** Whenever we encounter any missing values in our dataset it depends on the developer working on the project whether to delete the missing values or to impute some other values such as, mean, median, Predicting the missing value, or many other techniques.

3. **Filling the missing values:** The process of deleting the missing values can sometimes be efficient only if the missing values are a handful if the missing values are very large then in this case, we might also check the importance of the variables to our model and we can then delete the entire column with the most number of missing values, or we can impute some other value in place of missing value, by further analysis the dataset or by using statistical measurements, or by using some predicting algorithms to predict the missing values.

4. **Outlier’s detection:** As we have already seen, Outliers can be easily termed as extremely low values or extremely high values which are not related to the particular data variable. These are the error values are present in our data set that deviates drastically from other observations in a data set. [Example: human weight = 800 Kg; due to mistyping of extra 0, the entire value is changed, so this can be treated as an outlier, but in reality, outliers can be identified either by using plots or by statistical analysis.]

So, our main goal should be to train the best performing machine learning model nearly accurately using the cleaned/pre-processed data, so that it can help us to give the solutions whenever something new data is passed onto it related to the data that we have trained.

Note: The better your data is, the better the results will be.



## **Supervised Learning:**

Supervised learning is a branch of machine learning where for each row in the dataset, each row is tagged with a particular label known as the target class.

Supervised Learning is categorized into 2 other categories which are “**Classification**” and “**Regression**”.

## **Classification:**

**The classification**problem is when the target variable is **categorical**(i.e., the output variable consists of classes such as —Class A or B or something else, there might be 2 classes or more than 2 classes.).

A classification problem can be described as a problem where the target class in a dataset consists of categories, such as “Yes” or “No”, or “spam” or “not spam”.

Widely used Classification algorithms:

* **K-Nearest Neighbour**
* **Naive Bayes**
* **Decision Trees/Random Forest**
* **Support Vector Machine**
* **Logistic Regression**

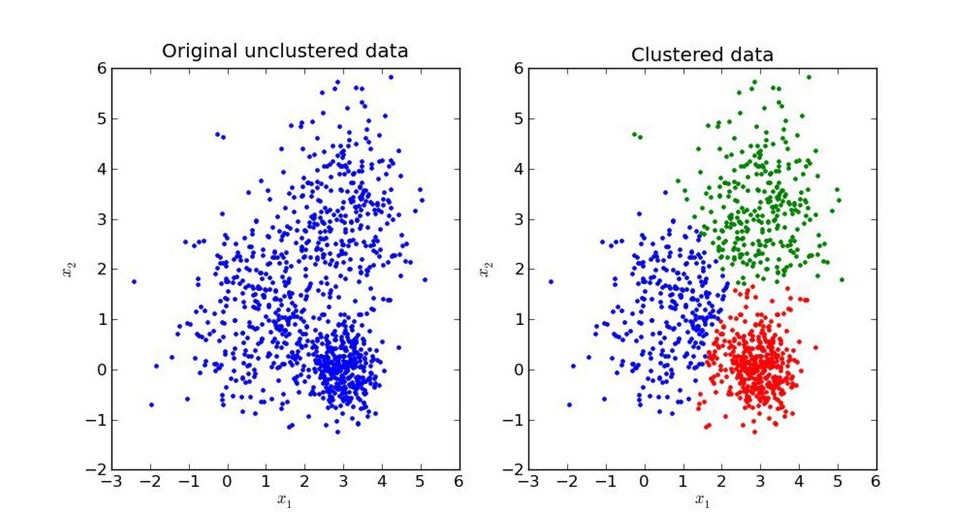
## **Regression:**

While a **Regression**problem is when the target variable is **continuous**(i.e., the output is numeric), Regression problem can be easily termed as the problem where we have to forecast about the future or what we do not know right now, it can be anything (Example: House Price Prediction, Stock market trends)

Widely used Regression Algorithms:

* **Linear Regression**
* **Support Vector Regression**
* **Decision Trees/Random Forest**
* **Gaussian Progresses Regression**
* **Ensemble Methods**

## **Unsupervised Learning:**



Unsupervised Learning is another branch of Machine Learning where we won’t be having any labels for each row of our data unlike supervised learning, so in this case, the model will try to segregate things based on the features and the data available. In simple terms it segregates the data in terms of clusters, the most important thing in unsupervised learning is the curse of finding the optimal k value (the number of clusters we would like to make).

