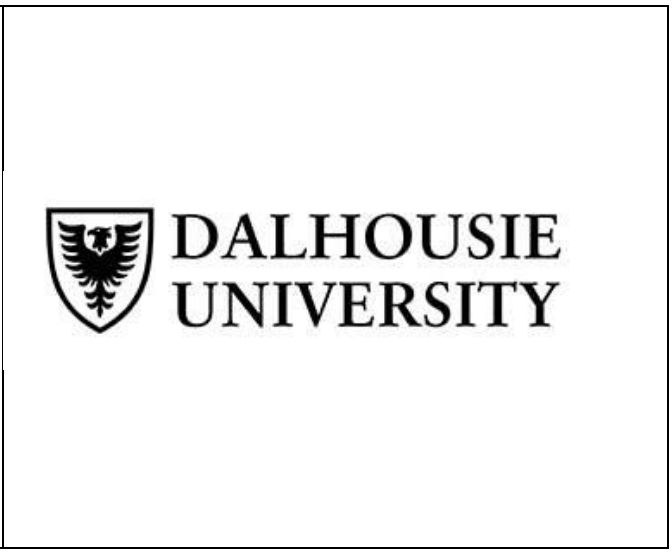


Internship report

Holistic Anomaly Detection for Enterprise Security

Trainee	Tutor
Yann KERDONCUFF Second-year student of Network and Telecommunication at Lannion IUT	**Nur ZINCIR-HEYWOOD** Computer Science professor at Dalhousie's University **Stéphanie LE MELEDER** English professor at Lannion IUT

April 2021 – July 2021

Internship report

Holistic Anomaly Detection for Enterprise Security

Trainee	Tutor
Yann KERDONCUFF Second-year student of Network and Telecommunication at Lannion IUT	**Nur ZINCIR-HEYWOOD** Computer Science professor at Dalhousie's University **Stéphanie LE MELEDER** English professor at Lannion IUT

April 2021 – July 2021

Acknowledgment

I would like to thanks Nur ZINCIR-HEYWOOD and Rafael Copstein for allowing me to work in the Network Information Management and Security Lab (NIMS). Moreover, they accompanied me very well throughout the internship

I also thank the other trainees: Nathan PORTAL, Gabin LE SAOUT, Ionel CIOBANU, David LE MANACH for their help and their views on certain technical points.

I thank the pedagogical team of Lannion IUT for their advice and the education I have received for two years.

Table of contents

Introduction....6
1 Context....7
1.1 Dalhousie University....7
1.2 NIMS lab....8
1.3 The project....8
1.4 Organization....9
1.4.1 Work environment....9
1.4.2 Planification....10
2 Logs files....11
2.1 Apache....11
2.2 Snort....11
2.3 Firewall....12
3 Log Parsing....15
3.1 SHISO....15
3.2 Free text....18
4 Machine learning....21
4.1 Supervised/Unsupervised Learning....21
4.1.1 Supervised Learning....21
4.1.2 Unsupervised Learning....23
4.2 Bag of words....24
4.3 Scikit-learn....25
4.4 Classification algorithm for Apache log....27
4.5 Result....30
5 Assessment....32
5.1 Technical skills....32
5.2 Professional skills....33
Conclusion....34
Lexicon....35
Figure Table....36
Annexes....37
Abstract....47

Introduction

Today the internet is an integral part of our lives. The world is becoming more and more connected with the arrival of new technologies such as 5G. This leads us to use different web services for connected objects, emailing, gaming, and more.

All these services generate a large number of logs. Logs are used to store a history of service events: this can be the connection of a client or an error. Logs are very interesting tools to find the origin of a failure. It is used in security to prevent an attack. We can also use them to make statistics.

However, before you can use them, they must be readable by a human. Because this is not the case, it may be that in a log file with more than a million lines it is only one line that can cause the failure. So, to find this line it's like looking for a needle in a haystack. To find this line, the logs must be structured and tools must be used to interpret them by a human.

It is at this point that my internship of validation of DUT (12 April 2021 - 2 July 2021) took place in NIMS lab with Dalhousie University. however, the framework of the internship is modified with the COVID-19 epidemic, everything is done remotely. This work has already been done by the lab. This internship has an educational purpose to make us understand the functioning of log processing.

Through this internship, I will work on skills that are not present in my training like data analysis and machine learning*.

To start, I had to study different points:

- Understand how logs are generated
- interprets different logs from different services
- structuring and extracting data from logs
- Learn to use machine learning

Before going into the content of the internship, I will introduce the background and the team in more detail.

The presence of an * means that the term is explained in the lexicon

1 Context

I will first introduce you to the university and then the research labs as well as the team and the organization.

1.1 Dalhousie University

Dalhousie University is located in Halifax, Novia Scotia. This university was founded in 1818 by George Ramsay, who was Earl of Dalhousie and Lieutenant Governor of Nova Scotia. The financing of this university was made possible by the money collected from maritime customs taxes. Today she brings more than 20,000 students from different countries. She has more than 13 faculties including Computer Science.

Figure 1: Dalhousie university, Halifax map

Source: https://commons.wikimedia.org/wiki/File:Canada_Nova_Scotia_location_map_2.svg

The university's extensive worldwide contacts enhance the quality and impact of teaching and research at local, national, and international levels. Their openness to the world enables them to cooperate and join forces to support interdisciplinary, intercultural, global learning and research that is oriented towards solving problems that transcend national boundaries.

1.2 NIMS lab

NIMS lab it's the Network Information Management and Security Group. The lab is led by Nur Zincir-Heywood and Malcolm Heywood, who is an important professor of the Faculty of Computer Science. The lab is hosting about 20 graduate and undergraduate students.

The NIMS lab is focused on autonomous systems. They study the learning capacity of the systems and also monitor their behavioral changes according to their environment. The lab aims it's to make more and more aware system to their environment. They are interested to discover new behaviors. Security network is very interesting for them because there are many different behaviors, it's a field of research very interesting.

During the project, I was supervised by Nur Zincir-Heywood and Rafael Copstein. Dr. Nur Zincir-Heywood is my tutor and professor at the Faculty of Computer Science. Rafael is a Computer Science Ph.D. student at Dalhousie University under the supervision of Dr. Nur Zincir-Heywood and also a member of the NIMS lab.

1.3 The project

In this project, I discover an introduction to the autonomous system. Like I say in the introduction it's more like training to develop skills because the work was already done by the lab. This is called HADES (Holistic Anomaly Detection for Enterprise Security) let's see this goal:

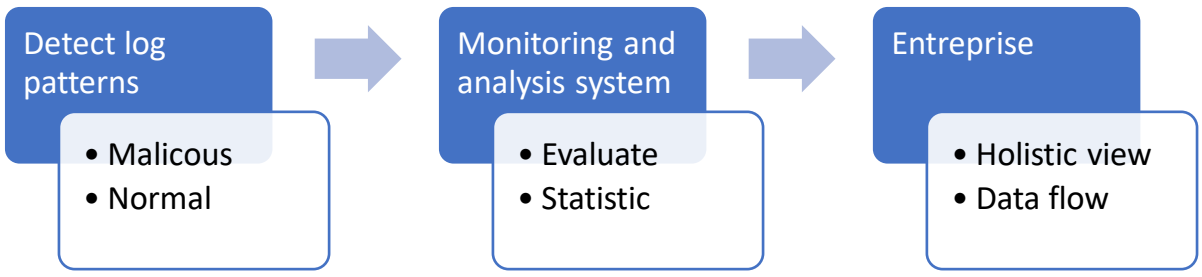


Figure 2: HADES project

This figure shows the different steps of this project. The first step is to classify in a log file each line into a malicious or normal log. Then with this information, we can make monitoring and some statistic. In the end, the enterprise can have a holistic view of this network, systems, devices, and services.

For the internship my job is to focus on the first part: Detect log patterns. I need to work with a different types of logs. My task is to make an algorithm to be able to classify logs. I work on three different services: Apache, Snort, and a Firewall.

