**INTRODUCTION**

**1.1 Overview**

An intrusion detection system (IDS) monitors network traffic and monitors for suspicious activity and alerts the system or network administrator. In some cases the IDS may also respond to anomalous or malicious traffic by taking action such as blocking the user or source IP address from accessing the network. An intrusion detection system can be a great tool for proactively monitoring and protecting your network from malicious activity, however they are also prone to false alarms.

An intrusion detection system (IDS) is a device that monitors network or system activities for malicious activities or policy violations and produces reports to a management station. Some systems may attempt to stop an intrusion attempt but this is neither required nor expected of a monitoring system. Network security and security analysis level using Intrusion detection system are primarily focused on identifying possible incidents, logging information about them, and reporting attempts. In addition, organizations use. Intrusion detection system for security and security analysis level for other purposes, such as identifying problems with security policies, documenting existing threats and deterring individuals from violating security policies. Network security and security analysis level using Intrusion detection system have become a necessary addition to the security infrastructure of nearly every organization.

Network security and security analysis level using Intrusion detection system typically record information related to observed events, notify security administrators of important observed events and produce reports. Many intrusion detection systems for security and security analysis level can also respond to a detected threat by attempting to prevent it from succeeding. They use several response techniques, which involve the IDPS stopping the attack itself, changing the security environment (e.g. reconfiguring a firewall) or changing the attack's content.

Intrusion detection systems do exactly as the name suggests: they detect possible intrusions. More specifically, ids tools aim to detect computer attacks and/or computer misuse, and to alert the proper individuals upon detection. An ids installed on a network provides much the same purpose as a burglar alarm system installed in a house. Through various methods, both detect when an intruder/attacker/burglar is present, and both subsequently issue some type of warning or alert.

Although ids may be used in conjunction with firewalls, which aim to regulate and control the flow of information into and out of a network, the two security tools should not be considered the same thing. Using the previous example, firewalls can be thought of as a fence or a security guard placed in front of a house. They protect a network and attempt to prevent intrusions, while ids tools detect whether or not the network is under attack or has, in fact, been breached. Ids tools thus form an integral part of a thorough and complete security system. They don’t fully guarantee security, but when used with security policy, vulnerability assessments, data encryption, user authentication, access control, and firewalls, they can greatly enhance network safety.

Intrusion detection systems serve three essential security functions: they monitor, detect, and respond to unauthorized activity by company insiders and outsider intrusion. Intrusion detection systems use policies to define certain events that, if detected will issue an alert. In other words, if a particular event is considered to constitute a security incident, an alert will be issued if that event is detected. Certain intrusion detection systems have the capability of sending out alerts, so that the administrator of the ids will receive a notification of a possible security incident in the form of a page, email. Many intrusion detection systems not only recognize a particular incident and issue an appropriate alert, they also respond automatically to the event. Such a response might include logging off a user, disabling a user account.

**1.2 Objectives:**

The Objectives of this project are as follows:

1. To design an Intrusion Detection System.
2. To determine the classifier to be used in this system.
3. To find out which classifier is best for intrusion detection.
4. To design IDS that has less execution time and higher accuracy.

**LITERATURE REVIEW**

**2.1 Background Study**

There are a number of common attacks on networked computers which, for their detection, require information from multiple sources, one of the most common of these is the so called doorknob attack. In a doorknob attack the goal is to discover, and gain access to, insufficiently protected systems. The incidents, which have raised the most concern in recent years, are the Denial-of-Service (DOS) attacks whose sole purpose is to reduce or eliminate the availability of a service provided over the Internet, to its legitimate users. The first kind of attacks can be avoided by patching-up vulnerable software and updating the host systems from time to time. In comparison, the second kinds of DOS attacks are much more difficult to defend. This works by sending a large number of packets to the target, so that some critical resources of the victim are exhausted and the victim can no longer communicate with other users. For second type of attack in IDS is most popular tool.

Packets sent using the IP protocol include the IP address of the sending host. The recipient directs replies to the sender using this source address. However, the correctness of this address is not verified by the protocol. The IP protocol specifies no method for validating the authenticity of the packet’s source. This implies that an attacker could forge the source address to be any he desires. This is a well-known problem and has been well described in all but a few rare cases; sending Network packets are done for illegitimate purposes. The original path identification marking is based on the use of the packet’s TTL field as an index into the IP Identification field where a router should add its marks. This method is not as lightweight as the Stack Path identification method. Legacy routers have a harmful effect on the original Path identification scheme because they decrement the TTL of a packet but do not add any markings. The Stack Path identification scheme is robust to legacy routers and even includes the write-ahead scheme to incorporate markings for single legacy routers in the path.

**2.2 Related Study**

Elhadj Benkhelifa et al. has proposed a paper that is concerned with advancements in intrusion detection practices in IoT. It provides a comprehensive review of current Intrusion Detection Systems (IDS) for IoT technologies, focusing upon architecture types. A proposal for future directions in IoT based IDS are then presented and evaluated. This then shows how traditional practices are unsuitable due to their inherent features providing poor coverage of the IoT domain [1].

John E. Dickerson et al. has proposed Fuzzy Intrusion Recognition Engine. The Fuzzy Intrusion Recognition Engine (FIRE) is an anomaly-based intrusion detection system that uses fuzzy logic to assess whether malicious activity is taking place on a network. It uses simple data mining techniques to process the network input data and help expose metrics that are particularly significant to anomaly detection. These metrics are then evaluated as fuzzy sets. FIRE uses a fuzzy analysis engine to evaluate the fuzzy inputs and trigger alert levels for the security administrator [2].

Preeti Aggarwal et al. has presented the analysis of KDD data set with respect to four classes which are Basic, Content, Traffic and Host in which all data attributes can be categorized. The analysis is done with respect to two prominent evaluation metrics, Detection Rate (DR) and False Alarm Rate (FAR) for an Intrusion Detection System (IDS). As a result of this empirical analysis on the data set, the contribution of each of four classes of attributes on DR and FAR is shown which can help enhance the suitability of data set to achieve maximum DR with minimum FAR [3].

Nabila Farnaaz et al. has built a model for intrusion detection system using random forest classifier. Random Forest (RF) is an ensemble classifier and performs well compared to other traditional classifiers for effective classification of attacks. To evaluate the performance of this model, the authors conducted experiments on NSL-KDD data set. Empirical result show that proposed model is efficient with low false alarm rate and high detection rate [4].

F.Sabahi et al. [2008] has presented a taxonomy of intrusion detection systems that is then used to survey and classify them. The taxonomy consists of the detection principle, and second of certain operational aspects of the intrusion detection system. The paper provides an overview of intrusion detection techniques and methods [5].

Fadi Salo et al. categorized the fields of data mining and intrusion detection systems (IDS), providing a systematic treatment of methodologies and techniques. A criterion-based approach to select 95 relevant articles from the period 2007-2017 has been applied. 19 separate data mining techniques used for intrusion detection has been identified, and this analysis encompasses rich information for future research based on the strengths and weaknesses of these techniques [6].

Hichem Sedjelmaci et al. has designed and implemented a novel intrusion detection and response scheme, which operates at the UAV and ground station levels, to detect malicious anomalies that threaten the network. In this scheme, a set of detection and response techniques are proposed to monitor the UAV behaviors and categorized them into the appropriate list (normal, abnormal, suspect, and malicious) according to the detected cyber-attack. Extensive simulations confirm that the proposed scheme performs well in terms of attack detection even with a large number of UAVs and attackers since it exhibits a high detection rate, a low number of false positives, and prompt detection with a low communication overhead. [7].

Alex Shenfield et al. has presents a novel approach to detection of malicious network traffic using artificial neural networks suitable for use in deep packet inspection based intrusion detection systems. Experimental results using a range of typical benign network traffic data (images, dynamic link library files, and a selection of other miscellaneous files such as logs, music files, and word processing documents) and malicious shell code files sourced from the online exploit and vulnerability repository exploit, have shown that the proposed artificial neural network architecture is able to distinguish between benign and malicious network traffic accurately [8].