Similar to Supervised, Unsupervised learning can also be categorized into 2 other categories as “**Clustering**” and “**Association**”.

## **Clustering**:

Clustering is a process of learning to assign labels to examples by leveraging an unlabelled dataset, Because the dataset is completely unlabelled, deciding on whether the learned model is optimal is much more complicated than in supervised learning.

Clustering Algorithms available:

* **DBSCAN**
* **HDBSCAN**
* **K-Means Clustering**
* **Hierarchical Clustering**
* **Gaussian Mixtures**
* **Spectral Clustering**

## **Overview of models under categories:**

**Machine Learning**

**Unsupervised**

**Supervised**

**Clustering**

**Regression**

**Classification**

**Decision Tree, Random Forest**

**Decision Tree**

**HDBSCAN**

**Hierarchical**

**Gaussian Mixture**

**K-Means**

**DBSCAN**

**Neural Networks**

**Ensemble Methods**

**SVR, GPR**

**Linear Regression**

**Neural Networks**

**K-Nearest Neighbors**

**SVM**

**Naïve Bayes**

# 4. Training and Testing the model.

Before building any machine learning Project, training is the most important part, where we train our model using the data available and make the machine learn and understand the data, after which when the model has learned from the data, we provide the model with another dataset to evaluate how good our model is performing, if it is performing well, we then test the model using test data, where we get to know the final performance of our model, which can be measure using various metrics, such as Accuracy, recall, precision, and through classification report.

This whole process of building and deploying a model is done using 3 different datasets which are split using train\_test\_split(), which are ‘**Training data**’, ‘**Validation data**’, and ‘**Testing data**’.

First, we train our classifier/regressor model using ‘**training data set**’, we then tune the parameters, to make the model more efficient ad accurate using ‘**validation set**’, and then we test the performance of our classifier on unseen data/ ‘**test data set**’ which should not be or by any means should be used for training or validating, if we use the same data for testing purpose, our model might perform well, but it will lead to overfitting. The test set will only be available during testing the classifier.

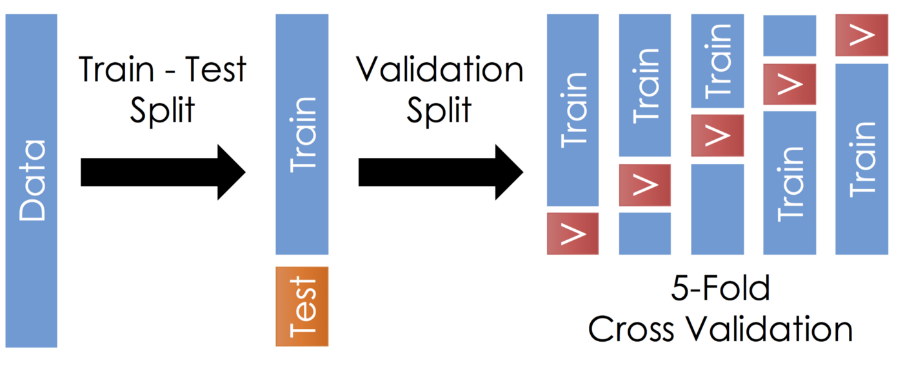


**Training set:** The training set is like a learning material which we give our model to learn so that it understands the data and applies what is understood to the data which we then use that trained model to predict the values with the new data. Training a machine learning model can be done using various algorithms, where the algorithms need to be selected carefully according to the problem that we are trying to solve.

**Validation set:** Cross-validation/ Validation Set is primarily used for estimating how good our model is performing on unseen. Based on the validation data, we can then tune our parameters to make the model more efficient and reliable for deployment.

**Test set:** Test Dataset is simply an unknown dataset, related to the original data which we split at the start so that the model is not aware of the values present in the dataset. The Test dataset is used for assessing the final performance of the model and how well the model is performing.

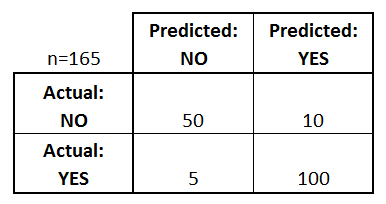
**Simple Example of Train, Test, Validation Splits.**



Once the model is trained by the required parameters and achieved a good accuracy score, we can then use the trained model for predicting our test data/unseen data. Once this is done, we can evaluate our model or plot the evaluation metrics of our model performance using **Classification report, Confusion Matrix**, **AUC\_ROC** Curve, etc.