1.4 Organization

The COVID-19 epidemic forces me to adapt to work remotely. It is very important to be well organized to work in the best condition. Besides, all the work was done in autonomy so it was essential to be well organized.

1.4.1 Work environment

It is necessary to use the tools at our service. I will present to you the different communication tools I use and my work environment.

Firstly, for the communication tools I used:

- Emailing
- Teams
- Discord

I use email to send my work and slide before a meeting, it is very useful when I have a question, it also allows me to have a written trace of the exchange and it is easy to find.

We use Teams for meetings to exchange in video conferencing. The meetings were taking place every Monday. During this time of exchange, we made a presentation of our work done during the week. We could also ask all our questions which is easier to discuss and show our problem than by mail. Moreover, it is a tool that we already know how to use it.

To work with the other trainees during the week I used Discord. It is an application for VoIP and instant messaging mainly used by gamers. Is not necessarily designed for collaborative work but it is very easy to use. We can share our screens and send screenshots.

Then for my work environment, I use a desktop with windows 10 and a Python interpreter, I decided to employ Anaconda. The Anaconda Browser is a graphical interface included in the Anaconda distribution, which allows users to launch applications, as well as manage Conda libraries, environments, and channels without using any command line.

We find applications like:

- Jupiter Notebook
- Spyder

Jupiter Notebook Spyder is a development environment for Python mainly use for data science, we can observe step by step your modification on a dataset and show decision trees of machine learning algorithms. Spyder is also a development environment for Python, this software is already used at the IUT for Python programming. In addition to that, I use to employ a Virtual Machine with Linux.

1.4.2 Planification

At the internship, I did not have to plan a schedule for the distribution of tasks. I was making progress as I went along. New tasks were giving each week. We work step by step, we were very supervised. The subject was very complicated to understand at the beginning. So, proceeding in this way in a subject that was unknown to us, allowed us to avoid getting lost.

Task	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10
Studie a log parser Algorithm										
Find log format										
Defined Free text										
Extract the free text of a log file										
Studie the machine learning										
Experiment Machine learning algorithm										

Figure 3: Gantt chart: organization of the internship

2 Logs files

Logs are used to store a history of service events. All the events are stored in a file. In this file, each line corresponds to an action of the service. The events can be very different from one service to another. However, we find frequently connections and errors. To begin, I worked on three types of logs that were provided to us: Apache, Snort, and a Firewall.

2.1 Apache

Apache is the most widely used web server software, it is open-source* and used to deploy websites on the internet. Apache is used to host a website. Most of the time in the logs we will find connections, and sometimes errors related to missing files.

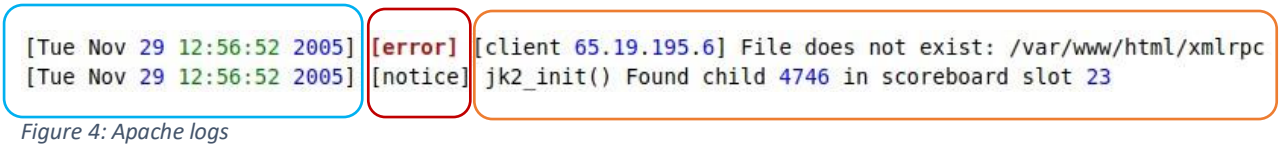


Figure 4: Apache logs

- Timestamp
- Label
- Content

This figure shows two very different logs but they have the same structure. They have in common a timestamp. In the first line, we can see it's an error then we find the content of this error. This error is formed by a request made by a client with IP address 65.19.195.6 and Apache don't find the file "xmlrpc" in the directory "/var/www/html/". In the second line is a notice that means the event is going well. In this "content", we can see the initialization of jk2(Apache module) is done.

After studying several different logs lines, I have found a structured format:

Timestamp	Label(error/notice)	Content

Figure 5 : Apache log format

2.2 Snort

Snort is an open-source network intrusion prevention system, capable of performing real-time traffic analysis and packet logging on IP networks. It can perform protocol analysis, content searching/matching, and can be used to detect a variety of attacks and probes, such as buffer overflows*, stealth port scans*, CGI* attacks, SMB* probes, OS fingerprinting* attempts, and much more.

Let’s see an example of Snort logs:

```
06/08-11:44:08.394506  [**] [1:527:8] BAD-TRAFFIC same SRC/DST [**] [Classification: Potentially
Bad Traffic] [Priority: 2] {UDP} 0.0.0.0:68 -> 255.255.255.255:67

06/08-11:44:26.556196  [**] [1:1917:6] SCAN UPnP service discover attempt [**] [Classification:
Detection of a Network Scan] [Priority: 3] {UDP} 192.168.79.10:1034 -> 239.255.255.250:1900

06/08-11:44:27.934468  [**] [1:2012811:2] ET DNS DNS Query to a .tk domain - Likely Hostile [**]
[Classification: Potentially Bad Traffic] [Priority: 2] {UDP} 192.168.79.10:1029 -> 4.2.2.3:53
```

Figure 6: Snort logs

We got three different log alerts here. We find common points concerning Apache: timestamp, content. Snort alert got a different format, we can see they have a classification, a priority, and the transport protocol with IP address source and destination.

The classification is used to categorize a rule as detecting an attack that is part of a more general type of attack class. Snort provides a default set of attack classes that are used by the default set of rules it provides. Defining classifications for rules provides a way to better organize the event data Snort produces. With the classification, a priority is also associated according to its category. A priority of 1 (high) is the most severe and 3 (low) is the least severe.

This is the example of log format for Snort logs:

Timestamp	ID	Content	Classification	Priority	Protocol	Source Address (address:port)	Destination address (address:port)

Figure 7: Snort log format

We can observe there are more categories for snort format. This because we have more detail and each line of the log is very similar. In Apache log we have “error” and “notice”, this impacts the content. We can't be more precise in Apache, but in Snort it’s easier to be precise for the format.

2.3 Firewall

A firewall is a software that prevents unauthorized access to a network. It checks incoming and outgoing traffic using a set of rules to identify and block threats.

Firewalls are used in both personal and enterprise settings, and many devices come with one built-in, including Mac, Windows, and Linux computers. it the first security element of a network. They are broadly considered an essential component of network security.

A firewall establishes a border between an external network and the network it guards. It is inserted inline across a network connection and inspects all packets entering and leaving the guarded network.

Firewall get information of IP packets:

- The source
- The destination
- Other parameters

We can find this information into IP header, this is the structure of IP header in IPv4:

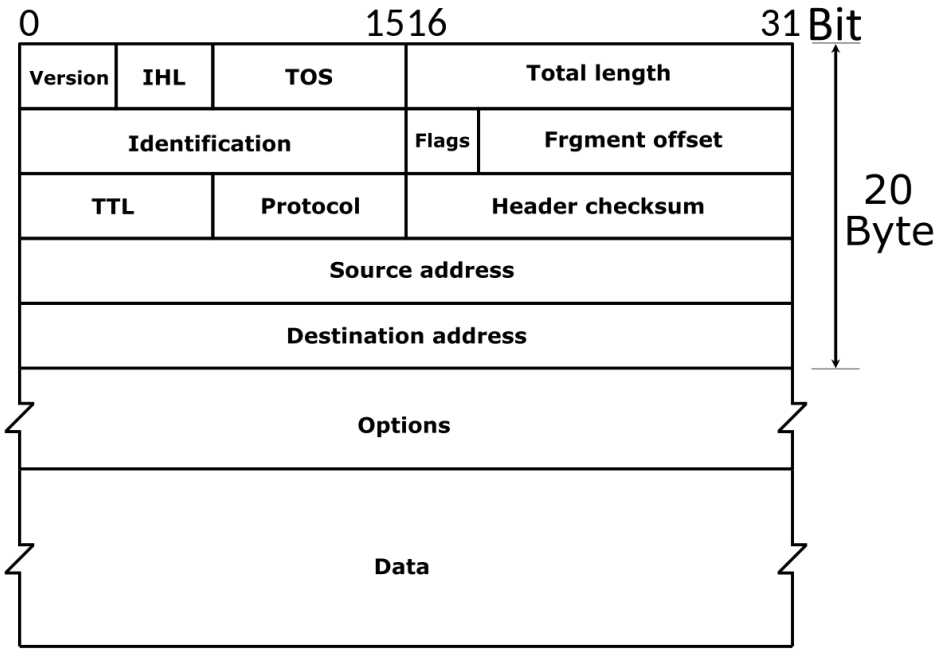


Figure 8: IP header (source https://en.wikipedia.org/wiki/IPv4)

