



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

Detection of Distributed Denial of Service Attacks in SDN using Machine Learning Techniques

Submitted in partial fulfilment of the requirements for the award of Master
of Engineering in Computer Science and Engineering.

by

Sanneboina Karthik (40310002)



**DEPARTMENT OF COMPUTER SCIENCE AND
ENGINEERING SCHOOL OF COMPUTING**

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI-600119

MAY - 2022

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **SANNEBOINA KARTHIK (40310002)** who carried out the project entitled **“Detection of Distributed Denial of Service Attacks in SDN using Machine Learning Techniques”** under my / our supervision from **November 2021 to May 2022.**

Internal Guide

Dr. G. Mathivanan B.Sc., B.Tech., M.E., Ph.D.

Head of the Department

Dr.S.Vigneshwari M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **Sanneboina Karthik** hereby declare that the Project Report entitled **Detection of Distributed Denial of Service Attacks in SDN using machine learning techniques** done by me under the guidance of **Dr. G. Mathivanan B.Sc., B.Tech., M.E., Ph.D., Sathyabama Institute of Science and Technology** Jeppiaar Nagar, RajivGandhi Salai, Chennai- 600119 is submitted in partial fulfilment of the requirements for the award of Master of Engineering in Computer Science and Technology

SIGNATURE OF THE CANDIDATE

DATE:

PLACE:

ACKNOWLEDGEMENT

I am pleased to submit my sincere thanks to Board of Management of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my gratitude to **Dr. T. Sasikala M.E., Ph.D., Dean**, School of Computing, **Dr.S. Vigneshwari M.E., Ph.D.**, and **Dr.L.Lakshmanan M.E., Ph.D.**, Head of the Department, Dept. of Computer Science and Engineering for providing me necessary support and details at the right time during the progress reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. G. Mathivanan B.Sc., B.Tech., M.E., Ph.D.**, for his valuable guidance, suggestions, and constant encouragement which paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT OF THE WORK

Software-defined network (SDN) is a network architecture that used to build, design the hardware components virtually. We can dynamically change the settings of network connections. In the traditional network, it's not possible to change dynamically, because it's a fixed connection. SDN is a good approach but still is vulnerable to DDoS attacks. The DDoS attack is menacing to the internet. To prevent the DDoS attack, the machine learning algorithm can be used. The DDoS attack is the multiple collaborated systems that are used to target a particular server at the same time. In SDN control layer is in the center that link with the application and infrastructure layer, where the devices in the infrastructure layer are controlled by software. In this project, I have used the machine learning techniques namely Random Forest and Support Vector Machine (SVM) to detect malicious traffic. The test outcome shows that the Random Forest and Support Vector Machine (SVM) algorithm provides better accuracy and detection rate.

It separates the control plane from the forwarding plane which enables administrators to define and control the network through the method of software programming, provides a new research direction for the next generation of network architecture.

CONTENTS WITH PAGE NUMBERS

TABLE OF CONTENT

CHAPTER NO	TITLE	PAGE NO
I.	INTRODUCTION	1
II.	LITERATURE SURVEY	3
III.	AIM	8
	3.1 OBJECTIVE	7
	3.2 SCOPE OF THE PRESENT INVESTIGATION	7
	3.3 EXISTING SYSTEM	7
	3.4 PROPOSED SYSTEM	9
IV.	EXPERIMENTAL OR MATERIALS AND METHODS	10
	4.1 PYTHON OVERVIEW	10
	4.2 PYTHON ENVIRONMENT	12
	4.3 METHODOLOGY	24
V.	RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS	49
VI.	SUMMARY AND CONCLUSIONS	52

LIST OF FIGURES

Figure1: Line Chart	20
Figure 2: Histogram	20
Figure 3: Bar-Chart	21
Figure: 4 Matplotlib	22
Figure :5 System Architecture	23
Figure: 6 Use Case Diagram	25
Figure :7 Class Diagram	26
Figure :8 Sequence Diagram	27
Figure :9 Activity Diagram	28
Figure 10: Machine Learning vs. Traditional Programming	31
Figure 11: Learning Phase	32
Figure 12: Inference from Model	33
Figure13: Machine Learning Algorithms	34
Figure 14: Graph of AL, ML,DL	44

LIST OF TABLES

Table1: Literature Survey	2
Table 2 : Classification	36
Table 3: Unsupervised learning	37
Table 4: Difference between Machine Learning and Deep Learning	42
Table 5: ML vs DL	43

CHAPTER – 1

Introduction

Software Defined Networking is an emerging paradigm which overcomes the limitations of conventional network architecture by separating the control from data plane devices. SDN consists of three planes such as data plane, control plane and application plane. Data plane carries the network traffic based on the decision made by controller. Control plane decides the flow of traffic by computing the routing tables. Application plane manages the other applications like load balancer, firewalls, Quality of Service (QoS) applications etc.

SDN architecture improves the network performance by decoupling the network control and forward function. The control programs running in a logically centralized controller will control multiple routers across the network.

Everybody is subjected to stress from time to time. When your body feels threatened, it experiences stress. In its most basic form, it prepares the body to confront a threat by fighting or running.

It is an approach to networking that uses software-based controllers or application programming interfaces (APIs) to communicate with underlying hardware infrastructure and direct traffic on a network.

It is distinguished by a racing heart, rapid breathing rate, and acute sensations. All of these modifications would be critical for someone who was preparing for a life-threatening catastrophe. However, because most of us now live in a culture where we rarely confront genuinely life-threatening occurrences, we don't need or want to have these emotions. In truth, most people face much too much stress in their daily lives.

Many people experience stress-related symptoms such as feeling unwell, trapped, overwhelmed, sad, nervous, or panicked.

These are not useful emotions when the capacity to think clearly and sensibly is necessary (and helpful). Many individuals spend much too much time worrying, feeling guilty, or humiliated for not doing things they believe they ought to or should do. The energy spent feeding these negative feelings might be better spent focused on achieving your objectives

To give a faster and less expensive solution to these challenges, an application of artificial intelligence (AI), namely machine learning, may be used, which allows systems to automatically learn and improve from experience without being explicitly designed.

It provide users with access to critical data on an anytime, anywhere scale, while the goal of a Software-Defined Network (SDN) is to allow a network admin to shape, construct and manage network resources – traffic, bandwidth etc.

improves network connectivity for sales, customer service, internal communications, and document sharing. SDN allows organizations to use software and hardware from multiple vendors to have customized network services and infrastructure.

CHAPTER 2

LITERATURE SURVEY

S.NO:	AUTHOUR	TITTLE	YEAR	ALGORITHM	DRAWBACK
1	Gustavode Carvalho Betroli	Network Intrusion Detection System	2021	Machine learning algorithms	Complexity and not accuracy
2	Santos, R., Souza, D., Santo, W., Ribeiro, A	to detect DDoS attacks in SDN	2020	Machine learning algorithms	Concurrency and Computation
3	Dong, S., & Sarem	DDoS Attack Detection Method	2019	KNN	Attack in Software- Defined Networks
4	Gu, Y., Li, K., Guo	SemiS upervised K- means DDoS detection method	2019	hybrid feature selection algorithm	Journal of High Speed Network

Table1: Literature Survey

TITLE: An End-to-End Framework for Machine Learning-Based Network Intrusion Detection System

AUTHOR: Gustavo De Carvalho Bertoli GUSTAVO DE CARVALHO BERTOLI

ABSTRACT: The increase of connected devices and the constantly evolving methods and techniques by attackers pose a challenge for network intrusion detection systems from conception to operation. As a result, we see a constant adoption of machine learning algorithms for network intrusion detection systems. However, the dataset used by these studies has become obsolete regarding both background and attack traffic. This work describes the AB-TRAP framework that enables the use of updated network traffic and considers operational concerns to enable the complete deployment of the solution. AB-TRAP is a five-step framework consisting of (i) the generation of the attack dataset, (ii) the bonafide dataset, (iii) training of machine learning models, (iv) realization (implementation) of the models, and (v) the performance evaluation of the realized model after deployment. We exercised the AB-TRAP for local (LAN) and global (internet) environments to detect TCP port scanning attacks. The LAN study case presented an f1-score of 0.96, and an area under the ROC curve of 0.99 using a decision tree with minimal CPU and RAM usage on kernel-space. For the internet case with eight machine learning algorithms with an average f1-score of 0.95, an average area under the ROC curve of 0.98, and an average overhead of 1.4% CPU and 3.6% RAM on user-space in a single-board computer. This framework has the following paramount characteristics: it is reproducible, uses the most up-to-date network traffic, attacks, and addresses the concerns to the model's realization and deployment.

TITLE: k-Zero Day Safety: A Network Security Metric for Measuring the Risk of Unknown Vulnerabilities.

AUTHOR: L. Wang, M. Zhang, and A. Singhal

ABSTRACT: By enabling a direct comparison of different security solutions with respect to their relative effectiveness, a network security metric may provide quantifiable evidences to assist security practitioners in securing computer networks. However, research on security metrics has been hindered by difficulties in handling zero day attacks exploiting unknown vulnerabilities. In fact, the security risk of unknown vulnerabilities has been considered as something unmeasurable due to the

less predictable nature of software flaws. This causes a major difficulty to security metrics, because a more secure configuration would be of little value if it were equally susceptible to zero day attacks. In this paper, we propose a novel security metric, k-zero day safety, to address this issue. Instead of attempting to rank unknown vulnerabilities, our metric counts how many such vulnerabilities would be required for compromising network assets; a larger count implies more security since the likelihood of having more unknown vulnerabilities available, applicable, and exploitable all at the same time will be significantly lower.

We formally define the metric, analyse the complexity of computing the metric, devise heuristic algorithms for intractable cases, and finally demonstrate through case studies that applying the metric to existing network security practices may generate actionable knowledge.

TITLE: A Survey on Network Security-Related Data Collection Technologies.

AUTHOR: Lin, Huaqing ; Yan, Zheng ; Chen, Yu ; Zhang, Lifang

ABSTRACT: Security threats and economic loss caused by network attacks, intrusions and vulnerabilities have motivated intensive studies on network security. Normally, data collected in a network system can reflect or can be used to detect security threats. We define these data as network security-related data. Studying and analyzing security-related data can help detect network attacks and intrusions, thus making it possible to further measure the security level of the whole network system. Obviously, the first step in detecting network attacks and intrusions is to collect security-related data. However, in the context of big data and 5G, there exist a number of challenges in collecting these security-related data. In this paper, we first briefly introduce network security-related data, including its definition and characteristics, and the applications of network data collection. We then provide the requirements and objectives for security-related data collection and present a taxonomy of data collection technologies. Moreover, we review existing collection nodes, collection tools and collection mechanisms in terms of network data collection and analyze them

based on the proposed requirements and objectives towards high quality security-related data collection. Finally, we discuss open research issues and conclude with suggestions for future research directions.

TITLE: Intrusion Detection System Modeling Based on Learning from Network Traffic Data.