**Confusion Matrix**

A confusion matrix is simply an evaluation metric matrix especially used for analyzing the behavior of the model and estimating how good the model is performing. It has 4 parameters, which are ‘**True positives’**,**‘True Negatives’**,**‘False Positives,** and ‘**False Negative’**. Which again derived into various formulas such as TPR, FPR, which helps us to get further performance analysis. Below mentioned is an ideal Confusion Matrix, In Confusion matrix the more the TP and Tn the better the model is, although depending on the project we are working on, we might care about FN and FP, when reducing the number of FP and FN, might become an important step during model evaluation.



* **True positives:** Both Predicted and Actual value is True.
* **True negatives:** Both Predicted and Actual Value is False.
* **False positives:** In this case, the actual value is False, but the model has predicted True.
* **False negatives: In this case, the actual value is True, but the model has predicted False.**

Using The TP, TN, FP, FN we can derive some formulas such as:

*Accuracy = (True Positives +True Negatives) / (Total number of classes)*

i.e., for the above example:

Accuracy = (100 + 50) / 165 = 0.9090 (90.9% accuracy)

Similarly, we can do it for Recall, Precision etc.

## 5. Evaluation

Model Evaluation is an integral part of the model development process. It helps us to find the best model that represents our data and it will help us to determine the best parameters which are used to fine-tune the model. This Evaluation step will help us determine the best model out of many models that we create.

**METHODOLOGIES**

**Dataset’s descriptions**:

* train.csv - the training set for the fault severity
* test.csv - the test set for fault severity
* sample\_submission.csv – a sample of the correct format for the input
* event\_type.csv: type of event related to the main dataset
* log\_feature.csv - features extracted from log files
* resource\_type.csv: resource type related to the main dataset
* severity\_type.csv: severity type of a warning message coming from the log

All the above CSV's except train.csv, test.csv, and sample\_submission.csv, have been merged to make it has a single CSV file based on a specific primary key.

**Machine Learning algorithms used**:

1. **The Random Forest Classifier**

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It is one of the widely used algorithms after Decision tree which perform well with any kind of dataset, be it classification or regression. It is based on the concept of **ensemble learning,** which is a process of combining multiple classifiers to solve a complex problem, and at the end, the results are either made an average of all the classifiers or mode of all the classifiers.

**"Random Forest is a classifier that contains several decision trees on different subsets of the given dataset and takes the average to improve the predictive accuracy of the model,"** as the name suggests. Rather than relying on a single decision tree, the random forest takes the predictions from each tree and predicts the final output based on the majority vote of predictions.

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

The Below Block diagram explains the working of the Random Forest algorithm:



## **Why use Random Forest?**

Below are some points that explain why we should use the Random Forest algorithm:

* Training time will be less when compared to other algorithms.
* It predicts output with high accuracy, even for the large dataset, it runs efficiently.
* It can also maintain accuracy when a large proportion of data is missing.
* Data transformation/scaling the data is optional in the case of random forest or decision tree.

1. **Decision Tree**

A Decision tree, as the name suggests, creates a branch of nodes, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and the last nodes are termed as the leaf nodes meaning there cannot be any nodes attached to them, and each leaf node (terminal node) holds a class label. The decision tree is one of the most popular algorithms in machine learning, it can be sued for both classification and regression, similar to a random forest, there are some exceptions to decision tree also, in terms of data scaling and data transformation, since decision tree works like a flowchart in the form of branches doing data transformation and scaling might be optional.



A Simple decision tree Model for predicting whether a person should play tennis or note given various parameters as inputs for the model.

**Construction of Decision Tree:**   
A tree can be "learned" by dividing the source set into subsets based on an attribute value test. This process is repeated on each derived subset in a recursive manner known as recursive partitioning. When the subset at a node all has the same value of the target variable, or when splitting no longer adds value to the predictions, the recursion is complete. Because the construction of a decision tree classifier does not require any domain knowledge or parameter setting, it is suitable for exploratory knowledge discovery. Decision trees are capable of dealing with high-dimensional data. In general, decision tree classifiers have high accuracy. A common inductive approach to learning classification knowledge is decision tree induction.