Kehe Wu et al. has proposed a novel network intrusion detection model utilizing convolutional neural networks (CNNs). CNN is used to select traffic features from raw dataset automatically, and set the cost function weight coefficient of each class based on its numbers to solve the imbalanced dataset problem. The model not only reduces the false alarm rate (FAR) but also improves the accuracy of the class with small numbers. To reduce the calculation cost further, the raw traffic vector format is converted into image format. The standard NSL-KDD dataset is used to evaluate the performance of the proposed CNN model. The experimental results show that the accuracy, FAR and calculation cost of the proposed model performs better than traditional standard algorithms. It is an effective and reliable solution for intrusion detection of a massive network [9].

Zhihua Zhang et al. has proposed intrusion detection based on dynamic state context and hierarchical trust in WSNs (IDSHT) which is flexible and suitable for constantly changing WSNs characterized by changes in the perceptual environment, transitions of states of nodes and variations in trust value. A multidimensional two-tier hierarchical trust mechanism in the level of sensor nodes (SNs) and cluster heads (CHs) considering interactive trust, honesty trust and content trust is put forward, which combines direct evaluation and feedback-based evaluation in the fixed hop range. The experiment simulation and evaluation indicate that the mechanism proposed here outperforms the existing typical system in malicious detection and resource overhead [10].

**ANALYSIS OF PROBLEM**

On observing the Literature Survey below, it is observed that the there are many problems that are present in that particular systems. These problems create disturbances in the system by breaking the security layer of the system thus allowing intrusion attack on the system. This Breaching of security may cause serious damage to the system as well as the users that are using it thus causing serious economic losses to the organization as well as the users accessing it. Also if the system met with a serious damage, it may cause much time delay in recovering the system form the damage. This must be avoided so as to continue the services of the system without causing any disturbances or discontinuation in the service provided. These problems are described below that cause the disturbances in the system.

**Problem Occurred in previous methods**

**False Alarm:**

It is well known that IDS suffers from the high rate of false alarms. Continuous efforts are being made to reduce the high false positive rate. It is believed that intrusion detection is a data analysis process and can be studied as a problem of classifying data correctly.

**Accuracy**

From this standpoint, it can also be observed that any classification scheme is as good as the data presented to it as input. More clean the data, higher accurate results are likely to be obtained. From anomaly-based IDS point of view, it implies that if it is possible to extract features that demarcate normal data from abnormal one properly, false positive rate can be reduced to a great extent.

**Execution Time**

The fast detection of attacks remains one of the focal points to be worried about. With the present complexity and variety of attacks, there is a need for a huge amount of data to analyze and produce results. But larger the amount of data, longer the time to analyze it, which delays the detection of attacks. An IDS will be of more use if it can trigger an alarm early enough to reduce the damage that an ongoing attack can do. Thus, there is a need to make IDS as fast as to operate on-line. It is believed that this can be achieved if it can reduce the data, to be analyzed, without degrading its quality.

In this project above problems are considered for overcoming it. The problems such as accuracy are solved with the help of different classifiers such as Decision Tree, Multilayer Perceptron, K – Means and Random Forest. The accuracy of this classifier is higher than that to previous systems. Also the execution time of this system is less than that of some other systems. This system tends to solve both this problems and improve the efficiency of the system. The accuracy of this system is higher with lesser execution time.

**SYSTEM DESIGN**

**4.1 Methodology**

The system is designed for the detection of any intruder in a system. The system is designed using KDD Cup Data Set. Various algorithms are applied and tested on this data set to find out the effectiveness of the system to detect and prevent intrusion. The KDD data set is a well - known benchmark in the research of Intrusion Detection techniques. A lot of work is going on for the improvement of intrusion detection strategies while the research on the data used for training and testing the detection model is equally of prime concern because better data quality can improve offline intrusion detection. This data set is the most widely used data set for intrusion detection on networks. This data set has many features which helps in contributing for the intrusion detection system.

Many different classifiers are used here in order to perform operations on this data set. The classifiers help to calculate the efficiency of the system in order to prevent the intrusion of the system.

The different classifier used here are as follows:

* Random Forest Classifier
* Multiple Perceptron Classifier
* Decision Tree Classifier
* K- Means Classifier
* **Random Forest Classifier**

Random forest (RF) is an ensemble classifier used to improve the accuracy. Random forest consists of many decision trees. Random forest has low classification error compared to other traditional classification algorithms. Number of trees, minimum node size and number of features used for splitting each node.

Advantages of RF are listed below.

1) Generated forests can be saved for future reference.

2) Random forest overcomes the problem over fitting.

3) In RF accuracy and variable importance is automatically generated.

When constructing individual trees in random forest, randomization is applied to select the best node to split on. This value is equal to √ A, where A is no. of attributes in the data set. However, RF will generate many noisy trees, which affect accuracy and wrong decision for new sample.

* **Multilayer Perceptron Classifier**

A multilayer perceptron (MLP) is a class of [feed forward](https://en.wikipedia.org/wiki/Feedforward_neural_network) [artificial neural network](https://en.wikipedia.org/wiki/Artificial_neural_network). A MLP consists of at least three layers of nodes: an input layer, a hidden layer and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear [activation function](https://en.wikipedia.org/wiki/Activation_function). MLP utilizes a [supervised learning](https://en.wikipedia.org/wiki/Supervised_learning) technique called [back propagation](https://en.wikipedia.org/wiki/Backpropagation) for training. Its multiple layers and non-linear activation distinguish MLP from a linear [perceptron](https://en.wikipedia.org/wiki/Perceptron). It can distinguish data that is not [linearly separable](https://en.wikipedia.org/wiki/Linear_separability).

Multilayer perceptron are sometimes colloquially referred to as "vanilla" neural networks, especially when they have a single hidden layer.

* **Decision Tree Classifier**

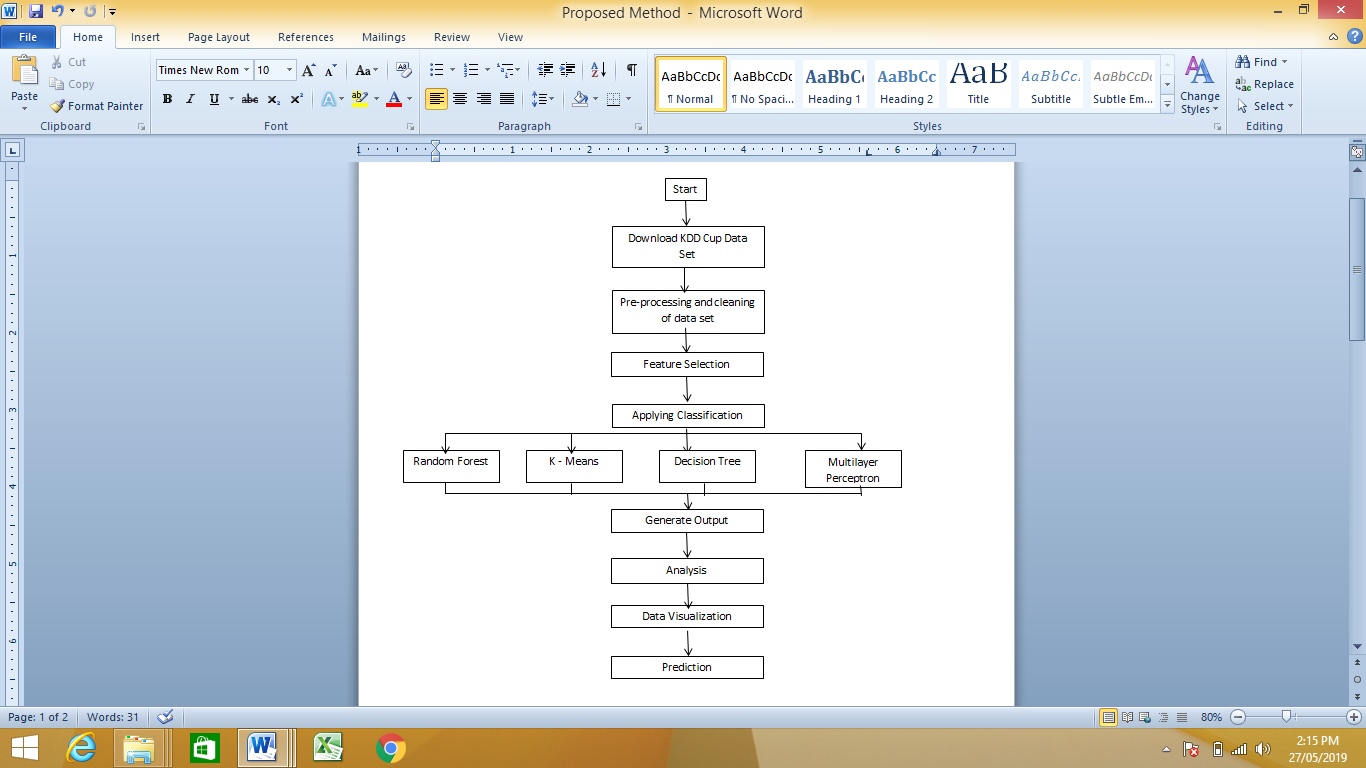
Decision Tree Classifier is a simple and widely used classification technique. It applies a straightforward idea to solve the classification problem. Decision Tree Classifier poses a series of carefully crafted questions about the attributes of the test record. Each time it receives an answer, a follow-up question is asked until a conclusion about the class label of the record is reached.