AUTHOR: A. Midzic, Z. Avdagic, and S. Omanovic

ABSTRACT: This research uses artificial intelligence methods for computer network intrusion detection system modeling. Primary classification is done using self-organized maps (SOM) in two levels, while the secondary classification of ambiguous data is done using Sugeno type Fuzzy Inference System (FIS). FIS is created by using Adaptive Neuro-Fuzzy Inference System (ANFIS). The main challenge for this system was to successfully detect attacks that are either unknown or that are represented by very small percentage of samples in training dataset. Improved algorithm for SOMs in second layer and for the FIS creation is developed for this purpose. Number of clusters in the second SOM layer is optimized by using our improved algorithm to minimize amount of ambiguous data forwarded to FIS. FIS is created using ANFIS that was built on ambiguous training dataset clustered by another SOM (which size is determined dynamically). Proposed hybrid model is created and tested using NSL KDD dataset. For our research, NSL KDD is especially interesting in terms of class distribution (overlapping). Objectives of this research were: to successfully detect intrusions represented in data with small percentage of the total traffic during early detection stages, to successfully deal with overlapping data (separate ambiguous data), to maximize detection rate (DR) and minimize false alarm rate (FAR). Proposed hybrid model with test data achieved acceptable DR value 0.8883 and FAR value 0.2415. The objectives were successfully achieved as it is presented (compared with the

similar researches on NSL KDD dataset). Proposed model can be used not only in further research related to this domain, but also in other research areas.

TITLE: A parallel sampling method for Bayesian networks.

AUTHOR: Y. Kobari and M. Kimura

ABSTRACT: Aberrant intracellular signaling plays an important role in many diseases. The causal structure of signal transduction networks can be modeled as Bayesian Networks (BNs), and computationally learned from experimental data. However, learning the structure of Bayesian Networks (BNs) is an NP-hard problem that, even with fast heuristics, is too time consuming for large, clinically important networks (20–50 nodes). In this paper, we present a novel graphics processing unit (GPU)-accelerated implementation of a Monte Carlo Markov Chain-based algorithm for learning BNs that is up to 7.5-fold faster than current general-purpose processor (GPP)-based implementations.

The GPU-based implementation is just one of several implementations within the larger application, each optimized for a different input or machine configuration. We describe the methodology we use to build an extensible application, assembled from these variants, that can target a broad range of heterogeneous systems, e.g., GPUs, multicore GPPs. Specifically we show how we use the Merge programming model to efficiently integrate, test and intelligently select among the different potential implementations.

CHAPTER 3

AIM:

The main aim of the project is to identify the DDoS attack to using SDN using ML techniques.

3.1 OBJECTIVE:

Identifying DDoS attacks using ML in SDN. I have used SVM and Random Forest algorithms to detect the attacks due to its accurate classification and less complexity.

3.2 SCOPE OF THE PRESENT INVESTIGATION

The SDN provides the sole ability to the applications to get to know the entire network information. During high traffic, the integration of different applications helps for load balancing and intrusion detection. If an anomaly is detected, the controller is instructed by the application to reprogram the data plane to alleviate it. Both control and data plane run on routers that are distributed across the network, where the devices have open interfaces that can be controlled by the software.

3.3 EXISTING SYSTEM:

In this project, various research works done in the field of SDN to detect the DDoS attacks have been analyzed. It is found that there are various techniques to avert the DDoS attack i.e. Naive Bayes, KNN, Neural Network, SVM, SOM. In the proposed work, Support Vector Machine (SVM) and Random forest algorithms are used to detect the DDoS attacks by analyzing the essential features of traffic

DISADVANTAGES:

- The experiment result shows the not accuracy
- More complexity

3.4 PROPOSED SYSTEM:

Identifying DDoS attacks using ML in SDN.I have used SVM and Random forest algorithms to detect the attacks due to its accurate classification and less complexity

ADVANTAGES:

- Accuracy
- To detect the attacks due to its accurate classification
- Less complexity

CHAPTER 4

EXPERIMENTAL OR MATERIALS AND METHODS; ALGORITHMS USED

4.1 PYTHON OVERVIEW

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **Python is Interpreted:** Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive:** You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented:** Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language:** Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, Unix shell, and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

Python Features

Python's features include:

- ❑ **Easy-to-learn:** Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- ❑ **Easy-to-read:** Python code is more clearly defined and visible to the eyes.
- ❑ **Easy-to-maintain:** Python's source code is fairly easy-to-maintain.
- ❑ **A broad standard library:** Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- ❑ **Interactive Mode:** Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- ❑ **Portable:** Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- ❑ **Extendable:** You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- ❑ **Databases:** Python provides interfaces to all major commercial databases.
- ❑ **GUI Programming:** Python supports GUI applications that can be created and ported to many system calls, libraries, and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- ❑ **Scalable:** Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below:

- ❑ IT supports functional and structured programming methods as well as OOP.

- ❑ It can be used as a scripting language or can be compiled to byte-code for building large applications.
- ❑ It provides very high-level dynamic data types and supports dynamic type checking.
- ❑ IT supports automatic garbage collection.
- ❑ It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

4.2 PYTHON ENVIRONMENT

Python is available on a wide variety of platforms including Linux and Mac OS X. Let's understand how to set up our Python environment.

Python's standard library

- Pandas
- Numpy
- Sklearn
- seaborn
- matplotlib
- Importing Datasets

PANDAS:

Pandas is quite a game changer when it comes to analyzing data with Python and it is one of the most preferred and widely used tools in data munging/wrangling if not THE most used one. Pandas is an open source

What's cool about Pandas is that it takes data (like a CSV or TSV file, or a SQL database) and creates a Python object with rows and columns called data frame that looks very similar to table in a statistical software (think Excel or SPSS for example. People who are familiar with R would see similarities to R too). This is so much easier

to work with in comparison to working with lists and/or dictionaries through for loops or list comprehension.

Installation and Getting Started

In order to “get” Pandas you would need to install it. You would also need to have Python 2.7 and above as a pre-requirement for installation. It is also dependent on other libraries (like NumPy) and has optional dependencies (like Matplotlib for plotting). Therefore, I think that the easiest way to get Pandas set up is to install it through a package like the Anaconda distribution , “a cross platform distribution for data analysis and scientific computing.”

In order to use Pandas in your Python IDE (Integrated Development Environment) like Jupyter Notebook or Spyder (both of them come with Anaconda by default), you need to import the Pandas library first. Importing a library means loading it into the memory and then it’s there for you to work with. In order to import Pandas all you have to do is run the following code:

- **import pandas as pd**
- **import numpy as np**

Usually you would add the second part (‘as pd’) so you can access Pandas with ‘pd.command’ instead of needing to write ‘pandas.command’ every time you need to use it. Also, you would import numpy as well, because it is very useful library for scientific computing with Python. Now Pandas is ready for use! Remember, you would need to do it every time you start a new Jupyter Notebook, Spyder file etc.

Working with Pandas

Loading and Saving Data with Pandas

When you want to use Pandas for data analysis, you’ll usually use it in one of three different ways:

- Convert a Python's list, dictionary or Numpy array to a Pandas data frame
- Open a local file using Pandas, usually a CSV file, but could also be a delimited text file (like TSV), Excel, etc
- Open a remote file or database like a CSV or a JSON on a website through a URL or read from a SQL table/database

There are different commands to each of these options, but when you open a file, they would look like this:

- **pd.read_filetype()**

As I mentioned before, there are different filetypes Pandas can work with, so you would replace "filetype" with the actual, well, filetype (like CSV). You would give the path, filename etc inside the parenthesis. Inside the parenthesis you can also pass different arguments that relate to how to open the file.

There are numerous arguments and in order to know all you them, you would have to read the documentation (for example, the documentation for `pd.read_csv()` would contain all the arguments you can pass in this Pandas command).

To convert a certain Python object (dictionary, lists etc) the basic command is:

- **pd.DataFrame()**

Inside the parenthesis you would specify the object(s) you're creating the data frame from. This command also has different arguments .

You can also save a data frame you're working with/on to different kinds of files (like CSV, Excel, JSON and SQL tables). The general code for that is:

- `df.to_filetype(filename)`

Viewing and Inspecting Data

Now that you've loaded your data, it's time to look. How does the data frame look? Running the name of the data frame would give you the entire table, but you can also get the first *n* rows with `df.head(n)` or the last *n* rows with `df.tail(n)`. `df.shape` would give you the number of rows and columns. `df.info()` would give you the index, datatype and memory information. The command `s.value_counts(dropna=False)` would allow you to view unique values and counts for a series (like a column or a few columns). A very useful command is `df.describe()` which inputs summary statistics for numerical columns. It is also possible to get statistics on the entire data frame or a series (a column etc):

- `df.mean()` Returns the mean of all columns
- `df.corr()` Returns the correlation between columns in a data frame
- `df.count()` Returns the number of non-null values in each data frame column
- `df.max()` Returns the highest value in each column
- `df.min()` Returns the lowest value in each column
- `df.median()` Returns the median of each column
- `df.std()` Returns the standard deviation of each column

Selection of Data

One of the things that is so much easier in Pandas is selecting the data you want in comparison to selecting a value from a list or a dictionary. You can select a column (`df[col]`) and return column with label `col` as Series or a few columns (`df[[col1, col2]]`) and returns columns as a new DataFrame. You can select by position (`s.iloc[0]`), or by index (`s.loc['index_one']`) . In order to select the first row you can use `df.iloc[0,:]` and in order to select the first element of the first column you would run `df.iloc[0,0]` . These can also be used in different combinations, so I hope it gives you an idea of the different selection and indexing you can perform in Pandas.

Filter, Sort and Groupby

You can use different conditions to filter columns. For example, `df[df['year'] > 1984]` would give you only the column year is greater than 1984. You can use `&` (and) or `|` (or) to add different conditions to your filtering. This is also called boolean filtering. It is possible to sort values in a certain column in an ascending order using `df.sort_values(col1)`; and also in a descending order using `df.sort_values(col2,ascending=False)`. Furthermore, it's possible to sort values by `col1` in ascending order then `col2` in descending order by using `df.sort_values([col1,col2],ascending=[True,False])`.

The last command in this section is `groupby`. It involves splitting the data into groups based on some criteria, applying a function to each group independently and combining the results into a data structure. `df.groupby(col)` returns a `groupby` object for values from one column while `df.groupby([col1,col2])` returns a `groupby` object for values from multiple columns.

Data Cleaning

Data cleaning is a very important step in data analysis. For example, we always check for missing values in the data by running `pd.isnull()` which checks for null Values, and returns a boolean array (an array of true for missing values and false for non-missing values). In order to get a sum of null/missing values, run `pd.isnull().sum()`. `pd.notnull()` is the opposite of `pd.isnull()`. After you get a list of missing values you can get rid of them, or drop them by using `df.dropna()` to drop the rows or `df.dropna(axis=1)` to drop the columns. A different approach would be to fill the missing values with other values by using `df.fillna(x)` which fills the missing values with `x` (you can put there whatever you want) or `s.fillna(s.mean())` to replace all null values with the mean (mean can be replaced with almost any function from the statistics section).

It is sometimes necessary to replace values with different values. For example, `s.replace(1,'one')` would replace all values equal to 1 with 'one'. It's possible to do it for multiple values: `s.replace([1,3],['one','three'])` would replace all 1 with 'one' and 3 with 'three'. You can also rename specific columns by running: `df.rename(columns={'old_name': 'new_name'})` or use `df.set_index('column_one')` to change the index of the data frame.

Join/Combine

The last set of basic Pandas commands are for joining or combining data frames or rows/columns. The three commands are:

- `df1.append(df2)`— add the rows in df1 to the end of df2 (columns should be identical)
- `df.concat([df1, df2],axis=1)`—add the columns in df1 to the end of df2 (rows should be identical)
- `df1.join(df2,on=col1,how='inner')`—SQL-style join the columns in df1 with the columns on df2 where the rows for col have identical values. how can be equal to one of: 'left', 'right', 'outer', 'inner'

NUMPY

Numpy is one such powerful library for array processing along with a large collection of high-level mathematical functions to operate on these arrays. These functions fall into categories like Linear Algebra, Trigonometry, Statistics, Matrix manipulation, etc.