**Decision Tree Representation:**   
Decision trees classify instances by sorting them down the tree from the root to some leaf node, which provides the classification of the instance. An instance is classified by starting at the root node of the tree, testing the attribute specified by this node, then moving down the tree branch corresponding to the value of the attribute as shown in the above figure. This process is then repeated for the subtree rooted at the new node.

The decision tree in the above figure classifies a particular morning according to whether it is suitable for playing tennis and returning the classification associated with the particular leaf. (In this case Yes or No). 

For example, the instance

(Outlook = Rain, Temperature = Hot, Humidity = High, Wind = Strong)  
would be sorted down the leftmost branch of this decision tree and would therefore be classified as a negative instance.

In other words, we can say that the decision tree represents a disjunction of conjunctions of constraints on the attribute values of instances.

(Outlook = Sunny ^ Humidity = Normal) v (Outlook = Overcast) v (Outlook = Rain ^ Wind = Weak) 

**Strengths and Weakness of Decision Tree approach**   
The strengths of decision tree methods are:

* Decision trees can generate understandable rules.
* Data transformation and scaling are optional while implementing decision trees.
* Decision trees can handle both classification and regression problems.
* Decision trees can automatically understand which features are important and which are not.

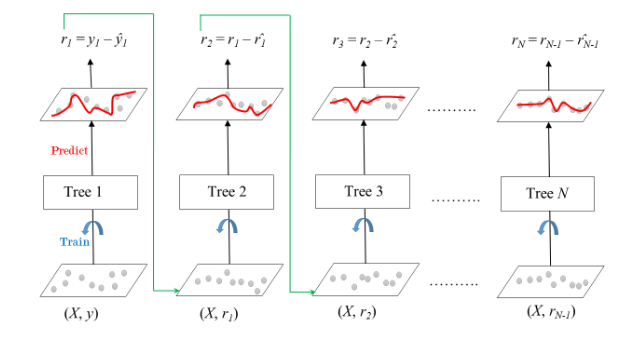
The weaknesses of decision tree methods:

* Decision trees are less appropriate for estimation tasks where the goal is to predict the value of a continuous attribute.
* Sometimes decision trees might give rise to the overfitting of a model.
* Decision trees are prone to errors in classification problems with many classes.
* Decision Trees cannot perform well with a small number of training examples.
* The process of growing a decision tree is computationally expensive. At each node, each candidate splitting field must be sorted before its best split can be found

**Gradient Boosting**

Gradient boosting is a technique used in the development of predictive models. The method is most commonly used in regression and classification procedures. Prediction models are frequently depicted as decision trees for selecting the best prediction. Gradient boosting, like other boosting methods, presents model building in stages while allowing the generalization and optimization of differentiable loss functions.

The below diagram explains how gradient boosted trees are trained for regression problems.



**Gradient Boosted Trees for Regression**

The whole gradient boosting algorithms works on the principle of learning from its predecessor / taking the output of its predecessor and giving it has the input to the next tree.

The ensemble consists of N trees. Tree1 is trained using the feature matrix X and the labels y. The predictions labeled y1(hat) are used to determine the training set residual errors r1. The feature matrix X and the residual errors r1 of Tree1 are then used as labels in training Tree2. The predicted r1(hat) results are then used to calculate the residual r2. This procedure is repeated until all N trees in the ensemble have been trained.

The **Shrinkage** parameter is an important one in this technique.

The term shrinkage refers to the fact that the prediction of each tree in the ensemble is shrunk after being multiplied by the learning rate (eta), which ranges from 0 to 1.

There is a trade-off between eta and the number of estimators, decreasing learning rate needs to be compensated with increasing estimators to reach certain model performance. Since all trees are trained now, predictions can be made.