The firewall focus on the first 20 byte of IP packet, we will see that most of the information in the header is used and stored in the log file. Now let's look at an example of log:

2021-02-23T00:12:52.857Z IN=eth0 OUT=eth1.217 SRC=185.153.197.146 DST=129.173.67.162 LEN=40 TOS=0x08 PREC=0x20 TTL=237 ID=15140 PROTO=TCP SPT=52886 DPT=5389 WINDOW=1024 RES=0x00 SYN URGP=0

2021-02-23T00:11:23.646Z IN= OUT=eth0 SRC=10.11.20.215 DST=216.197.228.230 LEN=76 TOS=0x00 PREC=0x00 TTL=63 ID=48853 DF PROTO=UDP SPT=52652 DPT=123 LEN=56

Figure 9: Firewall logs

For Firewall it's very easy to identify a structure because it is already in its shape. We only need to put it into a type of structured file.

time	IN	OUT	Source Address	Destination Address	LEN	TOS	PREC	TTL	ID	PROTO	SPT	DPT	Content

Figure 10: Firewall log format

After identifying all of the log formats for each service. I'm going to move on to the next step: the log parsing.

3 Log Parsing

Parsing is the process of splitting data into chunks of information that are easier to manipulate. Basically, a structured file easy it is going to be to analyze, query, and visualize your data for a human withing tools. Under the picture below we can observe the transformation of unstructured into structured Apache log.

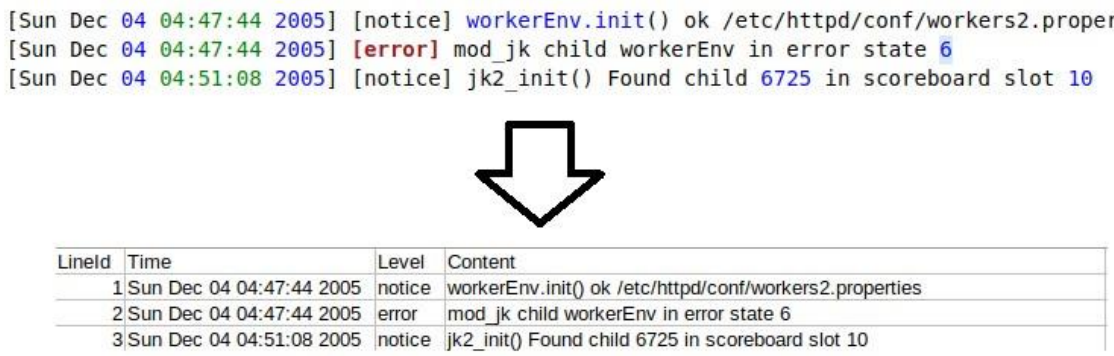


Figure 11: unstructured to structured Apache log

We can observe at the top the contents of an Apache log file. The contents are unstructured to be exploited to make an analysis. At the bottom, we get a structured version of these same log lines. This version can be useful to be exploited.

This transformation's possible with a log parser, we can see this structure respects the structure of figure 5. Previously Nur Zincir-Heywood asked us to choose to study an algorithm of log parsing. In my case, I chose the SHISO algorithm. I will explain how SHISO works.

3.1 SHISO

SHISO* is a log parser, he can structure our input log into a CSV* file. At the same time is going to make another structured file with the different template of each form of log that he may have found. To do this he proceeds in two steps. In the first step, the algorithm makes a pre-processing* to make the structured file. In the second step, he makes a file with a different template of the log with a structured tree.

Let's focus on the first step: pre-processing. Pre-processing is to clean and structure the data. We can see in figure 10 the log is structured and some characters are deleting like "[" and "]" because they don't have a utility.

To do that SHISO need the log format of the input. it is also required to adapt the suppression of the useless characters. I will show a figure to determine the final log format:

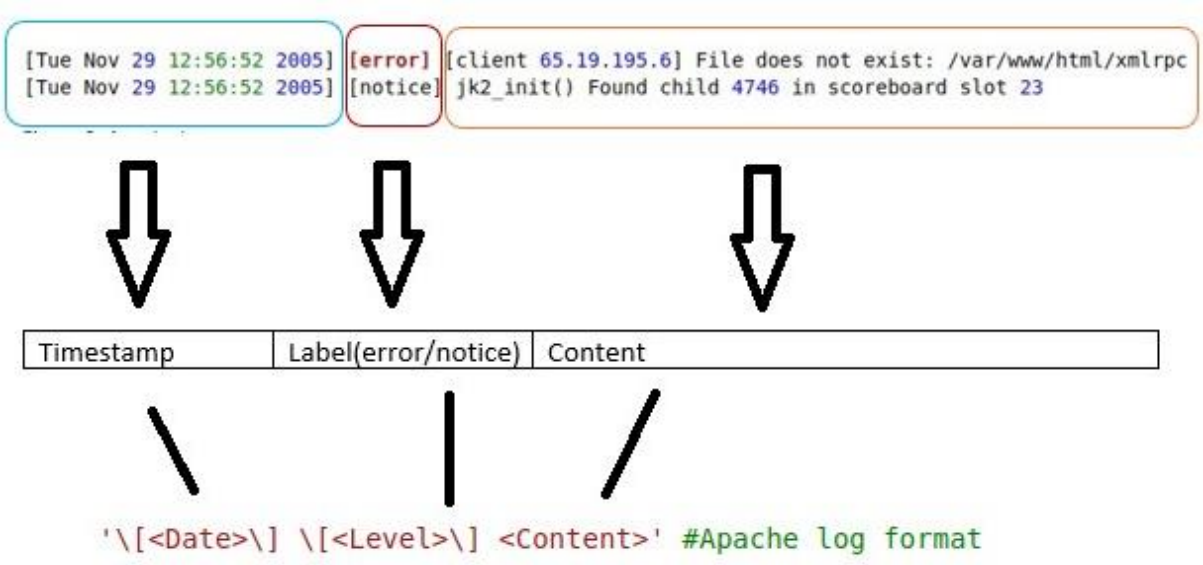


Figure 12: final Apache log format from SHISO

In this case, we compare logs with log previous log format what is useless characters. We using backslash to parse and delete these characters. In the first log we can see there are the same characters in the content but we can delete him because is not recursive for each content.

Then we got the good final log format, SHISO going to run all the log file to match with every line who have the same format. Sometimes, some lines don't match with this format. The algorithm goanna dodges them for the structured log files.