Getting NumPy

NumPy's main object is a homogeneous multidimensional array. Unlike python's array class which only handles one-dimensional array, NumPy's ndarray class can handle multidimensional array and provides more functionality. NumPy's dimensions are known as axes. For example, the array below has 2 dimensions or 2 axes namely rows and columns. Sometimes dimension is also known as a rank of that particular array or matrix.

Importing NumPy

NumPy is imported using the following command. Note here np is the convention followed for the alias so that we don't need to write numpy every time.

- `import numpy as np`

NumPy is the basic library for scientific computations in Python and this article illustrates some of its most frequently used functions. Understanding NumPy is the first major step in the journey of machine learning and deep learning.

Sklearn

In python, scikit-learn library has a pre-built functionality under sklearn. Pre processing.

Next thing is to do feature extraction Feature extraction is an attribute reduction process. Unlike feature selection, which ranks the existing attributes according to their predictive significance, feature extraction actually transforms the attributes. The transformed attributes, or features, are linear combinations of the original attributes. Finally our models are trained using Classifier algorithm.. We use nltk . classify module on Natural Language Toolkit library on Python. We use the labelled dataset gathered . The rest of our labelled data will be used to evaluate the models. Some machine learning algorithms were used to classify pre processed data. The chosen classifiers were Decision tree , Support Vector Machines and Random forest. These algorithms are very popular in text classification tasks.

SEABORN

Data Visualization in Python

Data visualization is the discipline of trying to understand data by placing it in a visual context, so that patterns, trends and correlations that might not otherwise be detected can be exposed.

Python offers multiple great graphing libraries that come packed with lots of different features. No matter if you want to create interactive, live or highly customized plots python has a excellent library for you.

To get a little overview here are a few popular plotting libraries:

- **Matplotlib:** low level, provides lots of freedom
- **Pandas Visualization:** easy to use interface, built on Matplotlib
- **Seaborn:** high-level interface, great default styles
- **ggplot:** based on R's ggplot2, uses Grammar of Graphics
- **Plotly:** can create interactive plots

In this article, we will learn how to create basic plots using Matplotlib, Pandas visualization and Seaborn as well as how to use some specific features of each library. This article will focus on the syntax and not on interpreting the graphs.

Matplotlib

Matplotlib is the most popular python plotting library. It is a low level library with a Matlab like interface which offers lots of freedom at the cost of having to write more code.

1. To install Matplotlib pip and conda can be used.
2. pip install matplotlib
3. conda install matplotlib

Matplotlib is specifically good for creating basic graphs like line charts, bar charts, histograms and many more. It can be imported by typing:

- **import matplotlib.pyplot as plt**

Line Chart

In Matplotlib we can create a line chart by calling the `plot` method. We can also plot multiple columns in one graph, by looping through the columns we want, and plotting each column on the same axis.

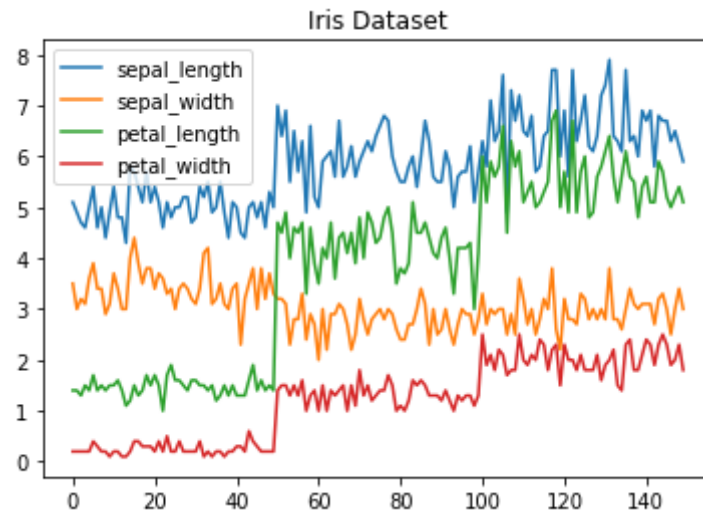


Figure1:Line Chart

Histogram

In Matplotlib we can create a Histogram using the `hist` method. If we pass it categorical data like the points column from the wine-review dataset it will automatically calculate how often each class occurs.

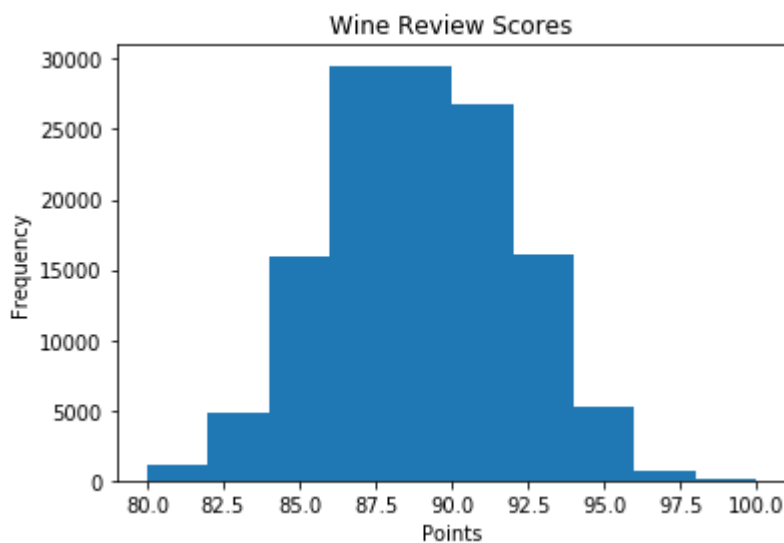


Figure 2: Histogram

Bar Chart

A bar-chart can be created using the bar method. The bar-chart isn't automatically calculating the frequency of a category so we are going to use pandas value_counts function to do this. The bar-chart is useful for categorical data that doesn't have a lot of different categories (less than 30) because else it can get quite messy.

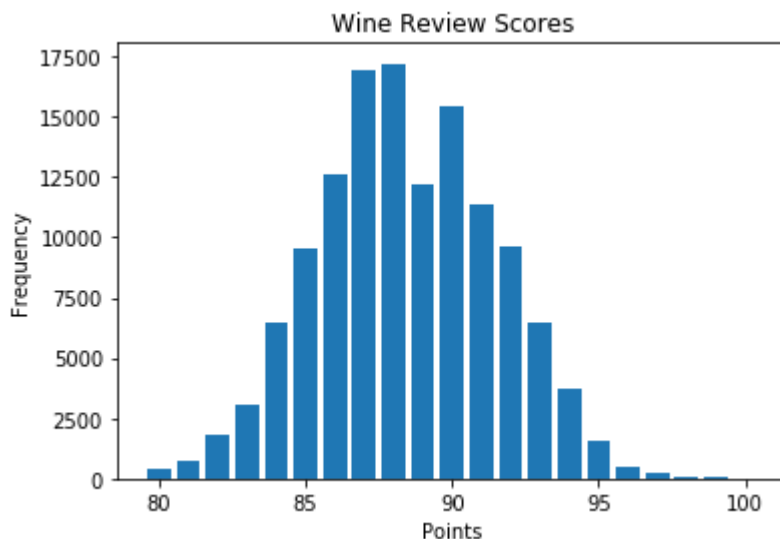


Figure 3: Bar-Chart

Pandas Visualization

Pandas is a open source high-performance, easy-to-use library providing data structures, such as data frames, and data analysis tools like the visualization tools we will use in this article.

Pandas Visualization makes it really easy to create plots out of a pandas dataframe and series. It also has a higher level API than Matplotlib and therefore we need less code for the same results.

1. Pandas can be installed using either pip or conda.

2. pip install pandas

3. conda install pandas

Heatmap

A Heatmap is a graphical representation of data where the individual values contained in a matrix are represented as colors. Heatmaps are perfect for exploring the correlation of features in a dataset.

To get the correlation of the features inside a dataset we can call `<dataset>.corr()`, which is a Pandas dataframe method. This will give use the correlation matrix.

We can now use either Matplotlib or Seaborn to create the heatmap.

Matplotlib:

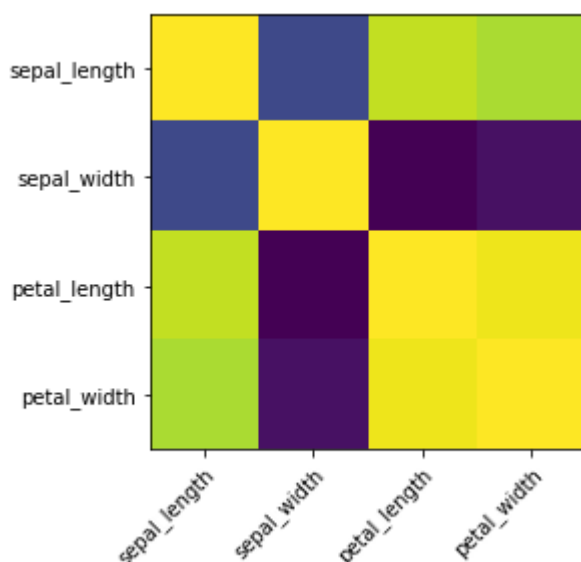


Figure :4 Matplotlib

Heatmap without annotations

Data visualization is the discipline of trying to understand data by placing it in a visual context, so that patterns, trends and correlations that might not otherwise be detected can be exposed.

Python offers multiple great graphing libraries that come packed with lots of different features. In this article we looked at Matplotlib, Pandas visualization and Seaborn.

SYSTEM ARCHITECTURE:

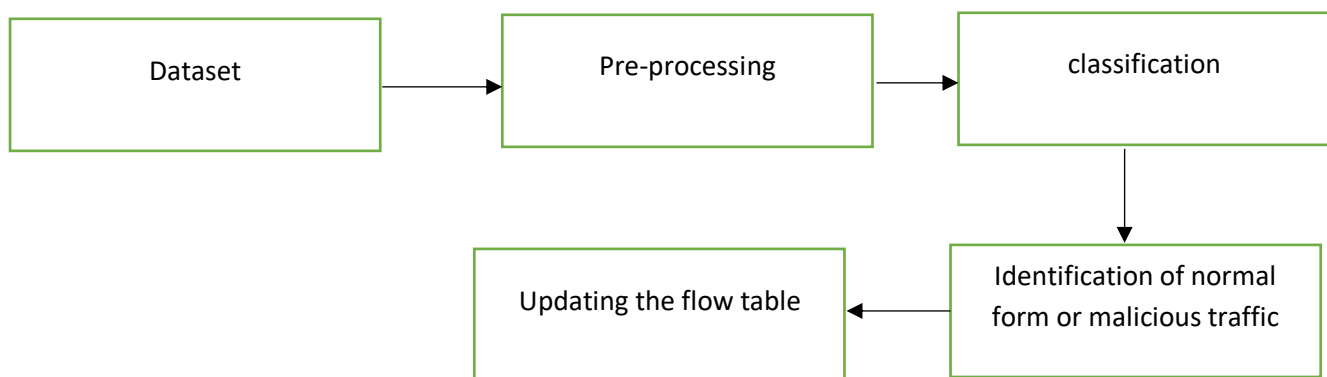


Figure :5 System Architecture

4.3 METHODOLOGY

Modules:

1. Dataset and Pre-processing:

We needed dataset of fake and genuine profiles. Various attributes included in dataset are number of friends, followers, status count. Dataset is divided into training and testing data. Classification algorithms are trained using training dataset and testing dataset is used to determine efficiency of algorithm. From the dataset used, 80% of both profiles (genuine and fake) are used to prepare a training dataset and 20% of both profiles are used to prepare a testing dataset.