* **K – Means Classifier**

K - means clustering is a method of [vector quantization](https://en.wikipedia.org/wiki/Vector_quantization), originally from [signal processing](https://en.wikipedia.org/wiki/Signal_processing), that is popular for [cluster analysis](https://en.wikipedia.org/wiki/Cluster_analysis) in [data mining](https://en.wikipedia.org/wiki/Data_mining). k-means clustering aims to [partition](https://en.wikipedia.org/wiki/Partition_of_a_set) n observations into k clusters in which each observation belongs to the [cluster](https://en.wikipedia.org/wiki/Cluster_(statistics)) with the nearest [mean](https://en.wikipedia.org/wiki/Mean), serving as a prototype of the cluster. This results in a partitioning of the data space into [Voronoi cells](https://en.wikipedia.org/wiki/Voronoi_cell" \o "Voronoi cell).

The problem is computationally difficult ([NP-hard](https://en.wikipedia.org/wiki/NP-hardness)); however, efficient [heuristic algorithms](https://en.wikipedia.org/wiki/Heuristic_algorithm) converge quickly to a [local optimum](https://en.wikipedia.org/wiki/Local_optimum). These are usually similar to the [expectation-maximization algorithm](https://en.wikipedia.org/wiki/Expectation-maximization_algorithm) for [mixtures](https://en.wikipedia.org/wiki/Mixture_model) of [Gaussian distributions](https://en.wikipedia.org/wiki/Gaussian_distribution) via an iterative refinement approach employed by both k-means and Gaussian mixture modeling. They both use cluster centers to model the data; however, k-means clustering tends to find clusters of comparable spatial extent, while the expectation-maximization mechanism allows clusters to have different shapes.

**4.2 Flowchart of Proposed System.**



**Figure 1:** Flowchart of proposed Framework

**4.3 Algorithm for Proposed System :**

**Step 1:** Downloading the KDD Cup Data Set for Performing the operations onto it.

**Step 2:** The downloaded Data Set needs to be pre – processed and cleaned to remove the unwanted values from the Data Set.

**Step 3:** The features are to be selected onto which the operation is to be performed.

**Step 4:** The Classification Algorithm needs to be applied on the processed data Set.

**Step 5:** After applying the Classifiers, the output is generated.

This process helps the designer to create the secure system and helps to fight against Intrusion Attack on network data.

**IMPLEMENTATION**

**5.1 SYSTEM REQUIREMENTS**

**Software Requirements:**

* Language Used: Python
* Tools Used: PyCharm IDE
* Operating System: Linux, Ubuntu
* Browsers: Firefox, Chrome.

**Hardware Requirements:**

* RAM: Minimum 4 GB
* Hard Disk: Minimum 500 GB
* Processor: Intel Core i5
* Input Devices: Keyboard, Mouse.

The proposed system is designed using Python programming language, Flask Framework and PyCharm IDE. **Python** is an interpreted, [high-level](https://en.wikipedia.org/wiki/High-level_programming_language), [general-purpose](https://en.wikipedia.org/wiki/General-purpose_programming_language) [programming language](https://en.wikipedia.org/wiki/Programming_language). Created by [Guido van Rossum](https://en.wikipedia.org/wiki/Guido_van_Rossum) and first released in 1991, Python's design philosophy emphasizes [code readability](https://en.wikipedia.org/wiki/Code_readability) with its notable use of [significant whitespace](https://en.wikipedia.org/wiki/Off-side_rule). Its language constructs and [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) approach aims to help programmers write clear, logical code for small and large-scale projects.

Python is [dynamically typed](https://en.wikipedia.org/wiki/Dynamic_programming_language) and [garbage-collected](https://en.wikipedia.org/wiki/Garbage_collection_(computer_science)). It supports multiple [programming paradigms](https://en.wikipedia.org/wiki/Programming_paradigm), including [procedural](https://en.wikipedia.org/wiki/Procedural_programming), object-oriented, and [functional programming](https://en.wikipedia.org/wiki/Functional_programming). Python is often described as a "batteries included" language due to its comprehensive [standard library](https://en.wikipedia.org/wiki/Standard_library).

Python was conceived in the late 1980s as a successor to the [ABC language](https://en.wikipedia.org/wiki/ABC_(programming_language)). Python 2.0, released 2000, introduced features like [list comprehensions](https://en.wikipedia.org/wiki/List_comprehension) and a [garbage collection](https://en.wikipedia.org/wiki/Garbage_collection_(computer_science)) system capable of collecting [reference cycles](https://en.wikipedia.org/wiki/Reference_cycle). Python 3.0, released 2008, was a major revision of the language that is not completely [backward-compatible](https://en.wikipedia.org/wiki/Backward_compatibility), and much Python 2 code does not run unmodified on Python 3. Due to concern about the amount of code written for Python 2, support for Python 2.7 (the last release in the 2.x series) was extended to 2020. Language developer Guido van Rossum shouldered sole responsibility for the project until July 2018 but now shares his leadership as a member of a five-person steering council.

Python [interpreters](https://en.wikipedia.org/wiki/Interpreter_(computing)) are available for many [operating systems](https://en.wikipedia.org/wiki/Operating_system). A global community of programmers develops and maintains [CPython](https://en.wikipedia.org/wiki/CPython" \o "CPython), an [open source](https://en.wikipedia.org/wiki/Open-source_software) [reference implementation](https://en.wikipedia.org/wiki/Reference_implementation). A [non-profit organization](https://en.wikipedia.org/wiki/Nonprofit_organization), the [Python Software Foundation](https://en.wikipedia.org/wiki/Python_Software_Foundation), manages and directs resources for Python and CPython development.

The Features of Python programming Language includes:

1. Python is easy to learn and use. It is developer-friendly and high level programming language.
2. Python language is more expressive means that it is more understandable and readable.
3. Python is an interpreted language i.e. interpreter executes the code line by line at a time. This makes debugging easy and thus suitable for beginners.
4. Python can run equally on different platforms such as Windows, Linux, Unix and Macintosh etc. So, it can be said that Python is a portable language.
5. Python language is freely available at official website. The source-code is also available. Therefore it is open source.