In the second step, SHISO makes a list of templates from these logs. He will make a classification of each different event type and its parameters. Firstly, SHISO splits the new log message into a word list using common delimiters (Space, "=", ";",) without separating file paths. Secondly, he is making a new node in the tree structure: that corresponding to the word list. This part can be difficult to understand, I make a figure to understand easily how does SHISO make a tree structure:

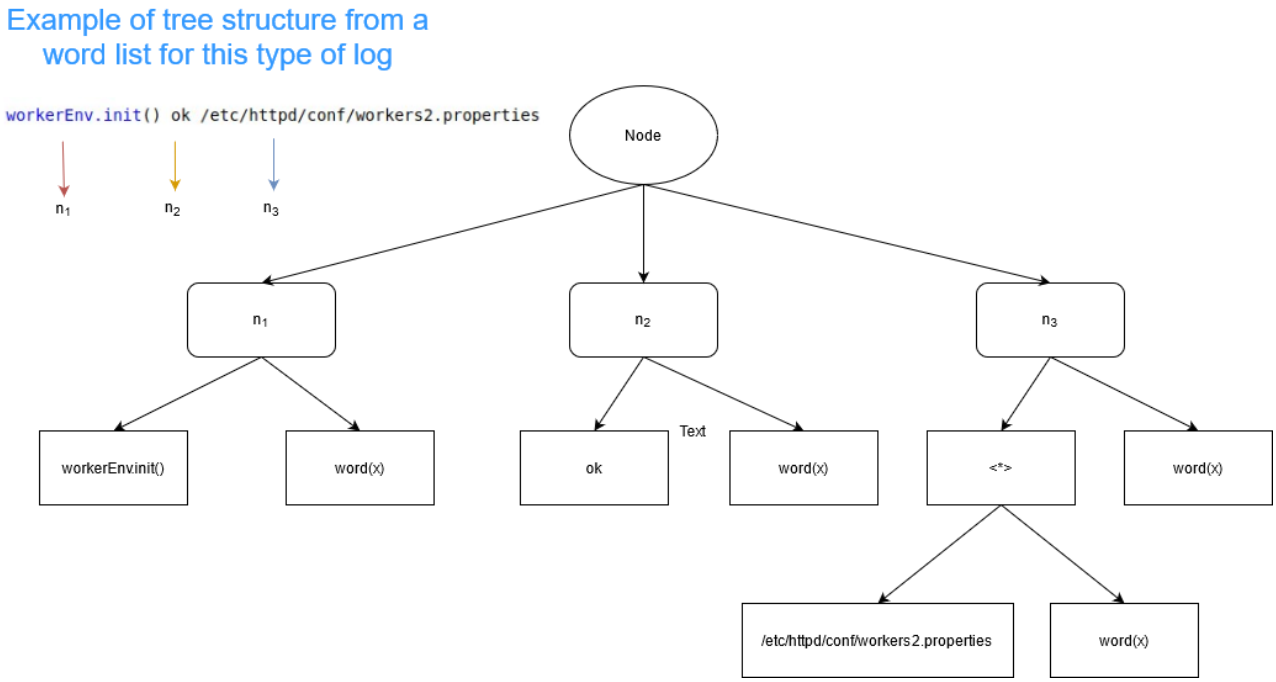


Figure 13: tree structure from a word list

We can observe that on the figure the structured tree is built from the log on the left. Each word of this log at a position "n_x". The tree will add each word according to its position. Word(x) corresponds to a word not yet indexed in the word list. We can see in case n_3 we got another sub-part into "<*>", this is corresponding to the others log with "workerEnv.init()" who has another path file. So, for all the logs with the same words at the beginning with the same position but on the word in the last position it's a different word. then the structured tree will modify itself and build a new subpart.

I will show you what the templates look like in different logs:

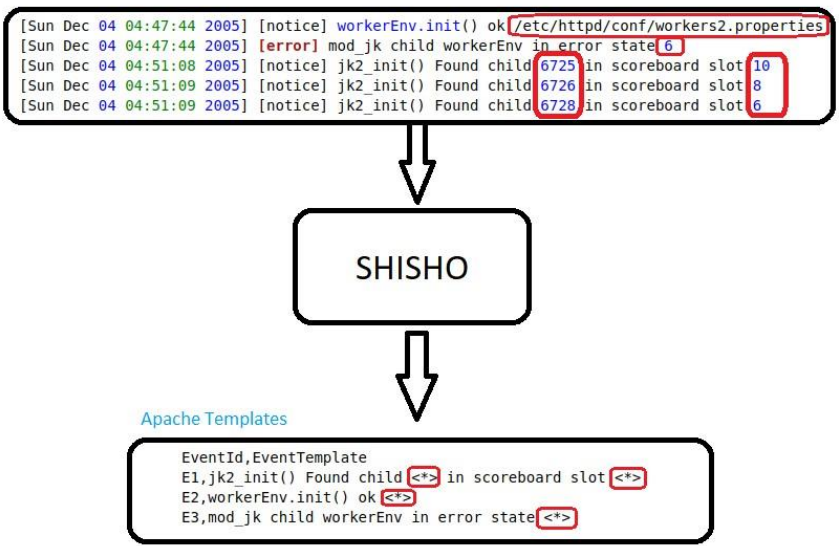


Figure 14: Apache templates

For this example, I take three different templates of logs. Of course, I don't show all the log but imagine the line 1 and 2 we have the same log with a different value inside the red box. The red box corresponds to the values that can change for this type of log. At the bottom of the figure with Apache templates, we can see once again the red box with this character "<*>". This corresponding to the different values that change for these templates.

Finally, after studying SHISO I'm goanna more interested in his pre-processing to make my script to have a structured log.

3.2 Free text

Free text is the information of log I find interesting to make a classification. To determine what information in a log can be interesting. I asked myself three questions that can be found in every log: "What? Who? Where or what is the action?". Let's see how I can apply it on Apache logs:

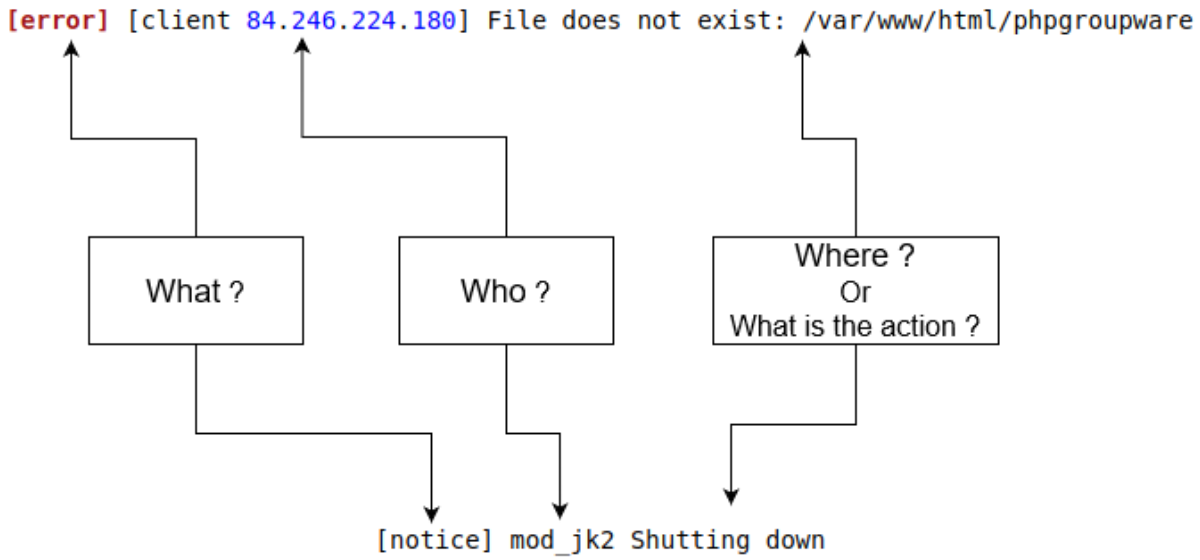


Figure 15: determining the free text

So, this is information I find useful and it's necessary to make a classification. However, this notion of "text" is really individual to each person. Right now, what I want is to extract his information. For that, I will use only the pre-processing SHISO into a script to structure my logs. Then I can extract this information when I have a structured file.

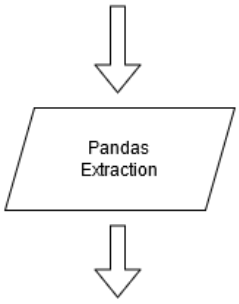
To explore my ".csv" structured file I use pandas library on Python 3.7. Pandas is an open-source Python package that is most widely used for data science/data analysis and machine learning tasks.