2. Feature Selection:

Features are selected to apply classification algorithms. The classification algorithm is discussed further. Attributes are selected as features if they are not dependent on other attributes and they increase efficiency of the classification. The features that we have chosen are discussed further.

After selection of attributes, the dataset of profiles that are already classified as fake or genuine are needed for the training purpose of the classification algorithm. We have used a publicly available dataset of 1337 fake users and 1481 genuine users consisting of various attributes including name, status count, number of friends, followers count, favourites, languages known etc.

3. Classification:

Classification is the process of categorizing a data object into categories called classes based upon features/attributes associated with that data object. Classification uses a classifier, an algorithm that processes the attributes of each data object and outputs a class based upon this information. In this project, we use Support Vector Machine as a classifier. Support Vector Machine is an elegant and robust technique for classification on a large data set not unlike the data sets of Social Network with several millions of profiles.

USECASE DIAGRAM

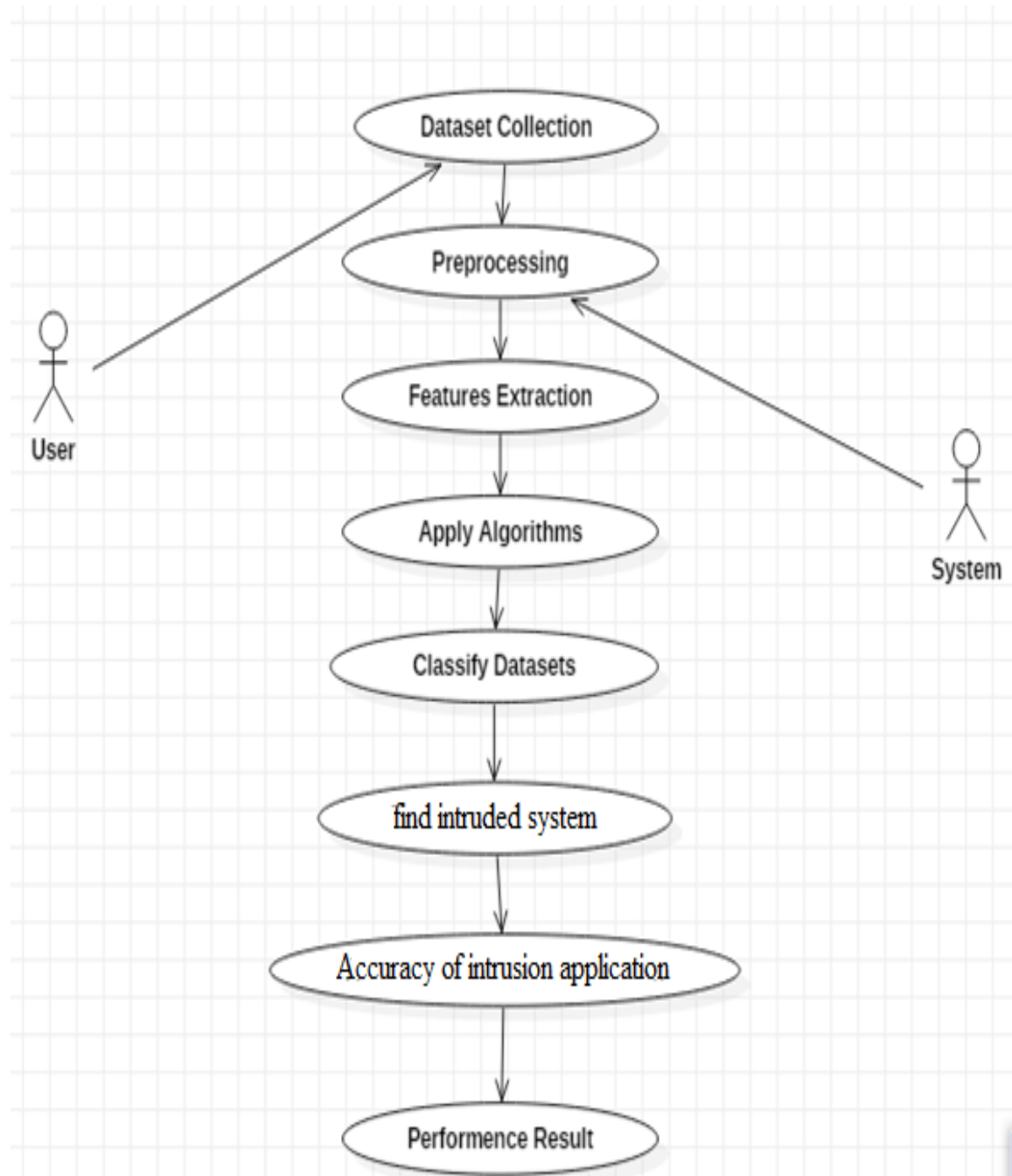


Figure: 6 Use Case Diagram

CLASS DIAGRAM:

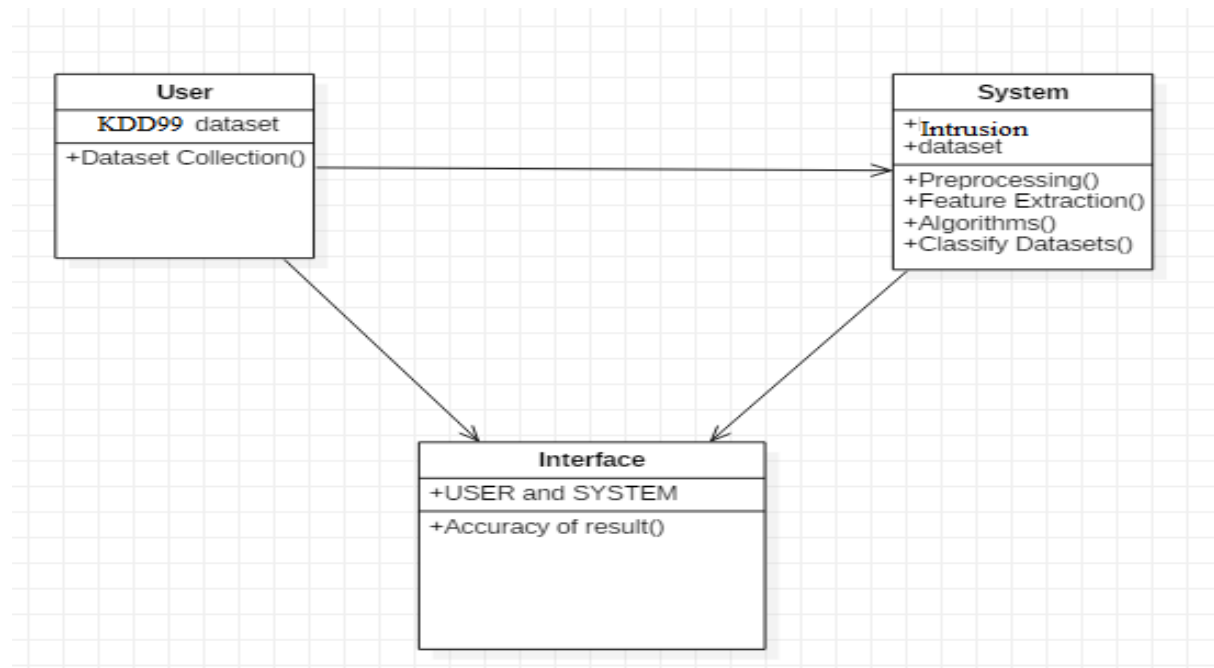


Figure :7 Class Diagram

SEQUENCE DIAGRAM:

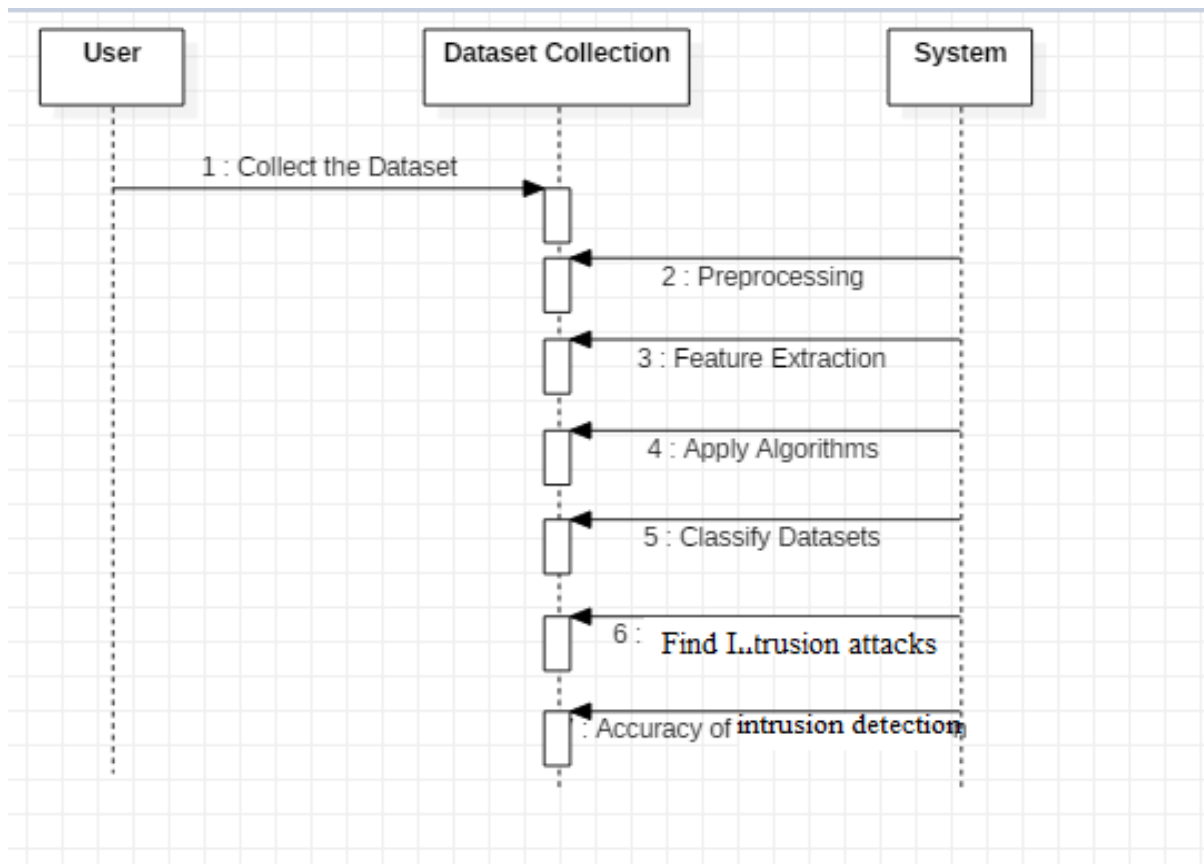


Figure :8 Sequence Diagram

ACTIVITY DIAGRAM:

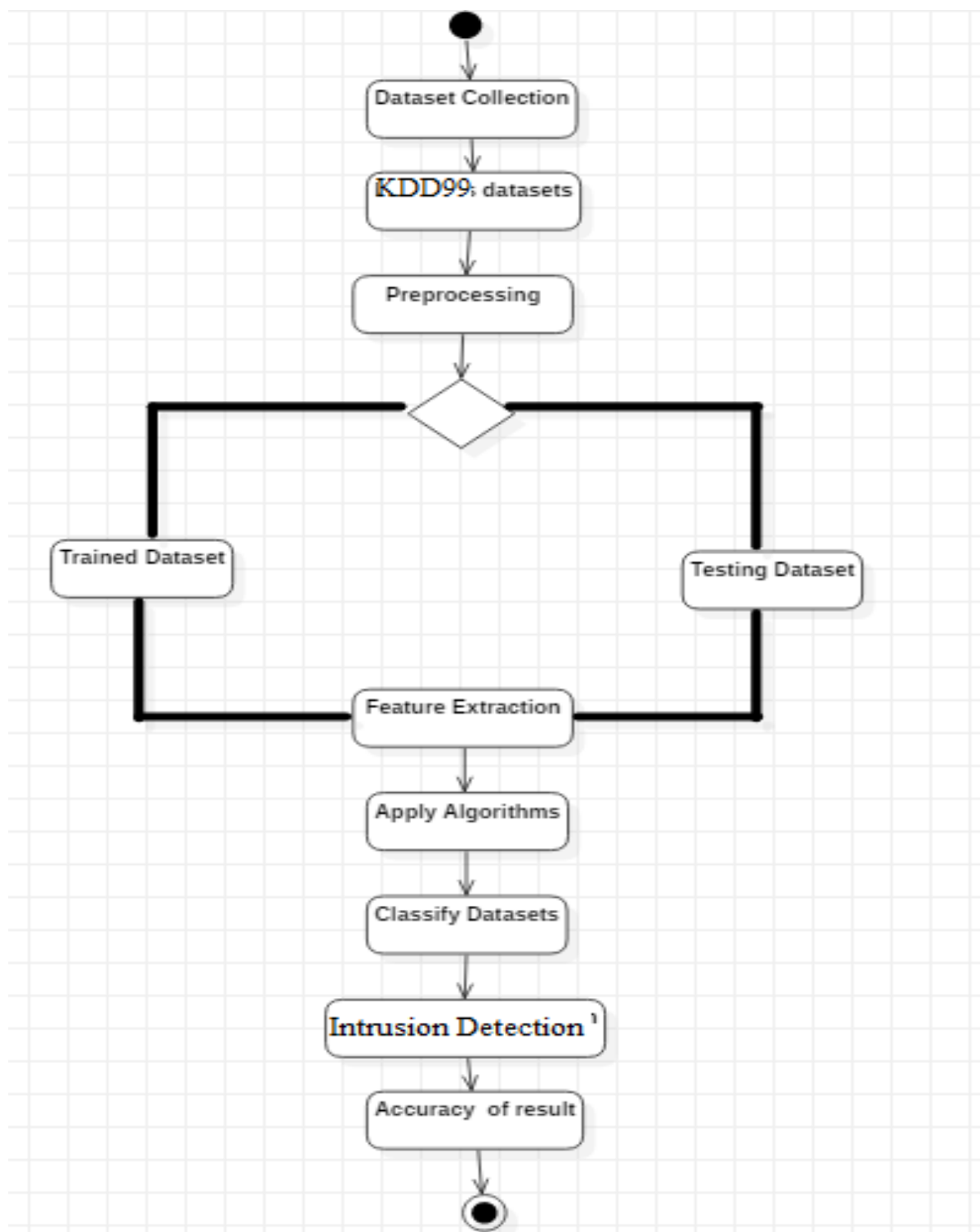


Figure :9 Acctivity Diagram

ALGORITHM

RANDOM FOREST

Random forest is a type of supervised machine learning algorithm based on ensemble learning. Ensemble learning is a type of learning where you join different types of algorithms or same algorithm multiple times to form a more powerful prediction model. The random forest algorithm combines multiple algorithm of the same type i.e. multiple decision *trees*, resulting in a *forest of trees*, hence the name "Random Forest". The random forest algorithm can be used for both regression and classification tasks.

HOW RANDOM FOREST WORKS

The following are the basic steps involved in performing the random forest algorithm

1. Pick N random records from the dataset.
2. Build a decision tree based on these N records.
3. Choose the number of trees you want in your algorithm and repeat steps 1 and 2.
4. For classification problem, each tree in the forest predicts the category to which the new record belongs. Finally, the new record is assigned to the category that wins the majority vote.

ADVANTAGES OF USING RANDOM FOREST

Pros of using random forest for classification and regression.

1. The random forest algorithm is not biased, since, there are multiple trees and each tree is trained on a subset of data. Basically, the random forest algorithm relies on the power of "the crowd"; therefore, the overall biasedness of the algorithm is reduced.

2. This algorithm is very stable. Even if a new data point is introduced in the dataset the overall algorithm is not affected much since new data may impact one tree, but it is very hard for it to impact all the trees.
3. The random forest algorithm works well when you have both categorical and numerical features.

Three common data pre-processing steps are:

- **Formatting:** The data you have selected may not be in a format that is suitable for you to work with. The data may be in a relational database and you would like it in a flat file, or the data may be in a proprietary file format and you would like it in a relational database or a text file.
- **Cleaning:** Cleaning data is the removal or fixing of missing data. There may be data instances that are incomplete and do not carry the data you believe you need to address the problem. These instances may need to be removed. Additionally, there may be sensitive information in some of the attributes and these attributes may need to be anonymized or removed from the data entirely.
- **Sampling:** There may be far more selected data available than you need to work with. More data can result in much longer running times for algorithms and larger computational and memory requirements. You can take a smaller representative sample of the selected data that may be much faster for exploring and prototyping solutions before considering the whole dataset.

Domain Specification

MACHINE LEARNING

Machine Learning is a system that can learn from example through self-improvement and without being explicitly coded by programmer. The breakthrough comes with the idea that a machine can singularly learn from the data (i.e., example) to produce accurate results.

Machine learning combines data with statistical tools to predict an output. This output is then used by corporate to makes actionable insights.

Machine learning is closely related to data mining and Bayesian predictive modeling. The machine receives data as input, use an algorithm to formulate answers.

A typical machine learning tasks are to provide a recommendation. For those who have a Netflix account, all recommendations of movies or series are based on the user's historical data. Tech companies are using unsupervised learning to improve the user experience with personalizing recommendation.

Machine learning is also used for a variety of task like fraud detection, predictive maintenance, portfolio optimization, automatize task and so on.

Machine Learning vs. Traditional Programming

Traditional programming differs significantly from machine learning. In traditional programming, a programmer code all the rules in consultation with an expert in the industry for which software is being developed. Each rule is based on a logical foundation; the machine will execute an output following the logical statement. When the system grows complex, more rules need to be written. It can quickly become unsustainable to maintain.

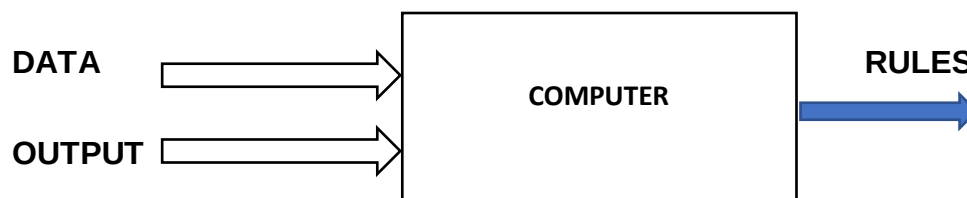


Figure 10:Machine Learning vs. Traditional Programming

Machine Learning

How does Machine learning work?

Machine learning is the brain where all the learning takes place. The way the machine learns is similar to the human being. Humans learn from experience. The more we know, the more easily we can predict.

By analogy, when we face an unknown situation, the likelihood of success is lower than the known situation. Machines are trained the same. To make an accurate prediction, the machine sees an example. When we give the machine a similar example, it can figure out the outcome. However, like a human, if its feed a previously unseen example, the machine has difficulties to predict.

The core objective of machine learning is the **learning** and **inference**. First of all, the machine learns through the discovery of patterns. This discovery is made thanks to the **data**. One crucial part of the data scientist is to choose carefully which data to provide to the machine. The list of attributes used to solve a problem is called a **feature vector**. You can think of a feature vector as a subset of data that is used to tackle a problem.

The machine uses some fancy algorithms to simplify the reality and transform this discovery into a **model**. Therefore, the learning stage is used to describe the data and summarize it into a model.

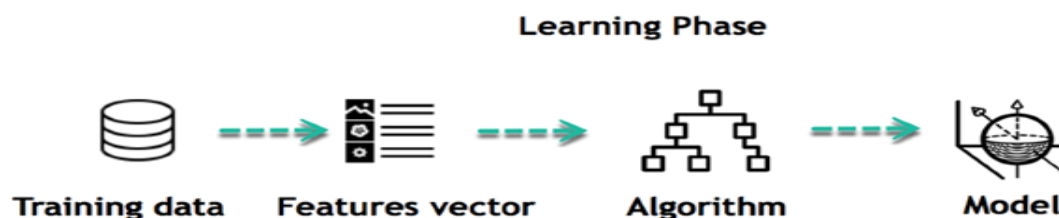


Figure 11: Learning Phase

For instance, the machine is trying to understand the relationship between the wage of an individual and the likelihood to go to a fancy restaurant. It turns out the machine

finds a positive relationship between wage and going to a high-end restaurant: This is the model

Inferring

When the model is built, it is possible to test how powerful it is on never-seen-before data. The new data are transformed into a features vector, go through the model and give a prediction.

This is all the beautiful part of machine learning. There is no need to update the rules or train again the model. You can use the model previously trained to make inference on new data.

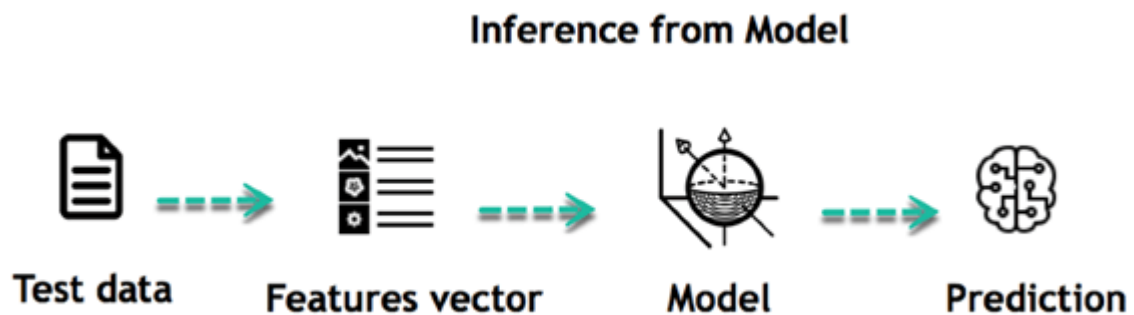


Figure 12: Inference from Model

The life of Machine Learning programs is straightforward and can be summarized in the following points:

1. Define a question
2. Collect data
3. Visualize data
4. Train algorithm
5. Test the Algorithm
6. Collect feedback
7. Refine the algorithm
8. Loop 4-7 until the results are satisfying
9. Use the model to make a prediction

Once the algorithm gets good at drawing the right conclusions, it applies that knowledge to new sets of data. Machine learning Algorithms and where they are used?