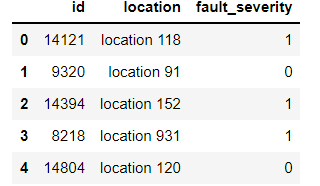
Each tree predicts a label and final prediction is given by the formula,

y(pred) = y1 + (eta \* r1) + (eta \* r2) + ....... + (eta \* rN)

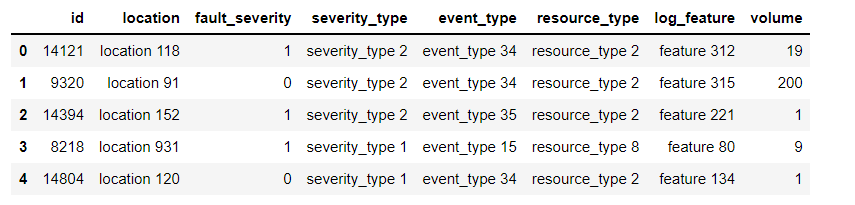
Similar to Decision Tree and Random Forest, Gradient Boosting can be trained applied to both Classification and Regression named as, as “**GradientBoostingClassifier”** and **“GradientBoostingRegressor”.**

**Results**

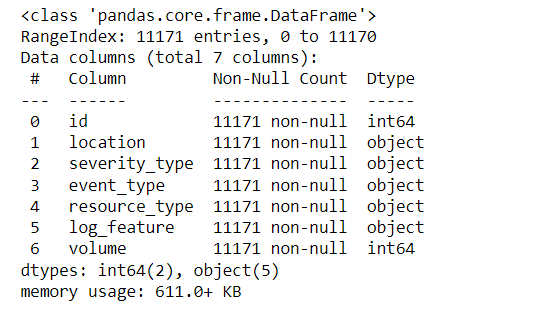
* 1. Preview of data set, df.head() or df.tail()



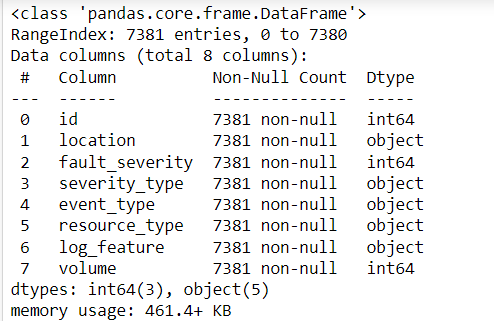
* 1. Entire dataset after merging

****

* 1. Train data set summary, train.info ()

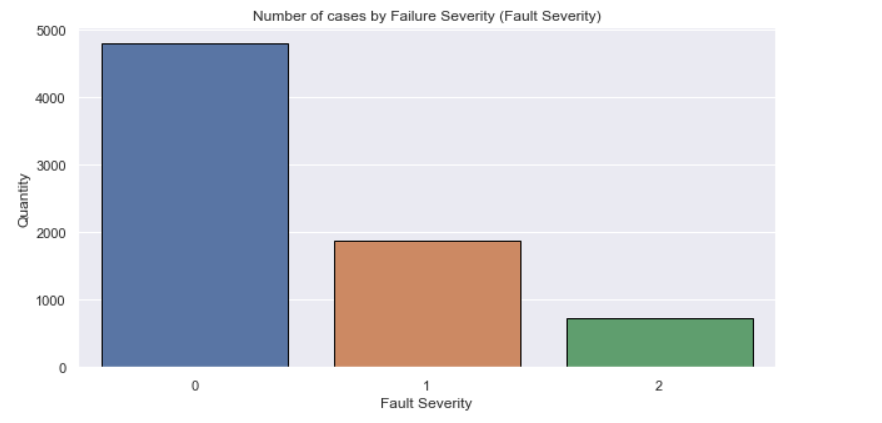
****

* 1. Information of the Test Data, extracted using test.info()