Pandas can:

- Load and save data
- Clean Data
- Visualize the data
- Do statistical analysis
- Add, replace, delete columns
- Add, replace, delete rows

Pandas can open ".csv" file to generate a Data Frame. A Data Frame is an object that represents a table of data with rows and columns. We can compare this to an Excel table. In the figure below you can see two data frames and the extraction of the previously determined free text :

	LineId	Date	Level	Content	EventId	EventTemplate
0	1	Sun Dec 04 04:47:44 2005	notice	workerEnv.init() ok /etc/httpd/conf/workers2.p...	e3dc5d82	workerEnv.init() ok /etc/httpd/conf/workers2.p...
1	2	Sun Dec 04 04:47:44 2005	error	mod_jk child workerEnv in error state 6	b13a7d5f	mod_jk child workerEnv in error state *
2	3	Sun Dec 04 04:51:08 2005	notice	jk2_init() Found child 6725 in scoreboard slot 10	ed253718	jk2_init() Found child * in scoreboard slot *



	Level	Content
0	notice	workerEnv.init() ok /etc/httpd/conf/workers2.p...
1	error	mod_jk child workerEnv in error state 6
2	notice	jk2_init() Found child 6725 in scoreboard slot 10

Figure 16: pandas extraction

To make this transformation I load my structured file into a pandas data frame. Then I drop some columns in the data frame like LineID, Date, EventId, and EventTemplate. This is given me a new data frame. Lastly, I export this data frame into a new CSV file. I will give you the script to do that, he is very short and easy to understand:

```
import pandas as pd

#Open csv file and create data frame
data=pd.read_csv('Apache_2k_log_structured.csv')

#remove columns (axis=1 represent axis of columns)
data=data.drop(['LineId','Date','EventTemplate','EventId'],axis=1)

#export into a csv file the new data frame
data.to_csv('freetext.csv',index=False)
```

Figure 17: Extract free text with panda library

To sum up, I use a log parser algorithm to structure my log file. Then I define what is useful information and what I need to make a classification of Apache log. Indeed, after determinate the free text I employ the pandas library to extract and clean the structured log into a new CSV file. With this file, I can go into the next step: the classification. To do this I will use machine learning.

4 Machine learning

What is Machine Learning? I will explain what Machine Learning is, which will allow you to understand how this technology works and its applications.

First, I will give you a definition of machine learning and its history. Machine Learning is a field of study in Artificial Intelligence that aims to give machines the ability to learn. This very powerful technology has allowed the development of autonomous cars, voice recognition, and autonomous systems. Machine Learning was invented by Arthur Samuel, an American computer scientist in 1959 after he developed the first checkers' program with artificial intelligence. The program won against the 4th best player in the United States.

Today, the main algorithms used in Machine Learning are statistical models developed from data. Among these models, we find for example decision trees, linear regression.

To understand how a machine can learn from data, we need to look at the 2 learning methods of Machine Learning:

- Supervised Learning
- Unsupervised Learning

I will present these different learning methods in the next part.

4.1 Supervised/Unsupervised Learning

I will start to present you first Supervised Learning. this is the “simplest” method to start machine learning.

4.1.1 Supervised Learning

Supervised Learning is used to develop predictive models, which mean models capable of predicting a certain value “Y” according to one or more variables “X”.

in the case of the Apache data that I used. we have Y for the label(error/notice) and X for the content (log message).

To develop these models, it is first necessary to provide the machine with a large amount of data (X, Y). We call this a dataset*. Then, we ask the machine to develop an approximation function that best represents the X -> Y relationship present in our data. For this, we use an optimization algorithm that minimizes the differences between the function and the data in the dataset.

Supervised Learning applications are very large. We can split them into 2 categories of problems: regressions, and classifications. I will present you only the classification because I work on it.

Classification problems correspond to situations in which the machine must predict the value of a qualitative variable. In other terms, the machine must classify what it is given in classes.

Let's see how it works with a figure. I took for my dataset a sun and star representation for the X and the name of these representations for Y.

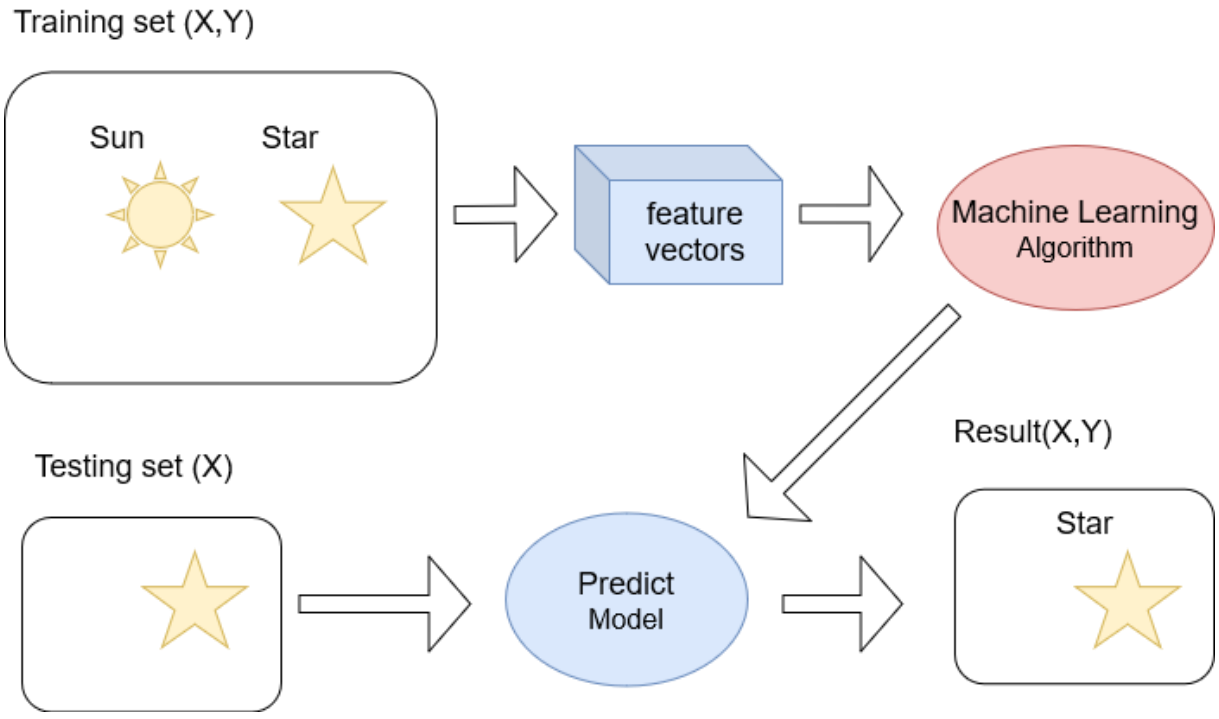


Figure 18: Example of a supervised algorithm with classification

In this example we gonna give a dataset(X,Y) into the algorithm to be able to predict a model, It's called the training set. First, we transform this training set into vectors X and Y because the algorithm can't interpret cannot directly interpret an image. Then he defines models with the training set. Next, we putting a testing set with only an image(X) into the predicted model. If all goes well, the algorithm gives a Y to the input X. So, with this example, the algorithm finds the good name of the image, in the output we got an image of a star with the classification "Star".

furthermore, supervised machine learning with classification can be used in vocal and image recognition.

4.1.2 Unsupervised Learning

there is another learning method. It's Unsupervised learning. This method is used when our dataset does not contain examples that indicate what we are looking for. We have only a dataset with an X. I explain to you with an example:

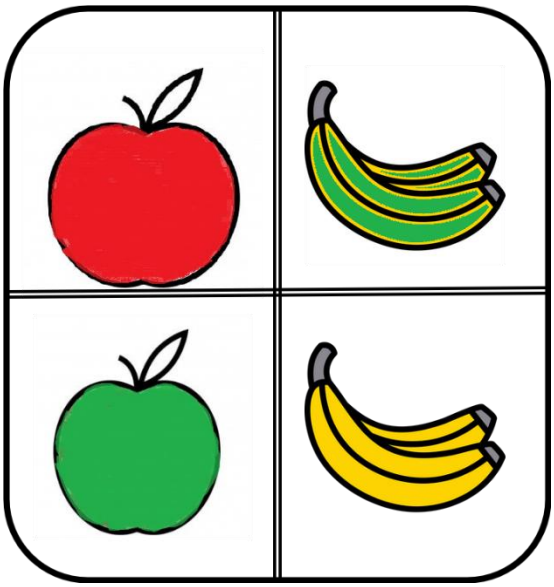


Figure 19: dataset for unsupervised learning

For us when we see these four fruits it easy to make a classification. Your brain recognized common structures in the data that you showed it. This is same for the unsupervised learning. The algorithm learns to recognize common structures in the dataset(X) we show it.