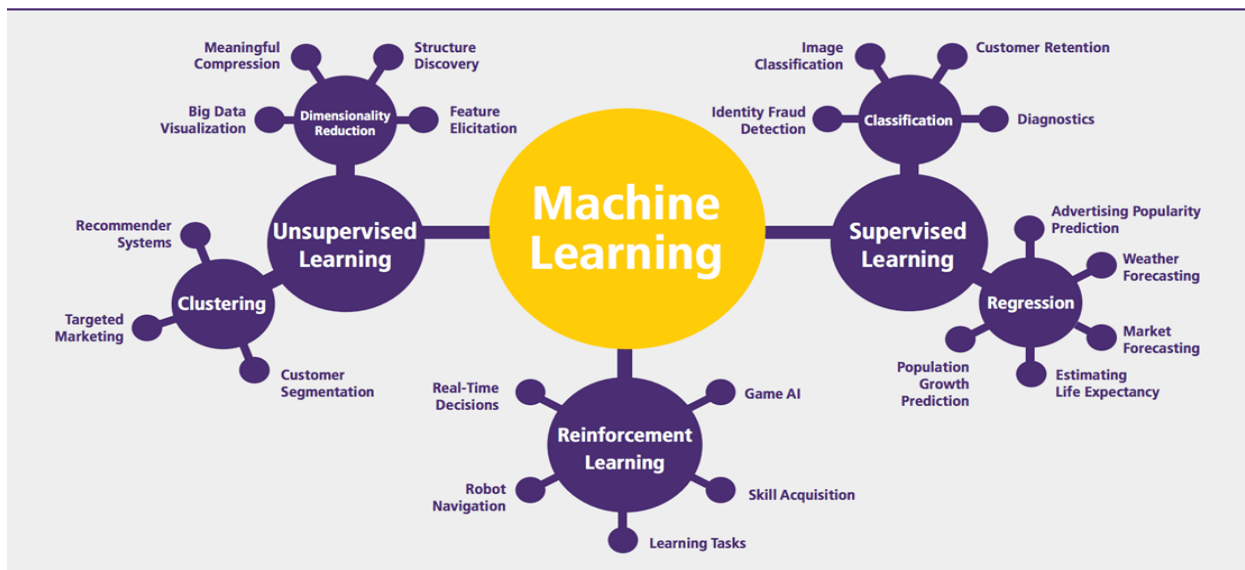


Figure13: Machine Learning Algorithms

Machine learning can be grouped into two broad learning tasks: Supervised and Unsupervised. There are many other algorithms

Supervised learning

An algorithm uses training data and feedback from humans to learn the relationship of given inputs to a given output. For instance, a practitioner can use marketing expense and weather forecast as input data to predict the sales of cans.

You can use supervised learning when the output data is known. The algorithm will predict new data.

There are two categories of supervised learning:

- Classification task
- Regression task

Classification

Imagine you want to predict the gender of a customer for a commercial. You will start gathering data on the height, weight, job, salary, purchasing basket, etc. from your customer database. You know the gender of each of your customer, it can only be male or female. The objective of the classifier will be to assign a probability of being a male or a female (i.e., the label) based on the information (i.e., features you have collected). When the model learned how to recognize male or female, you can use new data to make a prediction. For instance, you just got new information from an unknown customer, and you want to know if it is a male or female. If the classifier predicts male = 70%, it means the algorithm is sure at 70% that this customer is a male, and 30% it is a female.

The label can be of two or more classes. The above example has only two classes, but if a classifier needs to predict object, it has dozens of classes (e.g., glass, table, shoes, etc. each object represents a class)

Regression

When the output is a continuous value, the task is a regression. For instance, a financial analyst may need to forecast the value of a stock based on a range of feature like equity, previous stock performances, macroeconomics index. The system will be trained to estimate the price of the stocks with the lowest possible error.

Algorithm	Description	Type
Linear regression	Finds a way to correlate each feature to the output to help predict future values.	Regression
Logistic regression	Extension of linear regression that's used for classification tasks. The output variable is binary (e.g., only black or white) rather than continuous (e.g., an infinite list of potential colors)	Classification
Decision tree	Highly interpretable classification or regression model that splits data-feature values into branches at decision nodes (e.g., if a	Regression Classification

feature is a color, each possible color becomes a new branch) until a final decision output is made

Naive Bayes	The Bayesian method is a classification method that makes use of the Bayesian theorem. The theorem updates the prior knowledge of an event with the independent probability of each feature that can affect the event.	Regression Classification
Support vector machine	Support Vector Machine, or SVM, is typically used for the classification task. SVM algorithm finds a hyperplane that optimally divided the classes. It is best used with a non-linear solver.	Regression (not very common) Classification
Random forest	The algorithm is built upon a decision tree to improve the accuracy drastically. Random forest generates many times simple decision trees and uses the 'majority vote' method to decide on which label to return. For the classification task, the final prediction will be the one with the most votes; while for the regression task, the average prediction of all the trees is the final prediction.	Regression Classification
AdaBoost	Classification or regression technique that uses a multitude of models to come up with a decision but weighs them based on their accuracy in predicting the outcome	Regression Classification
Gradient-boosting trees	Gradient-boosting trees is a state-of-the-art classification/regression technique. It is focusing on the error committed by the previous trees and tries to correct it.	Regression Classification

Table 2 : Classification

Unsupervised learning

In unsupervised learning, an algorithm explores input data without being given an explicit output variable (e.g., explores customer demographic data to identify patterns) You can use it when you do not know how to classify the data, and you want the algorithm to find patterns and classify the data for you

Algorithm	Description	Type
K-means clustering	Puts data into some groups (k) that each contains data with similar characteristics (as determined by the model, not in advance by humans)	Clustering
Gaussian mixture model	A generalization of k-means clustering that provides more flexibility in the size and shape of groups (clusters	Clustering
Hierarchical clustering	Splits clusters along a hierarchical tree to form a classification system. Can be used for Cluster loyalty-card customer	Clustering
Recommender system	Help to define the relevant data for making a recommendation.	Clustering
PCA/T-SNE	the dimensionality of the data. The algorithms reduce the number of features to 3 or 4 vectors with the highest variances.	Dimension Reduction

Table 3: Unsupervised learning

Application of Machine learning

Augmentation:

- Machine learning, which assists humans with their day-to-day tasks, personally or commercially without having complete control of the output. Such machine learning is used in different ways such as Virtual Assistant, Data analysis, software solutions. The primary user is to reduce errors due to human bias.

Automation:

- Machine learning, which works entirely autonomously in any field without the need for any human intervention. For example, robots performing the essential process steps in manufacturing plants.

Finance Industry

- Machine learning is growing in popularity in the finance industry. Banks are mainly using ML to find patterns inside the data but also to prevent fraud.

Government organization

- The government makes use of ML to manage public safety and utilities. Take the example of China with the massive face recognition. The government uses Artificial intelligence to prevent jaywalker.

Healthcare industry

- Healthcare was one of the first industry to use machine learning with image detection.

Marketing

- Broad use of AI is done in marketing thanks to abundant access to data. Before the age of mass data, researchers develop advanced mathematical tools like Bayesian analysis to estimate the value of a customer. With the boom of data, marketing department relies on AI to optimize the customer relationship and marketing campaign.

Example of application of Machine Learning in Supply Chain

Machine learning gives terrific results for visual pattern recognition, opening up many potential applications in physical inspection and maintenance across the entire supply chain network.

Unsupervised learning can quickly search for comparable patterns in the diverse dataset. In turn, the machine can perform quality inspection throughout the logistics hub, shipment with damage and wear.

For instance, IBM's Watson platform can determine shipping container damage. Watson combines visual and systems-based data to track report and make recommendations in real-time.

In past year stock manager relies extensively on the primary method to evaluate and forecast the inventory. When combining big data and machine learning, better forecasting techniques have been implemented (an improvement of 20 to 30 % over traditional forecasting tools). In term of sales, it means an increase of 2 to 3 % due to the potential reduction in inventory costs.

Example of Machine Learning Google Car

For example, everybody knows the Google car. The car is full of lasers on the roof which are telling it where it is regarding the surrounding area. It has radar in the front, which is informing the car of the speed and motion of all the cars around it. It uses all of that data to figure out not only how to drive the car but also to figure out and predict what potential drivers around the car are going to do. What's impressive is that the car is processing almost a gigabyte a second of data.

Deep Learning

Deep learning is computer software that mimics the network of neurons in a brain. It is a subset of machine learning and is called deep learning because it makes use of deep neural networks. The machine uses different layers to learn from the data. The depth of the model is represented by the number of layers in the model. Deep learning is the new state of the art in term of AI. In deep learning, the learning phase is done through a neural network.

Reinforcement Learning

Reinforcement learning is a subfield of machine learning in which systems are trained by receiving virtual "rewards" or "punishments," essentially learning by trial and error. Google's DeepMind has used reinforcement learning to beat a human champion in the Go games. Reinforcement learning is also used in video games to improve the gaming experience by providing smarter bot.

One of the most famous algorithms are:

- Q-learning
- Deep Q network
- State-Action-Reward-State-Action (SARSA)
- Deep Deterministic Policy Gradient (DDPG)

Deep Learning

Deep learning is a computer software that mimics the network of neurons in a brain. It is a subset of machine learning and is called deep learning because it makes use of deep neural networks. The machine uses different layers to learn from the data. The depth of the model is represented by the number of layers in the model. Deep learning is the new state of the art in term of AI. In deep learning, the learning phase is done through a neural network.

Reinforcement Learning

Reinforcement learning is a subfield of machine learning in which systems are trained by receiving virtual "rewards" or "punishments," essentially learning by trial and error. Google's DeepMind has used reinforcement learning to beat a human champion in the Go games. Reinforcement learning is also used in video games to improve the gaming experience by providing smarter bot.

One of the most famous algorithms are:

- Q-learning
- Deep Q network
- State-Action-Reward-State-Action (SARSA)
- Deep Deterministic Policy Gradient (DDPG)

Applications/ Examples of deep learning applications

AI in Finance: The financial technology sector has already started using AI to save time, reduce costs, and add value. Deep learning is changing the lending industry by using more robust credit scoring. Credit decision-makers can use AI for robust credit lending applications to achieve faster, more accurate risk assessment, using machine intelligence to factor in the character and capacity of applicants.

Underwrite is a Fintech company providing an AI solution for credit makers company. underwrite.ai uses AI to detect which applicant is more likely to pay back a loan. Their approach radically outperforms traditional methods.

AI in HR: Under Armour, a sportswear company revolutionizes hiring and modernizes the candidate experience with the help of AI. In fact, Under Armour Reduces hiring time for its retail stores by 35%. Under Armour faced a growing popularity interest back in 2012. They had, on average, 30000 resumes a month. Reading all of those applications and begin to start the screening and interview process was taking too long. The lengthy process to get people hired and on-boarded impacted Under Armour's ability to have their retail stores fully staffed, ramped and ready to operate.

At that time, Under Armour had all of the 'must have' HR technology in place such as transactional solutions for sourcing, applying, tracking and onboarding but those tools weren't useful enough. Under armour choose **HireVue**, an AI provider for HR solution, for both on-demand and live interviews. The results were bluffing; they managed to decrease by 35% the time to fill. In return, the hired higher quality staffs.

AI in Marketing: AI is a valuable tool for customer service management and personalization challenges. Improved speech recognition in call-center management and call routing as a result of the application of AI techniques allows a more seamless experience for customers.

For example, deep-learning analysis of audio allows systems to assess a customer's emotional tone. If the customer is responding poorly to the AI chatbot, the system can be rerouted the conversation to real, human operators that take over the issue.