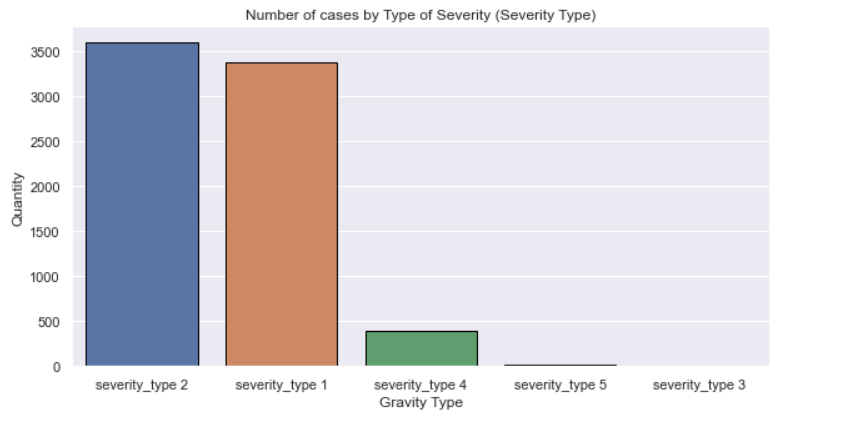


**Visual Analysis**

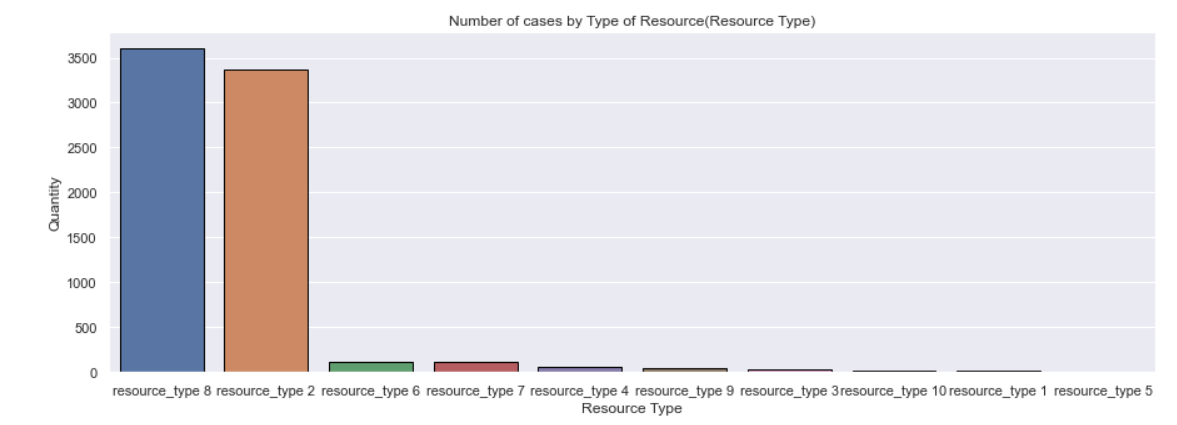
1. Number of cases by Failure Severity (Fault Severity)



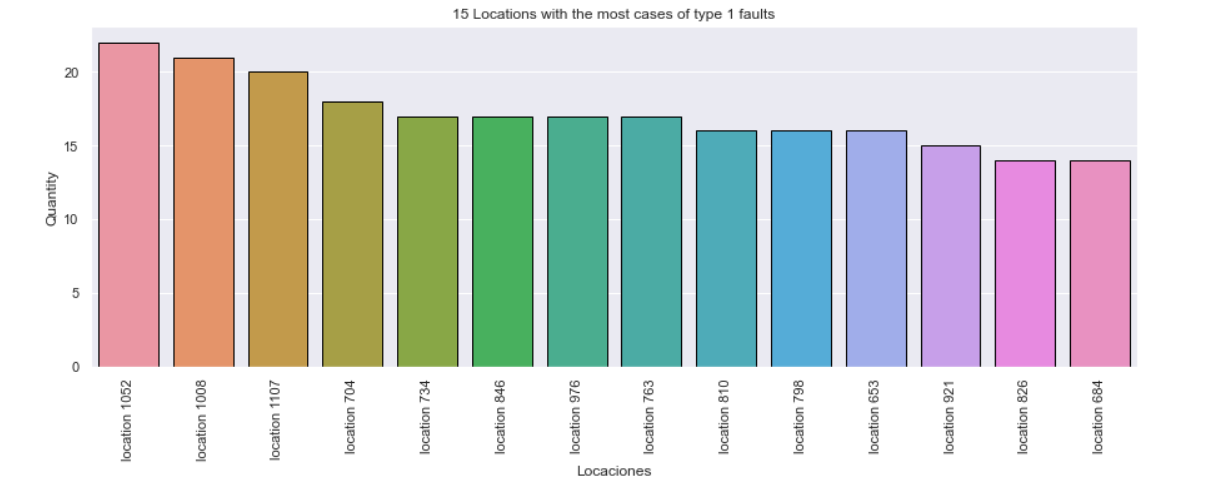
1. Number of cases by Type of Severity (Severity Type)