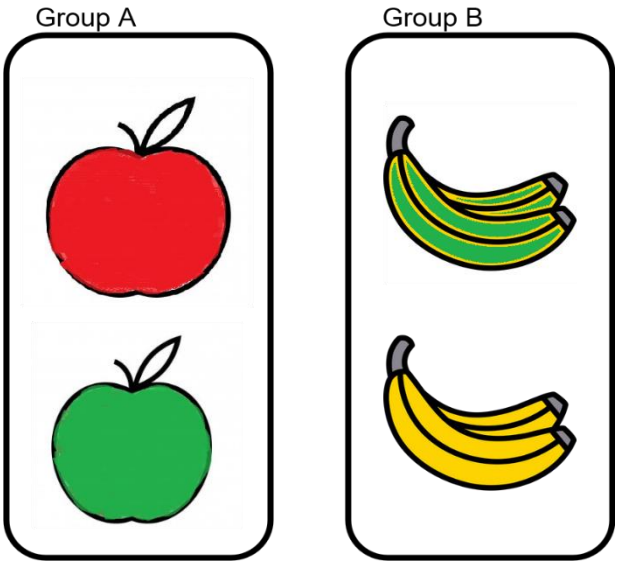


Figure 20: unsupervised learning clustering

We can group data in clusters and detect anomalies. In the figure above the algorithm make two clusters “group A” and “group B”. They are made depending on the shape of the fruit. But we can have three clusters if the algorithm takes the color for a feature. In the next part, I will explain to you what feature I take for my algorithm.

4.2 Bag of words

The bag of words model is a representation that converts text into vectors of fixed length by counting how many times each word appears. This process is often called vectorization. A bag of words can be a feature for supervised learning when we have text. Let’s do an example with two sentences:

- He likes Cisco
- He loves Cisco

Firstly, we made a list of different words: [“He”, “like”, “love”, “Cisco”]. We have only 4 words.

Secondly, we take this list to do a “table” and for each sentence, we will count the occurrence of each word.

Sentences	"He"	"likes"	"loves"	"Cisco"
"He likes Cisco"	1	1	0	1
"He loves Cisco"	1	0	1	1

Figure 21: Bag of words vectorization

Now we got a vectorized bag of words like this:

[[1101]
[1011]]

These two vectors can be interpreted now by an algorithm. So, each vector represents a feature of a sentence. A vector corresponding to an X in a dataset(X,Y). This bag of words with four words can be a training text for an algorithm. Now we want to predict a sentence but in this sentence, we got a new word: “He loves networks”.

This new word doesn’t appear in the bag of words. We goanna rework the bag of words:

Sentences	"He"	"likes"	"loves"	"Cisco"	"networks"
"He likes Cisco"	1	1	0	1	0
"He loves Cisco"	1	0	1	1	0
Predict sentence					
"He loves networks"	1	0	1	0	1

Figure 22: Bag of words vectorization reworked

As a result, the new bag of word change and now each vector have a length of 5. If we look in detail at the news vectors. In your opinion, what will be the prediction of the new sentence according to the two models generated previously? Which vector the sentence has more in common with the training set vector?

With the sentence “He loves Cisco” we have two words in common with “He loves Cisco” and just one word in common with “He likes Cisco”. I this case with this bag of words a supervised algorithm predicts “He loves networks” it’s the same model as “He loves Cisco”.

Now that we have understood how machine learning and bag of words work, I am going to present you the python library that allows you to realize this type of algorithm.

4.3 Scikit-learn

Scikit-learn is an open-source machine learning library that supports supervised and unsupervised learning. This library is based on other libraries like:

- NumPy: Base n-dimensional array package
- SciPy: Fundamental library for scientific computing
- Matplotlib: Comprehensive 2D/3D plotting
- IPython: Enhanced interactive console
- Sympy: Symbolic mathematics
- Pandas: Data structures and analysis

With scikit-learn we can do:

- Pre-processing
- Classification
- Clustering
- Model selection

Also, with Scikit-learn we can create a bag of words, I will show a little example of code to demonstrate how does it work with the previous example.

```
from sklearn.feature_extraction.text import CountVectorizer

corpus = [
' he likes Cisco',
' he loves Cisco'
]

vectorizer = CountVectorizer()
print( vectorizer.fit_transform(corpus).todense() )
print( vectorizer.vocabulary_ )
```

```
[[1 1 1 0]
 [1 1 0 1]]
{'he': 1, 'likes': 2, 'cisco': 0, 'loves': 3}
```

Figure 23: bag of words with scikit-learn

Here we have a list called “corpus”. In this list, I have two character strings. To make a bag of words I used the function CountVectorizer(). This function transforms each character strings in the list into a vector. In the last print, we have the list of the bag of words. For each word we have a number, this is corresponding to the index of this word in the bag of word vector. If I translate the bag of words readable for a human, we got the word “cisco” in the first position then “he” and “likes” to finish “loves”.

Now I will add a test set to see what will predict the algorithm:

```
Y=['likes','loves']

new=['he loves networks','he loves burger','he likes burger']
test=vectorizer.transform(new).toarray()

clf = tree.DecisionTreeClassifier()
clf = clf.fit(X, Y)

output=clf.predict(test)

print(output)
```

Figure 24: Predict model with scikit learn

This is the next part of the code in the previous figure. I give a label to the previous sentences, the first has a label “likes” and the second a label “loves”. I add 3 new sentences then I transform them into vectors for the test dataset(X). I create a decision tree classifier with my training dataset(X,Y). To finish I predict each sentence of the test set and I print the classification: `['loves' 'loves' 'likes']`

We can observe we got a label to defined each sentence, the first and second sentences have a class “loves” and the last have a class “likes”. The algorithm will predict the class of new sentences based on their similarity to the training test.

4.4 Classification algorithm for Apache log

Now we know how does work scikit learn. With supervised learning, I develop an algorithm to classify the content of the Apache log. First, select some logs into a structured log to have a dataset, I take 100 logs with 50 notices and 50 errors. You can find my script to extract and build a dataset in the annexes.

In the next step, I’m loading this dataset, for that, I used the pandas library. I’m put in a list “training” all the content of the log (the sentences) and at the same time, I create a list “Y” to put the label (notice or error) of the content. Of course, my list “Y” and “training” need to be synchronized.

```
training=[]

Y=[]

#add the label and the content into a list
for i in range(len(df)):
    if len(df.loc[i])<6 : #somtimes line can be corrupt, so i dodge him
        content=df.loc[i]['content']
        error=df.loc[i]['error']
        training.append(content)
        if error=='error':
            error=1
        else:
            error=0
        Y.append(error)
```

Figure 25: build the Apache dataset

In this part of the code, I use a loop “for” to explore all the data frame. For each line, I add the content to the list “training” and in the list “Y” I add 0 if is a notice or 1 for an error. It is better to use 0 or 1 in the list because the algorithm goanna interprets a faster number than a character string. I also do check before, because sometimes lines can be corrupt in the CSV file and they are not structured. Then I vectorized all the list “training” into a bag of words call “X”. Now we got the training dataset(X,Y).

At the same time to make the test set with only X. I use approximately the code than the previous one. I only take the content to put in a list and I vectorized the list.