**Pycharm**

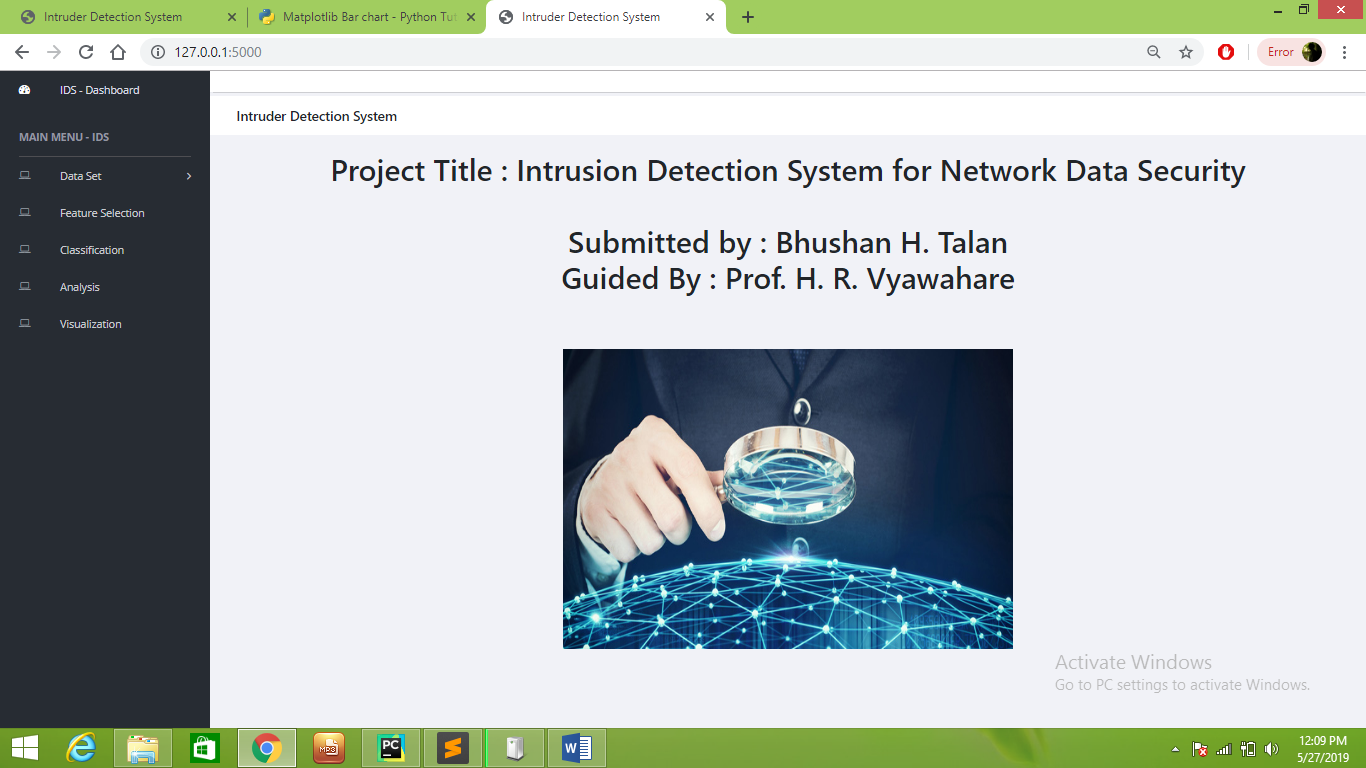
PyCharm is an [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) used in [computer programming](https://en.wikipedia.org/wiki/Computer_programming), specifically for the [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) language. It is developed by the Czech company [JetBrains](https://en.wikipedia.org/wiki/JetBrains" \o "JetBrains). It provides code analysis, a graphical debugger, an integrated unit tester, integration with [version control systems](https://en.wikipedia.org/wiki/Revision_control) (VCSes), and supports web development with [Django](https://en.wikipedia.org/wiki/Django_(web_framework)" \o "Django (web framework)).

PyCharm is [cross-platform](https://en.wikipedia.org/wiki/Cross-platform), with [Windows](https://en.wikipedia.org/wiki/Windows), [macOS](https://en.wikipedia.org/wiki/MacOS" \o "MacOS) and [Linux](https://en.wikipedia.org/wiki/Linux) versions. The Community Edition is released under the [Apache License](https://en.wikipedia.org/wiki/Apache_License), and there is also Professional Edition with extra features – released under a [proprietary license](https://en.wikipedia.org/wiki/Proprietary_software).

**Features of PyCharm IDE.**

* Coding assistance and [analysis](https://en.wikipedia.org/wiki/Code_analysis), with [code completion](https://en.wikipedia.org/wiki/Autocomplete), syntax and error highlighting, [linter integration](https://en.wikipedia.org/wiki/Lint_(software)), and quick fixes
* Project and code navigation: specialized project views, file structure views and quick jumping between files, classes, methods and usages
* Python [refactoring](https://en.wikipedia.org/wiki/Refactoring): including rename, extract method, introduce variable, introduce constant, pull up, push down and others
* Support for web frameworks: [Django](https://en.wikipedia.org/wiki/Django_(web_framework)" \o "Django (web framework)), [web2py](https://en.wikipedia.org/wiki/Web2py) and [Flask](https://en.wikipedia.org/wiki/Flask_(web_framework))
* Integrated Python [debugger](https://en.wikipedia.org/wiki/Debugger)
* Integrated [unit testing](https://en.wikipedia.org/wiki/Unit_testing), with line-by-line [code coverage](https://en.wikipedia.org/wiki/Code_coverage)
* [Google App Engine](https://en.wikipedia.org/wiki/Google_App_Engine) Python development
* Version control integration: unified user interface for [Mercurial](https://en.wikipedia.org/wiki/Mercurial), [Git](https://en.wikipedia.org/wiki/Git_(software)" \o "Git (software)), [Subversion](https://en.wikipedia.org/wiki/Apache_Subversion), [Perforce](https://en.wikipedia.org/wiki/Perforce) and [CVS](https://en.wikipedia.org/wiki/Concurrent_Versions_System) with change lists and merge.