****

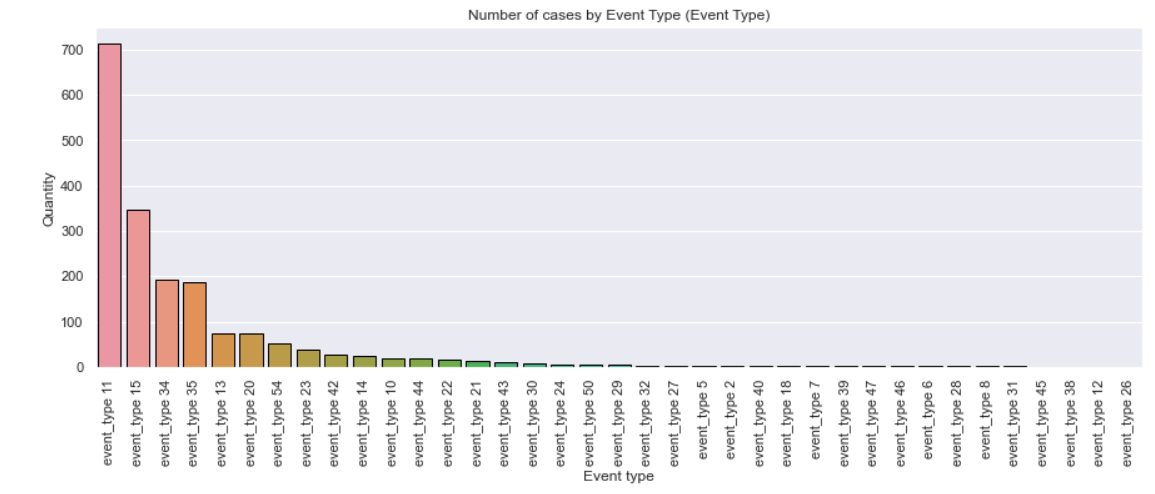
1. Number of cases by Type of Resource (Resource Type)



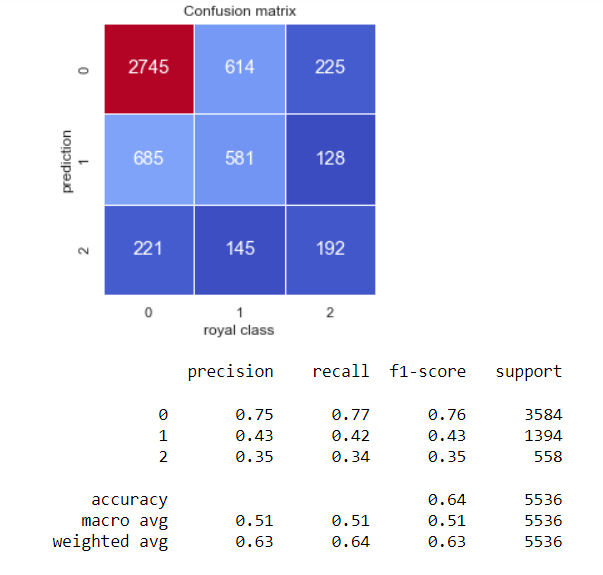
1. 15 Locations with the most cases of type 1 faults

****

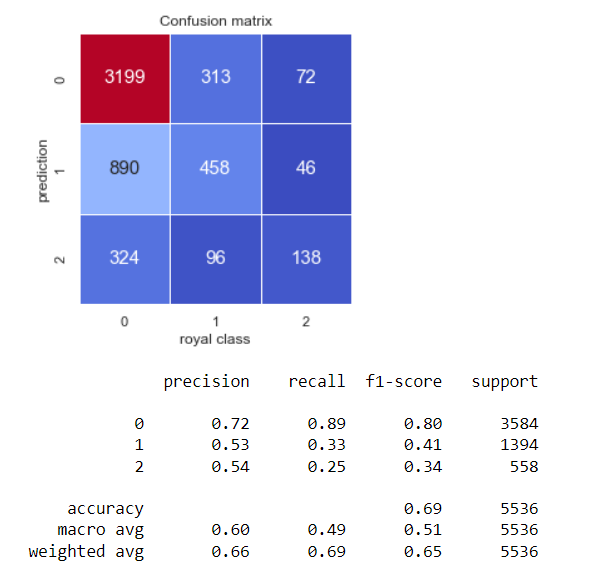
1. Number of cases by Event Type (Event Type)