I used the function: **clf.predict(train)** to predict the test set. For information, I used the same log for the training set and test set. If I print “**clf.predict(train)**” I got a list like this:

```
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
```

Figure 26: a print of clf.predict(train)

This list is not readable if you don’t create this code. But the “0” and “1” is only the class I used to define an error or a notice. And if you remember I used the same log for the training set and test set. So, in my test set, I have 50 notices and 50 errors. Here we have 50 “0” and 50 “1” corresponding to the notices and errors. In this case, I have 100% accuracy with this test set. Of course, now I need to convert “0” and “1” and put my prediction result into a CSV file.

```
df_output = pd.DataFrame(columns=['label', 'content'])
for i in range(len(output)):
    label=''
    if output[i]==1:
            label='error'
    else:
            label='notice'
    df_output.loc[i] = [label] + [test[i]]


df_output.to_csv('output_learning.csv')
```

Figure 27: convert predict model into a csv file

To output my result into a CSV file I need to create a pandas data frame. Then I use a loop for the length of the predict list. I add in the data frame the content of the test set and class of the predictive model. Of course, I convert “0” and “1” into error or notice. At the end, I export the data frame into a CSV file:

	A	B	C	D	E	F
1	,label,content					
2	0,notice,LDAP: SSL support unavailable					
3	1,notice,suEXEC mechanism enabled (wrapper: /usr/sbin/suexec)					
4	2,notice,Digest: generating secret for digest authentication ...					
5	3,notice,Digest: done					
6	4,notice,LDAP: Built with OpenLDAP LDAP SDK					
7	5,notice,LDAP: SSL support unavailable					

Figure 28:output_learning.csv

Here, we have the CSV file output. This file is structured, we got the correct label for each content. This file is the same one I used for the training test.

4.5 Result

In the previous case, I used the same log for the training set and test set. But what happens if I use a different test set? Now I use a similar test set but sometimes we don't have the same logs. We got the first 50 notices, then 50 errors in the test set.

Let's check the predicted model:

```
--------------------------------------------------
Predict Model :
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1]
--------------------------------------------------
output dataframe :
     label                                              content
0   notice  workerEnv.init() ok /etc/httpd/conf/workers2.p...
1   notice  workerEnv.init() ok /etc/httpd/conf/workers2.p...
2   notice  workerEnv.init() ok /etc/httpd/conf/workers2.p...
3   notice  workerEnv.init() ok /etc/httpd/conf/workers2.p...
4   notice  workerEnv.init() ok /etc/httpd/conf/workers2.p...
..     ...                                                ...
95   error  config.update(): Can't create worker.jni:onShu...
96   error                                mod_jk child init 1 0
97  notice  [client 213.205.73.192] Invalid method in requ...
98   error  [client 61.4.64.179] Directory index forbidden...
99   error  [client 61.10.7.8] Directory index forbidden b...

[100 rows x 2 columns]
--------------------------------------------------
accuracy score :
0.95
```

Figure 29: predict model with a new test set 100 logs (training set: 100 logs)

We can observe in the 50 errors, the algorithm predicts 5 notices which are supposed to be an error. So, we got a scoring accuracy of about 95%, which a good score but let's see with more logs.

Now I try with 100 logs for my training set and 1000 logs for the test set. You can find the screen in the annexes. The function for the score accuracy doesn't work when you have not the same dimension between the training and test set. So do manually the score accuracy. I find 14 logs that have a bad prediction into the 500 errors. We get an accuracy score of 97.5%.

For the 2000 log, I got an accuracy score of 97.5%. but now if I do with 2000 logs but use 1000 logs for my training set, we get an accuracy score of 98.4%.

We can deduct from these results when put more log in the training set, we reduce the possibility of the unknown log. Theoretically, the algorithm has less chance to make a mistake. That the limit of supervised learning. When we try to predict a new type of data the system hasn't been trained to recognize. There is a high chance that the results it delivers will be far from being true.

Moreover, if we use this kind of algorithm for network monitoring, what tells us that we know all the types of errors generated by our web service for example? Indeed, we may have to wait for a vulnerability to be detected manually by a human to know this type of log. it may be already too late it is a critical vulnerability.

Finally, supervised learning may not be the best solution for monitoring if we have to deal with unknown parameters for the algorithm. But if we know all the possible parameters and we teach him it is an excellent solution.

5 Assessment

This internship with Dalhousie University was very informative because of its international aspect and the discovery of a new field.

From an organizational point of view, it was interesting to work step by step however it may have slowed the progress. I wasn't late or early for all the steps. But arriving so close to the end in the case of the Learning machine I would have liked to have more time to test an algorithm of «unsupervised learning».

5.1 Technical skills

During this internship, I was introduced to many data science tools. During my DUT I have hardly used these tools. I discovered very quickly how to set up a virtual machine with Docker. I also discovered Jupiter is very used in data science. I understand why it is so used because the display console is optimized to show data frames and allows to make a "notebook". I also use data science libraries like Pandas and Scikit-Learn.

I have developed ease in programming, especially with python. I was a little afraid of this aspect, especially when I realized that the entire internship was going to be programming. Before starting this internship, I had some difficulties in programming with the manipulation of a character string, stack, and list. But working 12 weeks continuously on this aspect I gained skills.

I took the first step into the world of machine learning. It was a field completely unknown to me before. However, although I have discovered only a small part of this area, I have realized its usefulness and effectiveness for some tasks. I realized a rather simple «supervised learning» algorithm but this gave me a good basis to deepen my knowledge in the future.

During this internship, I had the chance to work with a team that aimed to make me learn new things. They decided to let me work in total autonomy and let me search by myself. I had to search on the internet for solutions to my problems via documentation and tutorials. I met many problems but it allowed me to learn and to be able to not make this kind of mistake anymore. I think that if I had been more supervised and helped, I would not have learned so much during this internship.

5.2 Professional skills

Being an intern is not only about technical skills, it is also an opportunity to meet people. This internship may have been in teleworking, but I had the chance to be supervised by a nice, pleasant, and always a present team. I regret not to have been able to meet more people in the laboratory, the teleworking stop a lot of social contacts.

During this internship, I also developed a lot of my autonomy, either thanks to the remote work which promotes this quality or by the choice of my internship master to let me fumble alone to get some knowledge. I think that this autonomy is an essential quality for any company but even more in the field of computer science which will be very useful for me.

Working in an international team can be scary for the first time, but it is a fantastic experience, which you should not hesitate to take. In my case, everyone was kind to me, and the many meetings in English were not a problem and were even very pleasant. As my internship progressed, I felt more and more comfortable with the language and I felt a lot of progress in both sentence construction and technical English. I also became more comfortable speaking in English.

Conclusion

Thus ends this 12-week internship in the NIMS lab, the research group of the Department of Computer Science at Dalhousie University. The goal of this internship was to make me understand the functioning of autonomous systems. The team presented us with a system they had developed for network monitoring.

After having understood the global idea of their project I studied a precise part of this project: the detection of an anomaly. Indeed, the system must be able to recognize errors when there are some errors. To progress more easily each week a new task was assigned to us.

First of all, I started by studying Apache, Snort, and firewall logs. I had to determine their structures and the useful information to determine an anomaly. Then I focused on the functioning of Machine Learning with the different types that exist, such as "supervised/unsupervised learning". I extracted the different features of each log necessary for a learning algorithm. Finally, I have realized my supervised algorithm to classify each Apache log.

Although not complete, because the algorithm only works with Apache logs. I realized only a supervised algorithm. It would have been interesting to realize an «unsupervised learning» algorithm. However, this experience was very rewarding:

From a theoretical point of view, I discovered the world of data science, I made my first steps in machine learning. I learned to use new tools such as Jupiter, scikit-learn, and pandas. I also strengthened my programming skills.

From a practical point of view, telecommuting may have slowed down my learning of English, but I still evolved. Working independently has allowed me to evolve at my own pace, to self-train, to research scientific papers.

From a personal point of view, I enjoyed this international experience, it was also the discovery of machine learning. And I hope that even though I'm not going directly into data science, this knowledge will be useful in my future professional life.