**5.2 Project Screenshots:**

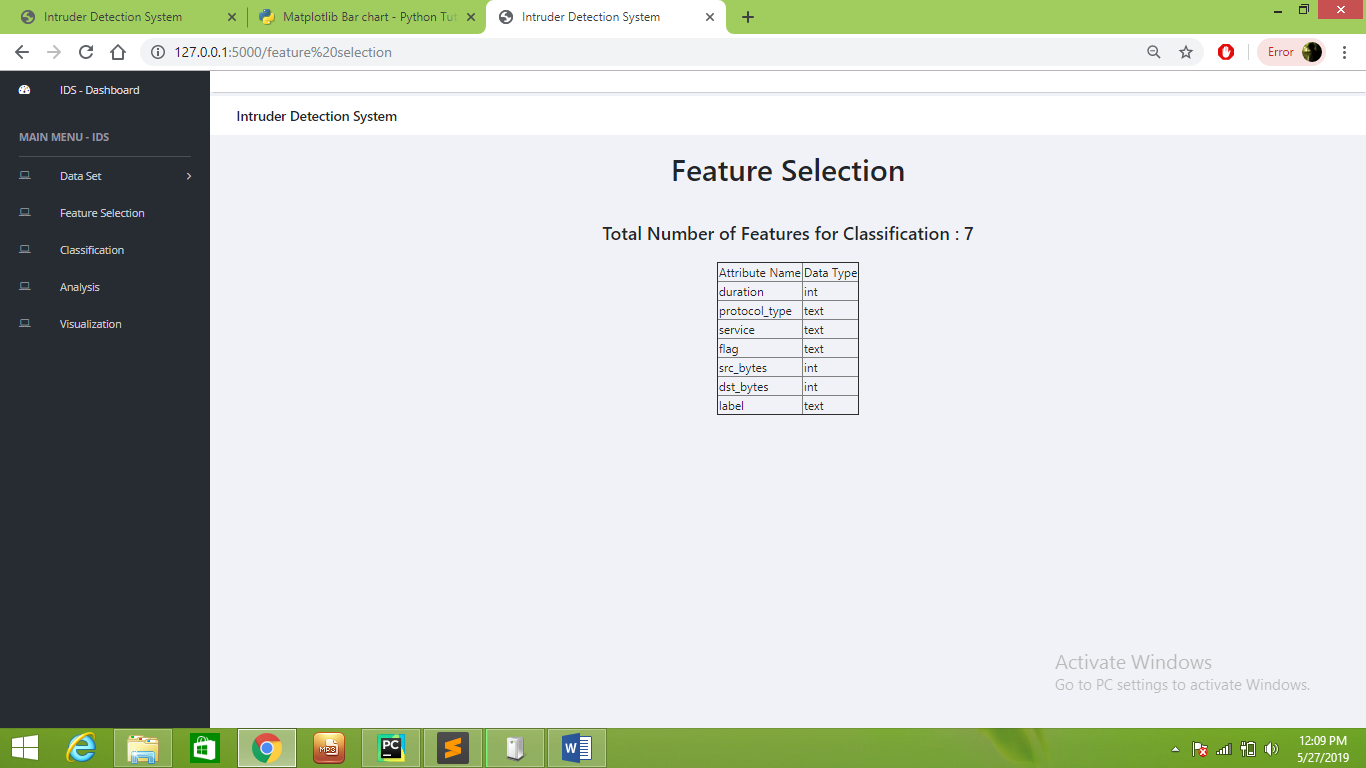


**Figure 2:** Screenshot showing Home Page



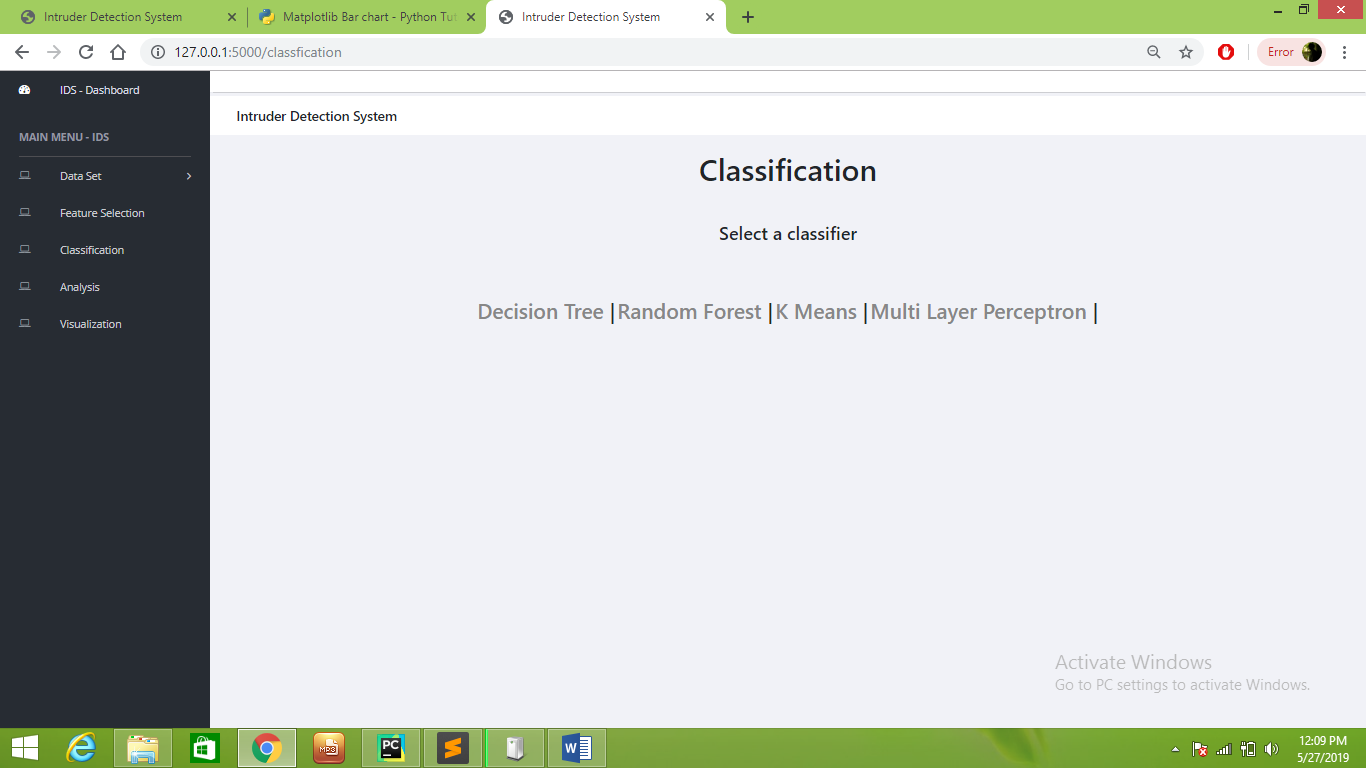
**Figure 3:** Screenshot showing the KDD Cup Data Set

This screen shows the processed and cleaned KDD Cup data Set which is to be used in the project.



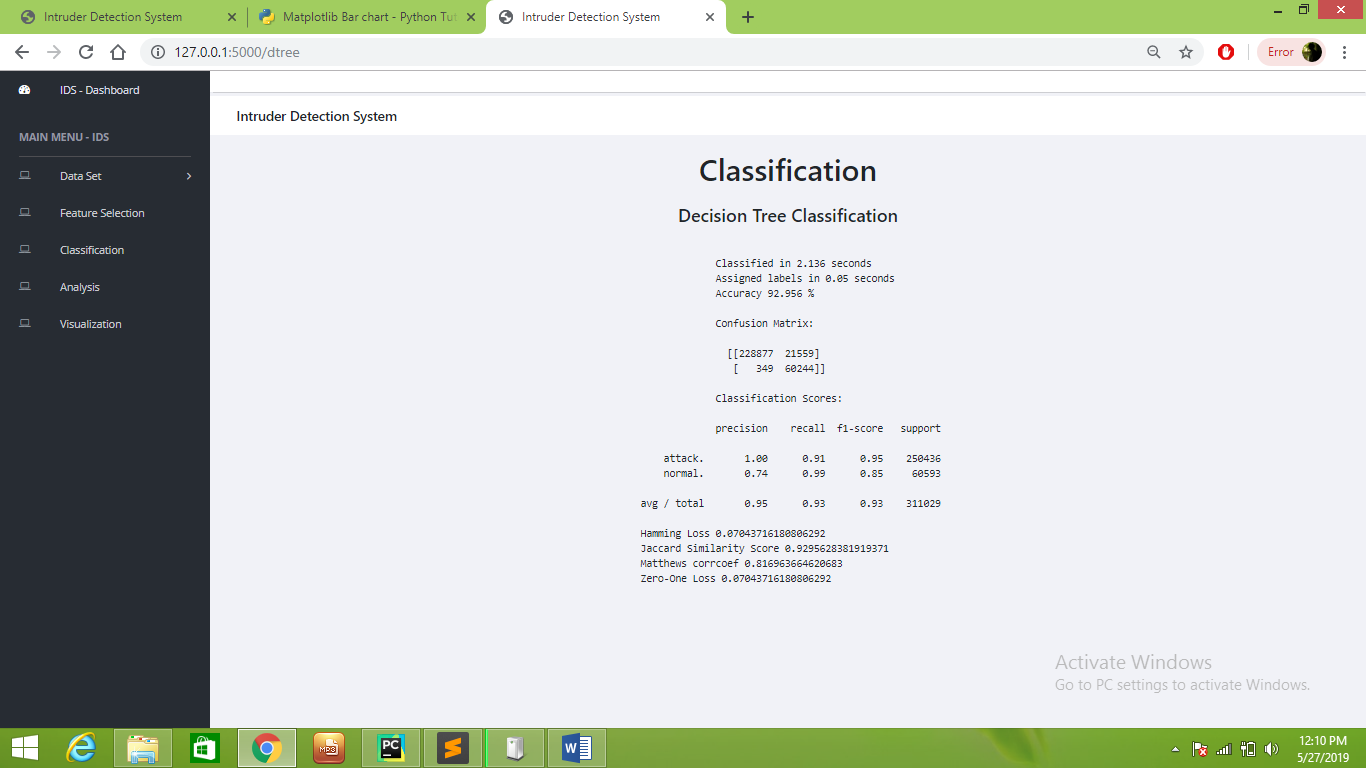
**Figure 4:** Screenshot showing selected Features

The Features are selected according to the requirements on which the Intrusion Detection is to done.