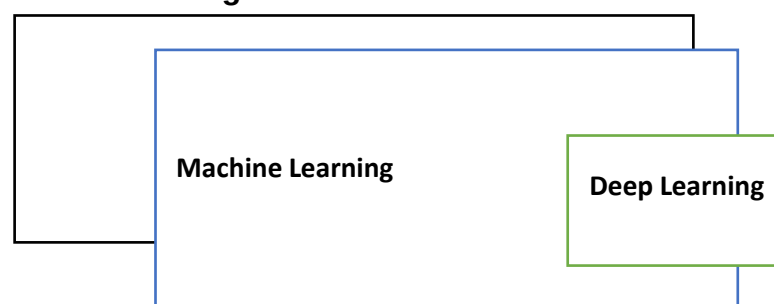
Difference between Machine Learning and Deep Learning

	Machine Learning	Deep Learning
Data Dependencies	Excellent performances on a small/medium dataset	Excellent performance on a big dataset
Hardware dependencies	Work on a low-end machine.	Requires powerful machine, preferably with GPU: DL performs a significant amount of matrix multiplication
Feature engineering	Need to understand the features that represent the data	No need to understand the best feature that represents the data
Execution time	From few minutes to hours	Up to weeks. Neural Network needs to compute a significant number of weights
Interpretability	Some algorithms are easy to interpret (logistic, decision tree), some are almost impossible (SVM, XGBoost)	Difficult to impossible

Table 4: Difference between Machine Learning and Deep Learning

Apart from the three examples above, AI is widely used in other sectors/industries.

Artificial Intelligence



When to use ML or DL?

In the table below, we summarize the difference between machine learning and deep learning.

	Machine learning	Deep learning
Training dataset	Small	Large
Choose features	Yes	No
Number of algorithms	Many	Few
Training time	Short	Long

Table 5: ML vs DL

With machine learning, you need fewer data to train the algorithm than deep learning. Deep learning requires an extensive and diverse set of data to identify the underlying structure. Besides, machine learning provides a faster-trained model. Most advanced deep learning architecture can take days to a week to train. The advantage of deep learning over machine learning is it is highly accurate. You do not need to understand what features are the best representation of the data; the neural network learned how to select critical features. In machine learning, you need to choose for yourself what features to include in the model.

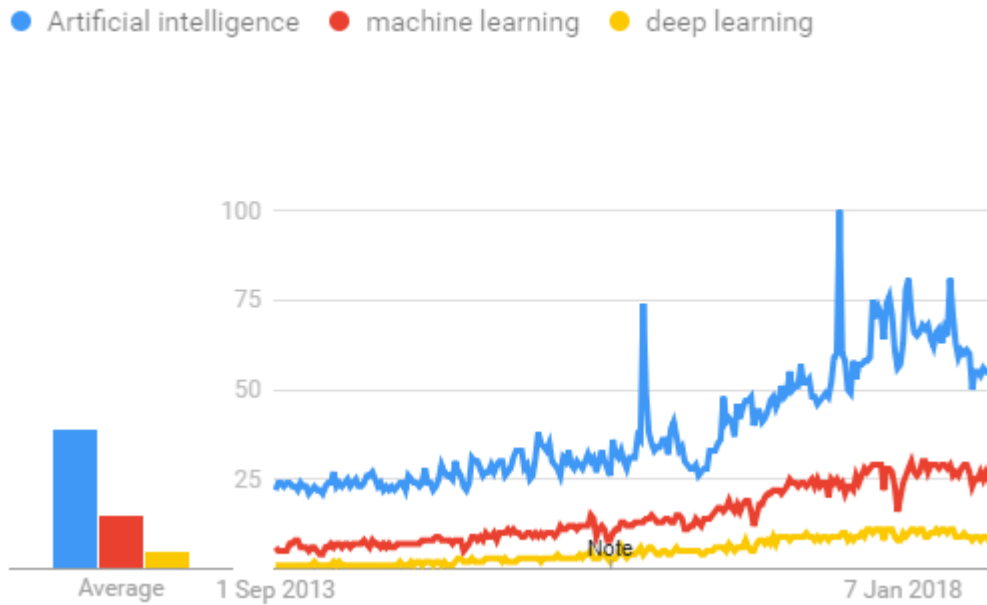


Figure 14: Graph of AL,ML,DL

TensorFlow

The most famous deep learning library in the world is Google's TensorFlow. Google product uses machine learning in all of its products to improve the search engine, translation, image captioning or recommendations.

To give a concrete example, Google users can experience a faster and more refined the search with AI. If the user types a keyword a the search bar, Google provides a recommendation about what could be the next word.

Google wants to use machine learning to take advantage of their massive datasets to give users the best experience. Three different groups use machine learning:

- Researchers
- Data scientists
- Programmers.

They can all use the same toolset to collaborate with each other and improve their efficiency.

Google does not just have any data; they have the world's most massive computer, so TensorFlow was built to scale.

TensorFlow is a library developed by the Google Brain Team to accelerate machine learning and deep neural network research.

It was built to run on multiple CPUs or GPUs and even mobile operating systems, and it has several wrappers in several languages like Python, C++ or Java.

In this tutorial, you will learn

TensorFlow Architecture

Tensorflow architecture works in three parts:

- Preprocessing the data
- Build the model
- Train and estimate the model

It is called Tensorflow because it takes input as a multi-dimensional array, also known as **tensors**. You can construct a sort of **flowchart** of operations (called a Graph) that you want to perform on that input. The input goes in at one end, and then it flows through this system of multiple operations and comes out the other end as output.

This is why it is called TensorFlow because the tensor goes in it flows through a list of operations, and then it comes out the other side.

Where can Tensorflow run?

TensorFlow can hardware, and software requirements can be classified into

Development Phase: This is when you train the mode. Training is usually done on your Desktop or laptop.

Run Phase or Inference Phase: Once training is done Tensorflow can be run on many different platforms. You can run it on

- Desktop running Windows, macOS or Linux

- Cloud as a web service
- Mobile devices like iOS and Android

You can train it on multiple machines then you can run it on a different machine, once you have the trained model.

The model can be trained and used on GPUs as well as CPUs. GPUs were initially designed for video games. In late 2010, Stanford researchers found that GPU was also very good at matrix operations and algebra so that it makes them very fast for doing these kinds of calculations. Deep learning relies on a lot of matrix multiplication. TensorFlow is very fast at computing the matrix multiplication because it is written in C++. Although it is implemented in C++, TensorFlow can be accessed and controlled by other languages mainly, Python.

Finally, a significant feature of TensorFlow is the TensorBoard. The TensorBoard enables to monitor graphically and visually what TensorFlow is doing.

List of Prominent Algorithms supported by TensorFlow

- Linear regression: `tf.estimator.LinearRegressor`
- Classification: `tf.estimator.LinearClassifier`
- Deep learning classification: `tf.estimator.DNNClassifier`
- Deep learning wide and deep: `tf.estimator.DNNLinearCombinedClassifier`
- Booster tree regression: `tf.estimator.BoostedTreesRegressor`
- Boosted tree classification: `tf.estimator.BoostedTreesClassifier`

REQUIREMENTS ANALYSIS

SOFTWARE REQUIREMENTS

- Python
- Anaconda Navigator
- Python built-in modules

- Numpy
- Pandas
- Matplotlib
- Sklearn
- Seaborn

ANACONDA NAVIGATOR

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows you to launch applications and easily manage conda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository. It is available for Windows, mac OS and Linux.

Why use Navigator?

In order to run, many scientific packages depend on specific versions of other packages. Data scientists often use multiple versions of many packages, and use multiple environments to separate these different versions.

The command line program conda is both a package manager and an environment manager, to help data scientists ensure that each version of each package has all the dependencies it requires and works correctly.

Navigator is an easy, point-and-click way to work with packages and environments without needing to type conda commands in a terminal window. You can use it to find the packages you want, install them in an environment, run the packages and update them, all inside Navigator.

WHAT APPLICATIONS CAN I ACCESS USING NAVIGATOR?

The following applications are available by default in Navigator:

- JupyterLab
- Jupyter Notebook
- QtConsole
- Spyder

- VSCode
- Glueviz
- Orange 3 App
- Rodeo
- RStudio

Advanced conda users can also build your own Navigator applications

How can I run code with Navigator?

The simplest way is with Spyder. From the Navigator Home tab, click Spyder, and write and execute your code.

You can also use Jupyter Notebooks the same way. Jupyter Notebooks are an increasingly popular system that combine your code, descriptive text, output, images and interactive interfaces into a single notebook file that is edited, viewed and used in a web browser.

What's new in 1.9?

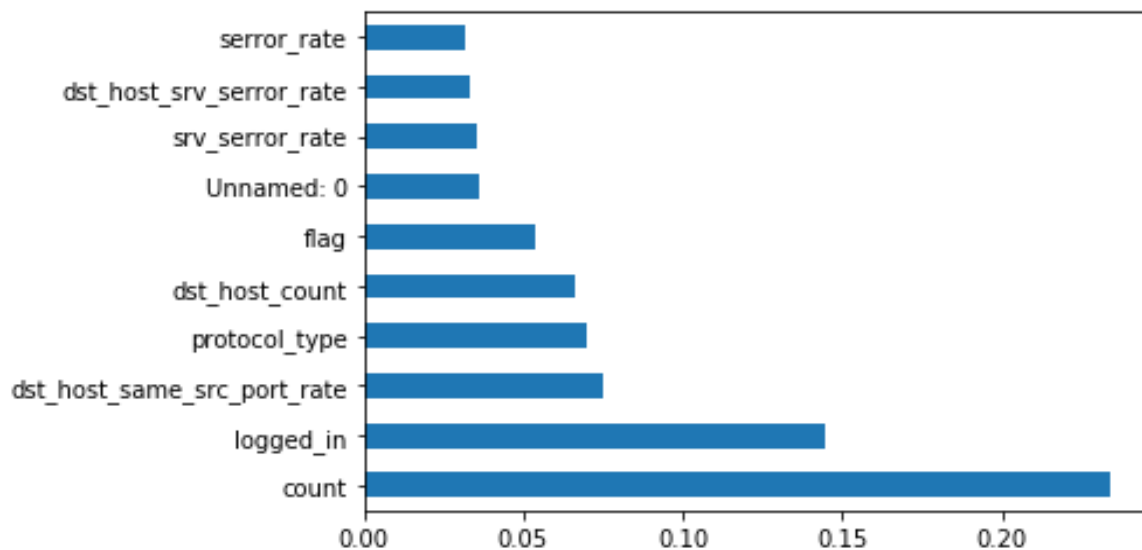
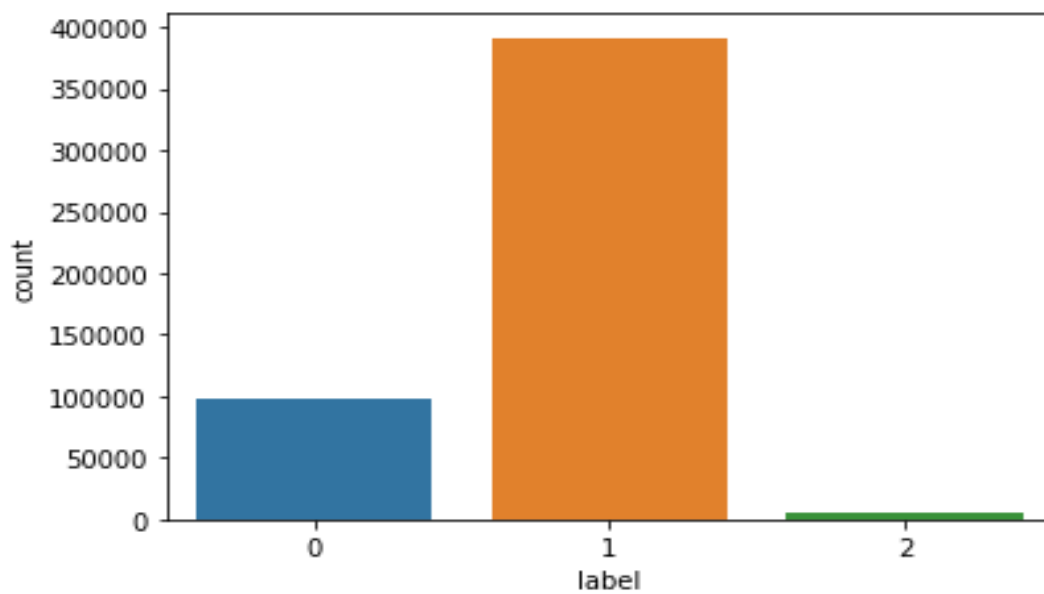
- Add support for **Offline Mode** for all environment related actions.
- Add support for custom configuration of main windows links.
- Numerous bug fixes and performance enhancement