****

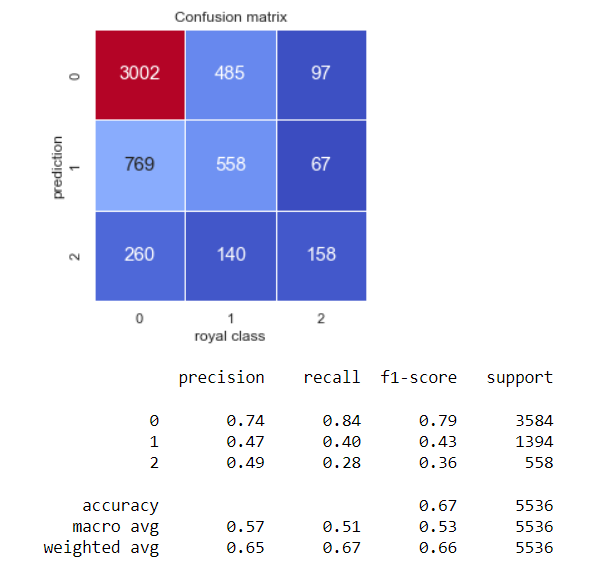
**Results of Decision Tree Classifier**

****

**Results of Gradient Boost Classifier**

****

**Results of Random Forest Classifier**

****

**CONCLUSION AND FUTURE SCOPE**

As per the main objective of the project is to classify and identify the unknown network disruptions based on ML algorithms is being discussed throughout the project. Through this method, first, we have extracted the disrupted data information of the network traffic. Then the dataset is being sent for cleaning and data pre-processing to bring the data to the same scale which should be understandable to the machine and in the process of that we have merged all the files as one file to get a better understanding of the data to further help us classify and identify the fault severity. Finally, feature engineering is done to intelligently select the feature vectors to efficiently and accurately realize the classification and identification of unknown network disruption. This method made full use of the advantage of Machine Learning algorithms. Based on ensuring the classification and identification accuracy, it avoided the complex steps of manually extracting features and reduced the training time of the intelligent algorithm as well as the amount of labeled data required.

As part of the future scope, we hope to try out different algorithms to optimize the feature output process, increase the feature similarity of the same disruption data and widen the differences between different disruption data to improve the model's representation capability. We will also do further research on encrypted traffic, and try to use neural networks to find the potential characteristics of encrypted data.

**REFERENCE**

1. Hong Z, Gong Q, Feng W, Li Y. Unknown Application Layer Protocol Identification Based on Adaptive Clustering. Computer Engineering and Applications. 2020, 56(05): 109-117.
2. Moore A W, Zuev D. Internet traffic classification using Bayesian analysis techniques[C]//Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems. 2005: 50-60.
3. Guo L. Research on Multi-Business Identification Technology Oriented High-Speed Network Management and Control. Doctor, The PLA Information Engineering University, Zhengzhou, Henan, China, 2012.
4. Peng D, Xiang L, Li S, Yang C, Qiu Y. Classification of intelligent home protocol under multiprotocol’s. Journal of Chongqing University of Posts and Telecommunications (Natural Science Edition), 2018,30 (03): 321-328.
5. Zhang F, Zhou H, Zhang J, Liu Y, Zhang C. A protocol classification algorithm based on improved AGNES. Computer Engineering and Science, 2017,39 (04): 796-803.
6. Wang Z. The applications of deep learning on traffic identification[J]. Blackhat USA, 2015, 24(11): 1-10.
7. Zhang, L.; Liao, P.; Zhao, J.; Guo, L. A Method of Unknown Protocol Identification Based on Convolution Neural Network. Microelectronics & Computer, 2018,35 (07): 106-108.
8. Wang W, Zhu M, Zeng X, et al. Malware traffic classification using convolutional neural network for representation learning[C]//2017 International Conference on Information Networking (ICOIN). IEEE, 2017: 712-717.
9. Li R, Xiao X, Ni S, et al. Byte segment neural network for network traffic classification[C]//2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS). IEEE, 2018: 1-10.
10. Feng W, Hong Z, Wu L, Fu M. Review of network protocol identification techniques. Computer Applications. 2019, 39: 3604-3614.