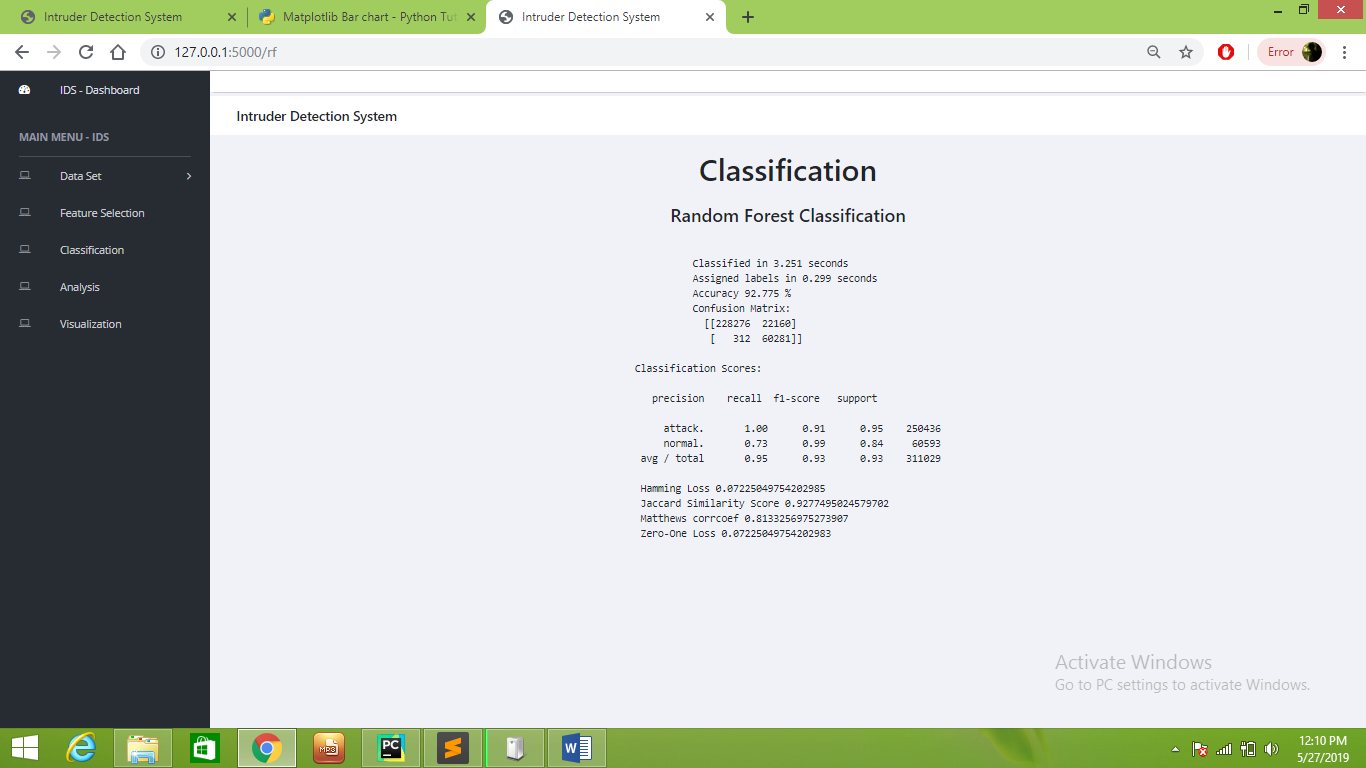
**Figure 5:**  Screenshot showing the Classifiers to be used.

This Screen shows the Classifiers which are used in this project for performing various operations.



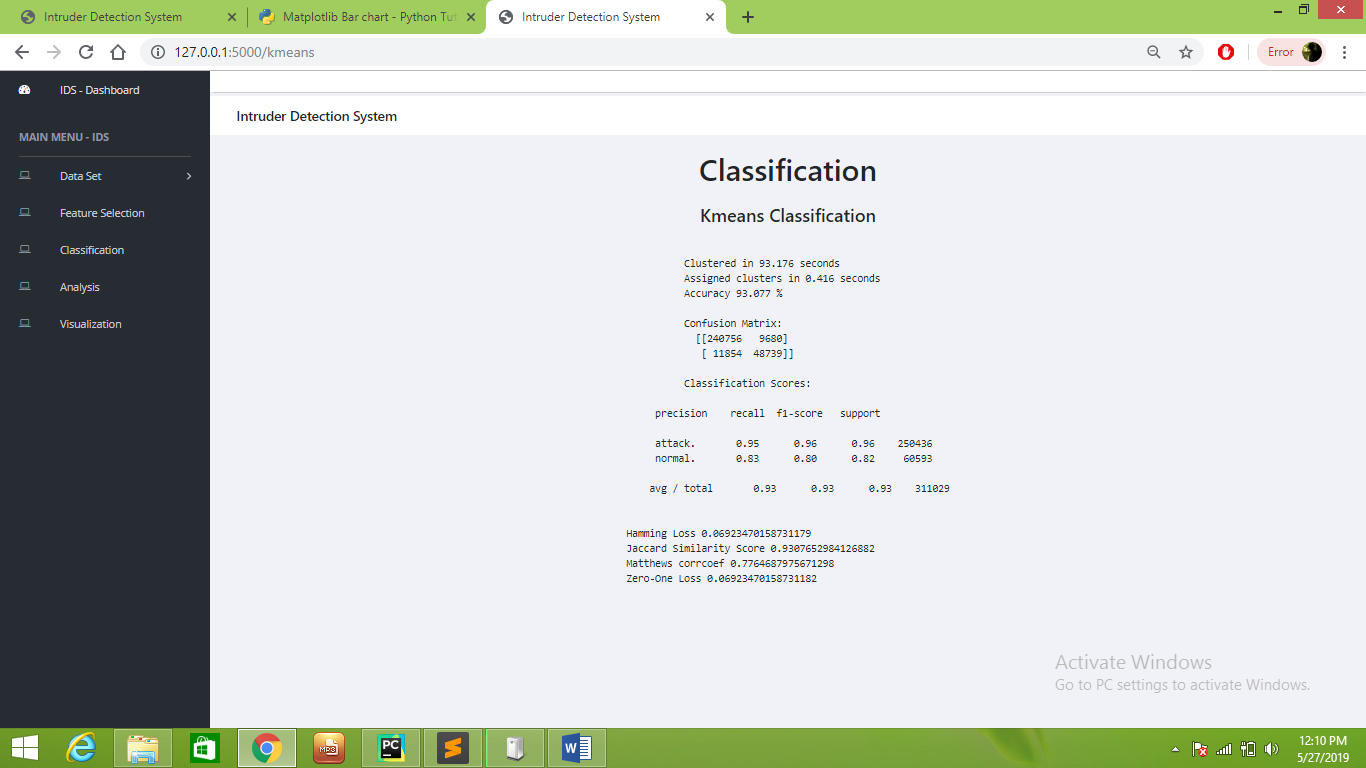
**Figure 6:** Screenshot showing the result from Decision Tree Classifier

The screen above shows the output when Decision Tree Classifier is used.



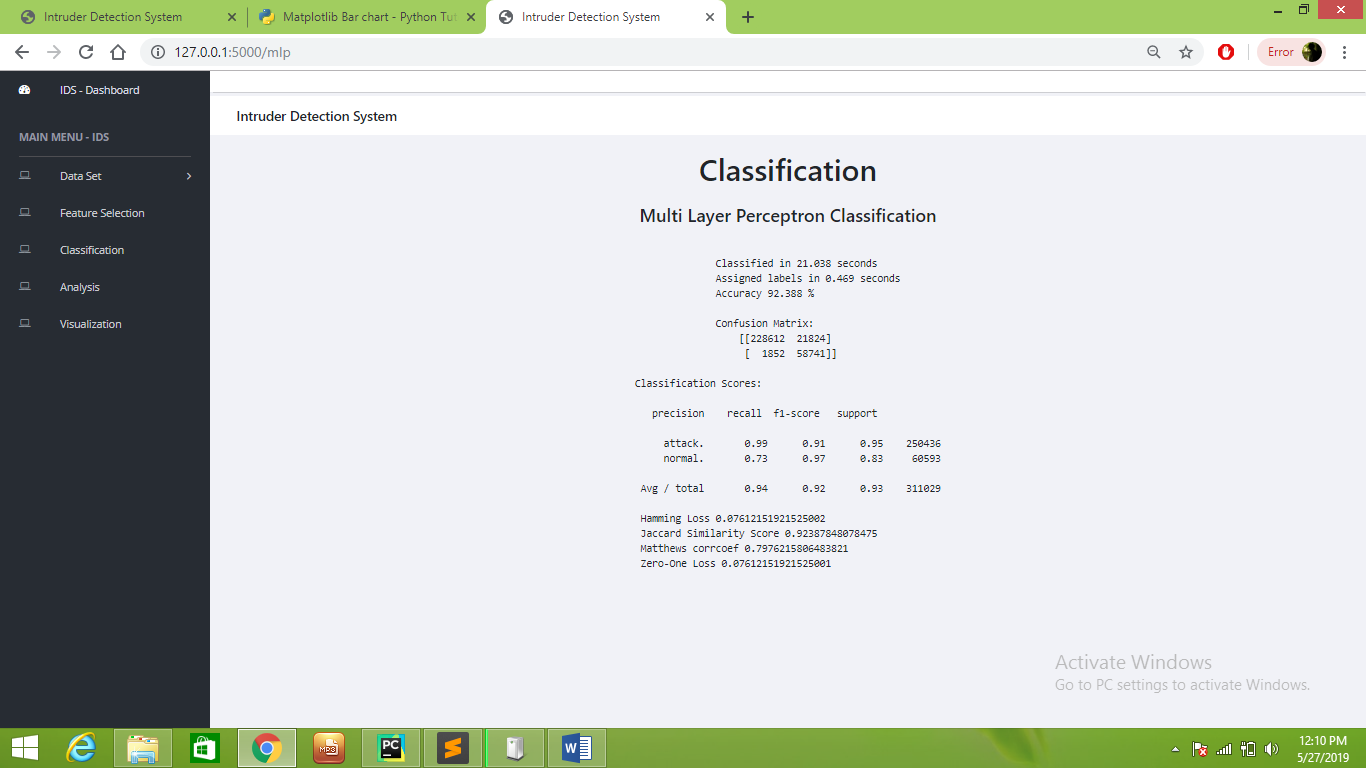
**Figure 7:** Screenshot showing the result from Random Forest Classifier.