CHAPTER 5

RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS

OUTPUT RESULTS:

```
<matplotlib.axes._subplots.AxesSubplot at 0x2c4ade44d08>
```

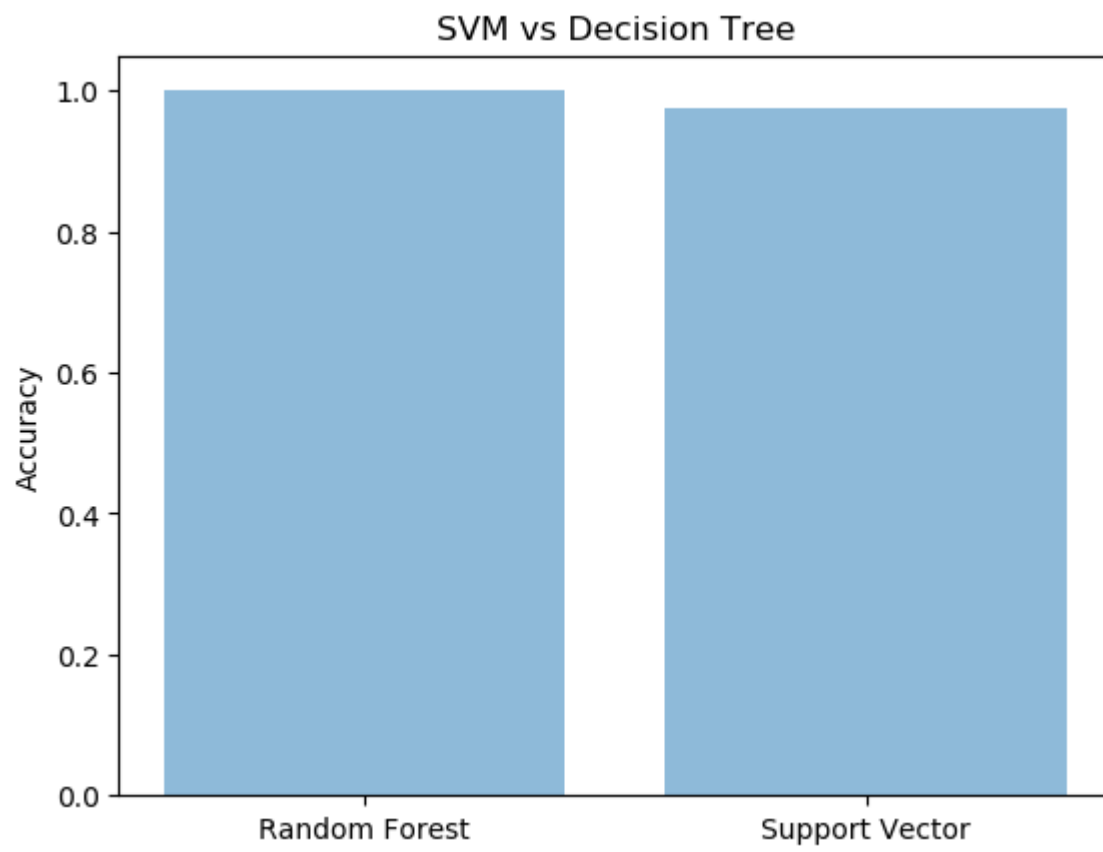


RANDOM FOREST RESULT:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	32059
1	1.00	1.00	1.00	129255
2	0.98	1.00	0.99	1713
accuracy			1.00	163027
macro avg	0.99	1.00	1.00	163027
weighted avg	1.00	1.00	1.00	163027

SVM RESULT:

	precision	recall	f1-score	support
0	0.95	0.93	0.94	32721
1	0.99	0.99	0.99	128855
2	0.07	0.08	0.07	1451
accuracy			0.97	163027
macro avg	0.67	0.67	0.67	163027
weighted avg	0.98	0.97	0.97	163027



CHAPTER 6

SUMMARY AND CONCLUSIONS

In this I have used the KDD99 dataset to train and test our proposed model. We used SVM and Random Forest algorithm to detect the DDoS attack. The classification module is deployed on the SDN environment. The Random Forest and SVM techniques are used to distinguish between normal and malicious traffic data. Our experimental shows that, Random Forest works better that SVM technique in this simulated environment

REFERENCES

- [1]. Gustavode Carvalho Betroli (2021). Network Intrusion Detection System With the Degree of An End-to-End Framework for Machine Learning. IEEE Access 10.1109/ACCESS.2021.3101188
- [2]. Dong, S., & Sarem, M. (2020). DDoS Attack Detection Method Based on Improved KNN With the Degree of DDoS Attack in Software-Defined Networks. IEEE Access, 8, 5039-5048.
- [3]. Gu, Y., Li, K., Guo, Z., & Wang, Y. (2019). Semisupervised K-means DDoS detection method using hybrid feature selection algorithm. IEEE Access, 7, 64351-64365.
- [4]. Meti, N., Narayan, D. G., & Baligar, V. P. (2017, September). Detection of distributed denial of service attacks using machine learning algorithms in software defined networks. In 2017 international conference on advances in computing, communications and informatics (ICACCI) (pp. 1366-1371). IEEE
- [5]. 15th International Symposium on Pervasive Systems, Algorithms and Networks IEEE DDoS Attack Identification and Defense using SDN based on Machine Learning Method, 2018
- [6]. Muthamil Sudar, K., & Deepalakshmi, P. (2020). A two level security mechanism to detect a DDoS flooding attack in software-defined networks using entropy-based and C4. 5 technique. Journal of High Speed Networks, (Preprint), 1- 22.
- [7]. Deepa, V., Sudar, K. M., & Deepalakshmi, P. (2018, December). Detection of DDoS attack on SDN control plane using Hybrid Machine Learning Techniques. In 2018 International Conference on Smart Systems and Inventive Technology (ICSSIT) (pp. 299-303). IEEE.
- [8]. Deepa, V., K. Muthamil Sudar, and P. Deepalakshmi. "Design of Ensemble Learning Methods for DDoS Detection in SDN Environment." 2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN). IEEE, 2019

[9]. J. Cui, M. Wang, and Y. Luo, "DDoS detection and defense mechanism based on cognitive-inspired computing in SDN," *Future Gener. Comput. Syst.*, vol. 97, pp. 275_283, Aug. 2019.

[10]. N. I. G. Dharma, M. F. Muthohar, J. D. A. Prayuda, K. Priagung, and D. Choi, "Time-based DDoS detection and mitigation for SDN controller," in *Proc. 17th Asia_Pacific Netw. Oper. Manage. Symp. (APNOMS)*, Aug. 2015, pp. 550_553.

APPENDIX OF CODE:

```
import warnings

warnings.filterwarnings('ignore')

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

df=pd.read_csv('kddcup99_csv.csv')

df.head()

df.info()

df.shape

(494020, 42)

df.isna().sum()

print('Data set:')

for col_name in df.columns:

    if df[col_name].dtypes == 'object' :

        unique_cat = len(df[col_name].unique())

        print("Feature '{col_name}' has {unique_cat}

categories".format(col_name=col_name, unique_cat=unique_cat))

print()

print('Distribution of categories in service:')
```

```

print(df['service'].value_counts().sort_values(ascending=False).head())

lab=preprocessing.LabelEncoder()

df['protocol_type']=lab.fit_transform(df['protocol_type'])

df['service']=lab.fit_transform(df['service'])

df['flag']=lab.fit_transform(df['flag'])

df.head()

df.info()

df1=df['label']

print('Label distribution Training set:')

print(df['label'].value_counts())

newdf=df1.replace({'normal':0,'smurf':1,'neptune':1,'back':1,'satan':2,'ipsweep':2,'port
sweep':2,'warezclient': 2,'teardrop': 1,

                    'pod': 1,'nmap' : 2,'guess_passwd': 2,'buffer_overflow': 2,'land':
1,'warezmaster': 2,'imap': 2,'rootkit': 2,

                    'loadmodule': 2,'ftp_write': 2,'multihop': 2,'phf': 2,'perl': 2,'spy': 2})

print(newdf.head())

#newdf.to_csv('label.csv')

df['label'] = newdf

df.head()

df.info()

df.to_csv('New_Data.csv')

data = pd.read_csv("New_Data.csv")

sns.countplot(x='label',data=data)

```

FUTURE EXTRACTION

```
from sklearn.ensemble import ExtraTreesClassifier

model = ExtraTreesClassifier()

X = data.iloc[:,data.columns!='label']
y = data.iloc[:,data.columns=='label']

model.fit(X,y)

print(model.feature_importances_)

feat_importances = pd.Series(model.feature_importances_, index=X.columns)

feat_importances.nlargest(10).plot(kind='barh')

plt.show()

from sklearn.model_selection import train_test_split

from sklearn.metrics import accuracy_score

xtrain, xtest, ytrain, ytest=train_test_split(X, y, test_size=0.33)

from sklearn.ensemble import RandomForestClassifier

rf=RandomForestClassifier()

rf.fit(xtrain,ytrain)

predic=rf.predict(xtest)

model1=accuracy_score(predic,ytest)

model1

0.9997607758224097

from sklearn import metrics

from sklearn.metrics import confusion_matrix

from sklearn import svm
```

```

support = svm.LinearSVC(random_state=20)

support.fit(xtrain,ytrain)

LinearSVC(random_state=20)
y_pred = support.predict(xtest)

y_pred

array([1, 1, 1, ..., 1, 1, 1], dtype=int64)
model2=metrics.accuracy_score(ytest,y_pred)

print(model2)

cnf_matrix = confusion_matrix(ytest,y_pred)

labels = ['Good','Malware']

sns.heatmap(cnf_matrix, annot=True, cmap="YlGnBu", fmt=".3f", xticklabels=labels,
yticklabels=labels)

plt.show()

import matplotlib.pyplot as plt; plt.rcdefaults()

import numpy as np

import matplotlib.pyplot as plt

objects = ('RF','SVM')

y_pos = np.arange(len(objects))

performance = [model1,model2]

plt.bar(y_pos, performance, align='center', alpha=0.5)

plt.xticks(y_pos, objects)

plt.ylabel('Accuracy Score')

```

```
plt.title('RANDOM FOREST vs SVM')
```

```
plt.show()
```