The screen above shows the output when Random Forest Classifier is used.



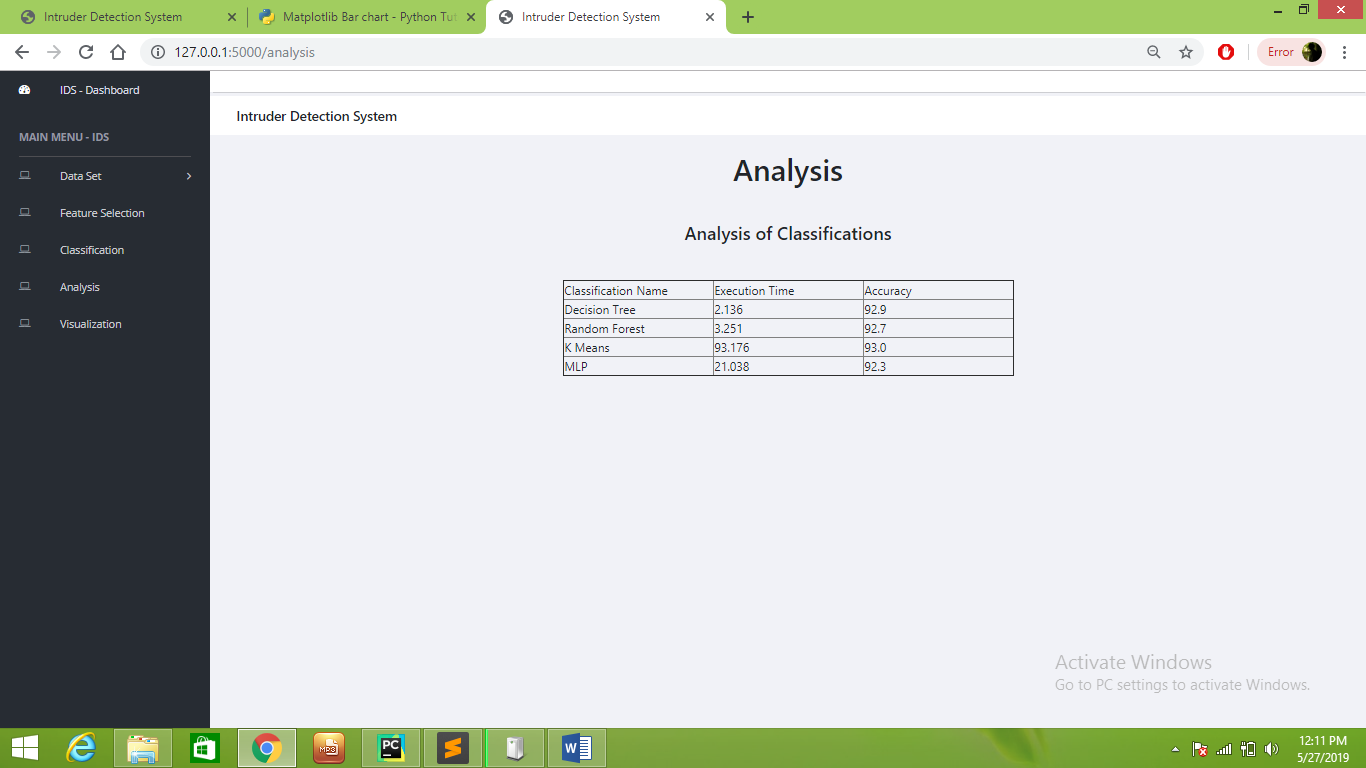
**Figure 8:** Screenshot showing result from K – Means Classifier.

The screen above shows the output when K – Means Classifier is used.



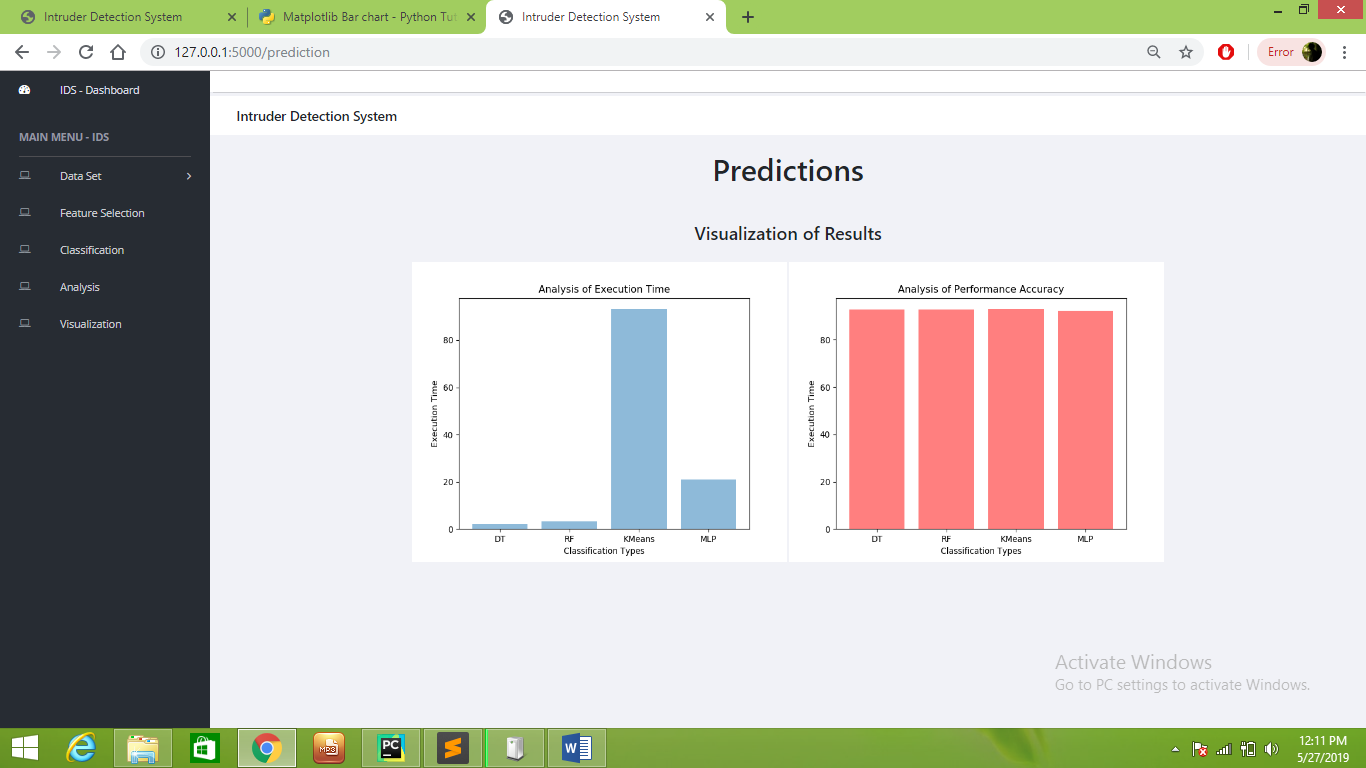
**Figure 9:** Screenshot showing results from Multilayer Perceptron Classifier

The screen above shows the output when Multilayer Perceptron Classifier is used.



**Figure 10:** Screenshot showing the Analysis of Classifiers

The above table shows the execution time and accuracy of the system. The accuracy is calculated by executing various classifiers on KDD Cup Data Set.



**Figure 11:** Screenshot showing the prediction and visualization of the outputs.

The above screen shows the visualization of the results. The visualization consists of the graph showing Analysis of Execution Time and Analysis of Performance Accuracy.

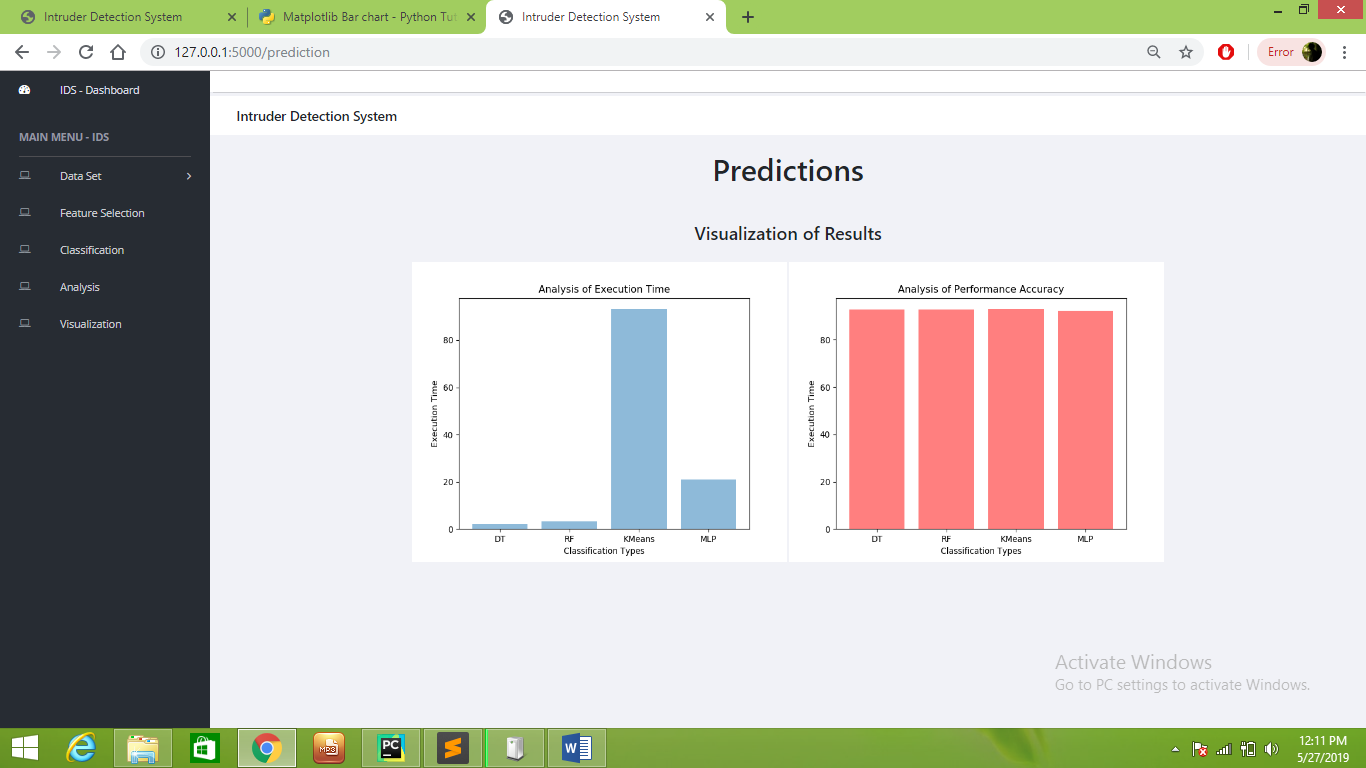
**RESULT**

This intrusion detection system works on four classifiers namely:

1. Random Forest Classifier
2. Decision Tree Classifier
3. Multilayer Perceptron Classifier
4. K – Means Classifier

The project is evaluated for all these classifiers to find out the best available classifier for intrusion detection. It is found out that the Decision Tree classifier is the best classifier that is available for Intrusion Detection. The intrusion detection works well in time as well as space. The execution time and accuracy are the two points that are considered to determine which system is best and prevents the system from intrusion. It is found out the the Execution time of the Decision Tree Classifier is the least while the accuracy is highest. It requires 2.136 milli-seconds which providing the accuracy of 92.9 %. Whereas the Random Forest requires 3.251 milli-seconds with an accuracy of 92.7 %.

The graph and table below shows the execution result and accuracy of different Classifiers.



**Figure 12: Visualization of Results**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No.** | **Classification Name** | **Execution Time** | **Accuracy** |
| 1 | Decision Tree | 2.136 | 92.9 |
| 2 | Random Forest | 3.251 | 92.7 |
| 3 | K Means | 93.176 | 93.0 |
| 4 | MLP | 21.038 | 92.3 |

**Table 1: Execution Time and Accuracy of different Classifiers**

**Advantages**

* The proposed system is effective in preventing the system from intrusion attack.
* The system uses 4 classifiers and all of them work efficiently.

**Disadvantages**

* The proposed method is carried on readily available data set so the system is not tested in real time.

**CONCLUSION**

In this project, an Intrusion Detection system is designed that helps to detect and prevent the intrusion attack on a particular system. The Intrusions need to be prevented in order to avoid various infections in the system. This project uses 4 classifiers which are:

1. Random Forest Classifier
2. Decision Tree Classifier
3. Multilayer Perceptron Classifier
4. K – Means Classifier

These classifiers help to detect the intrusion which is ongoing in the system. The results are generated and it is found out that the decision tree classifier is the best one in this Intrusion Detection System. This system proves to be effective in preventing the system from intrusion attack.

**FUTURE SCOPE**

The main aim of this project is to prevent the system from intrusion attack and make the system more secure. This proposed system works well for the specified task of intrusion prevention and it is found out that a specific classifier works well in it. This project uses 4 classifiers and it is found out that the decision tree classifier produces the best result. But this result is evaluated as per the KDD Cup Data Set which is already available on the Internet. So the Future Work of this project will include the analysis of the project on real data set captured at real times. This can further lead to the modifications of this project making it better, more reliable and secure.

**REFERENCES**

[1] Elhadj Benkhelifa, Thomas Welsh, Walaa Hamouda, “A Critical Review of Practices and Challenges in Intrusion Detection Systems for IoT: Towards Universal and Resilient Systems” IEEE Communications Surveys & Tutorials, 2018.

[2] John E. Dickerson, Julie A. Dickerson, “Fuzzy Network Profiling for Intrusion Detection”, IEEE, 2000.

[3] Preeti Aggarwal, Sudhir Kumar Sharma, “Analysis of KDD Dataset Attributes - Class wise For Intrusion Detection”, Procedia Computer Science, 2015.

[4] Nabila Farnaaz, M. A. Jabbar, “Random Forest Modeling for Network Intrusion Detection System”, Procedia Computer Science, 2016.

[5] F.Sabahi, A.Movaghar, “Intrusion Detection: A Survey”, IEEE, 2008.

[6] Fadi Salo, Mohammad Noor Injadat, Ali Bou Nassif, Abdalla Shami, Aleksandar Essex, “Data Mining Techniques in Intrusion Detection Systems: A Systematic Literature Review”, IEEE Access, 2016.

[7] Hichem Sedjelmaci, Sidi Mohammed Senouci, Nirwan Ansari, “A Hierarchical Detection and Response System to Enhance Security Against Lethal Cyber-Attacks in UAV Networks”, IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS, 2017.

[8] Alex Shenfield, David Day, Aladdin Ayesh, “Intelligent intrusion detection systems using artificial neural networks”, The Korean Institute of Communications and Information Sciences (KICS) (Science Direct), 2018.

[9] Kehe Wu, Zuge Chen, Wei Li, “A Novel Intrusion Detection Model for a Massive Network Using Convolutional Neural Networks”, IEEE Access, 2017.

[10] Zhihua Zhang, Hongliang Zhu, Shoushan Luo, Yang Xin, Xiaoming Liu, “Intrusion Detection Based on State Context and Hierarchical Trust in Wireless Sensor Networks”, IEEE Access, 2016.

[11] Bhushan H. Talan, H. R. Vyawahare, “Intrusion Detection System with Machine Learning Techniques for Network Data Security”, 2019.

**Publications**

[1] Bhushan H. Talan, H. R. Vyawahare, “Intrusion Detection System with Machine Learning Techniques for Network Data Security”, IJRSD ( Online Journal) 2019.