Lexicon

Buffer overflows	**Buffer overflows** can often be triggered by malformed inputs; if one assumes all inputs will be smaller than a certain size and the buffer is created to be that size, then an anomalous transaction that produces more data could cause it to write past the end of the buffer. If this overwrites adjacent data or executable code, this may result in erratic program behavior, including memory access errors, incorrect results, and crashes.
CGI	**Common Gateway Interface** (**CGI**) is an interface specification that enables web servers to execute an external program, typically to process user requests.
CSV	**comma-separated values** (CSV) file is a delimited text file that uses a comma to separate values. Each line of the file is a data record. Each record consists of one or more fields, separated by commas. The use of the comma as a field separator is the source of the name for this file format. A CSV file typically stores tabular data (numbers and text) in plain text, in which case each line will have the same number of fields.
IP address	An **Internet Protocol address** (**IP address**) is a numerical label assigned to each device connected to a computer network that uses the Internet Protocol for communication. An IP address serves two main functions: host or network interface identification and location addressing.
Machine learning	the process of computers changing the way they carry out tasks by learning from new data, without a human being needing to give instructions in the form of a program
Open-source	The term **open source** typically refers to a program whose source code is released for use or modification by the community. Developers are free to download and make changes to the code as they please and create their own personalized product.
OS fingerprinting	**OSfingerprinting** is the art of identifying the operating system of a remote server. In practice, this step often follows a port scan. Although TCP/IP communications are subject to standards, the implementations of these protocols are different depending on the OS. Let's discover these different techniques to distinguish the different operating systems.
Pre processing	**pre-processing** is an important step in the data mining process. This is particularly applicable to data mining and machine learning projects. Data-gathering methods are often loosely controlled, resulting in out-of-range values, impossible data combinations, and missing values,
SHISO	Scalable Handler for Incremental System log
SMB	**Server Message Block** is a network protocol used by Windows-based computers that allows systems within the same network to share files. It allows computers connected to the same network or domain to access files from other local computers as easily as if they were on the computer's local hard drive.
Stealth port scan	Mechanism to perform reconnaissance on a network while remaining undetected. Uses SYN scan, FIN scan, or other techniques to prevent logging of a scan

Figure Table

Figure 1: Dalhousie university, Halifax map 7
Figure 2: Gantt chart: organization of the internship 10
Figure 3: Apache logs 11
Figure 4 : Apache log format 11
Figure 5: Snort logs 12
Figure 6: Snort log format 12
Figure 7: IP header (source https://en.wikipedia.org/wiki/IPv4) 13
Figure 8: Firewall logs 14
Figure 9: Firewall log format 14
Figure 10: unstructured to structured Apache log 15
Figure 11: final Apache log format from SHISO 16
Figure 12: tree structure from a word list 17
Figure 13: Apache templates 17
Figure 14: determining the free text 18
Figure 15: pandas extraction 19
Figure 16: Extract free text with panda library 20
Figure 17: Example of a supervised algorithm with classification 22
Figure 18: dataset for unsupervised learning 23
Figure 19: unsupervised learning clustering 23
Figure 20: Bag of words vectorization 24
Figure 21: Bag of words vectorization reworked 25
Figure 22: bag of words with scikit-learn 26
Figure 23: Predict model with scikit learn 26
Figure 24: build the Apache dataset 27
Figure 25: a print of clf.predict(train) 28
Figure 26: convert predict model into a csv file 28
Figure 27:output_learning.csv 29
Figure 28: predict model with a new test set 100 logs (training set: 100 logs) 30

Annexes

Script to extract free text from CSV file:

```
# -*- coding: utf-8 -*-
"""
Created on Tue May  4 09:29:43 2021

@author: Yann Kerdoncuff
"""
import pandas as pd
import numpy as np

data=pd.read_csv('Apache_2k_log_structured.csv')
print(data.shape)
print(data.columns)
print(data.head)
"""print(data.head)"""
data=data.drop(['LineId','EventTemplate','EventId'],axis=1)
print(data.head)
"""print(data.describe())"""
data.to_csv('freetext.csv',index=False)
```

To compare different process to get free text with SHISO algorithm I make a script to have a report between two version of free text:

```
def template_line(df):
    numbers_of_lines = df['Occurrences'].sum()
    numbers_of_rows=len(df.index)
    heterogeneity=numbers_of_rows/numbers_of_lines
    #output data
    numbers_of_templates='Numbers of templates : '+str(numbers_of_rows)
    numbers_of_lines='Numbers of lines : '+str(numbers_of_lines)
    heterogeneity='heterogeneity : '+str(heterogeneity)
    file1.writelines(numbers_of_templates+'\n')
    file1.writelines(numbers_of_lines+'\n')
    file1.writelines(heterogeneity+'\n\n')


#open csv files
df1=pd.read_csv(cautious_file)
df2=pd.read_csv(bold_file,error_bad_lines=False)

#open txt file
file1 = open(name_file,"w")

#heterogeneity
```

```
file1.writelines('Heterogeneity Bold version\n')
file1.writelines('-------------------------------------------------\n')
template_line(df1)
file1.writelines('Heterogeneity Cautious version\n')
file1.writelines('-------------------------------------------------\n')
template_line(df2)

#compare with datacompy library
compare = datacompy.Compare(
    df1,
    df2,
    join_columns='EventTemplate',  #You can also specify a list of columns
    abs_tol=0, #Optional, defaults to 0
    rel_tol=0, #Optional, defaults to 0
    df1_name='Cautious', #Optional, defaults to 'df1'
    df2_name='Bold' #Optional, defaults to 'df2'
    )
compare.matches(ignore_extra_columns=False)


report=compare.report()


file1.writelines(report)
occurence()

file1.close()
```

With this script we got a text file with the occurrence of templates, and different templates. Let's see the report Apache compare:

```
Heterogeneity Bold version
-------------------------------------------------
Numbers of templates : 34
Numbers of lines : 51618
heterogeneity : 0.0006586849548607075

Heterogeneity Cautious version
-------------------------------------------------
Numbers of templates : 37
Numbers of lines : 51887
heterogeneity : 0.0007130880567386822

DataComPy Comparison
--------------------

DataFrame Summary
-----------------
```

```
  DataFrame  Columns  Rows
0  Cautious        3    34
1      Bold        3    37

Column Summary
--------------

Number of columns in common: 3
Number of columns in Cautious but not in Bold: 0
Number of columns in Bold but not in Cautious: 0

Row Summary
-----------

Matched on: eventid
Any duplicates on match values: No
Absolute Tolerance: 0
Relative Tolerance: 0
Number of rows in common: 34
Number of rows in Cautious but not in Bold: 0
Number of rows in Bold but not in Cautious: 3

Number of rows with some compared columns unequal: 0
Number of rows with all compared columns equal: 34

Column Comparison
-----------------

Number of columns compared with some values unequal: 0
Number of columns compared with all values equal: 3
Total number of values which compare unequal: 0

Sample Rows Only in Bold (First 10 Columns)
-------------------------------------------

     eventid                                                                e
venttemplate  occurrences
36  06ccbbbc                          child process * still did not exit sendi
ng a SIGTERM         168
34       sqs
        asas           56
35  88e87b54  env.createBean2(): Factory error creating channel.jni:jni ( chan
nel.jni jni)           45

-------------------------------------------------------
maintained kept templates : 34
increased kept templates : 0
decreased kept templates : 0
```

Classification algorithm for Apache log :

```
# -*- coding: utf-8 -*-
"""
Created on Fri Jun 18 11:41:12 2021

@author: Yann Kerdoncuff
"""
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer
from sklearn import tree
from sklearn.metrics import accuracy_score

vectorizer = CountVectorizer()

df_test=pd.read_csv('error_2k.csv')

df=pd.read_csv('error_1k.csv')


training=[]

Y=[]

#add the label and the content into a list
for i in range(len(df)):
    if len(df.loc[i]['index'])<6 : #somtimes line can be corrupt, so i dodge h
im
        content=df.loc[i]['content']
        error=df.loc[i]['error']
        training.append(content)
        if error=='error':
            error=1
        else:
            error=0
        Y.append(error)

X = vectorizer.fit_transform(training)


test=[]

for i in range(len(df_test)):#run all line of the dataframe
    if len(df_test.loc[i])<6 : #somtimes line can be corrupt
        content=df_test.loc[i]['content']
        test.append(content)
```

```
train=vectorizer.transform(test).toarray()
print('-----------------------------------------------------')

print(X.shape)

print('-----------------------------------------------------')
print( vectorizer.fit_transform(training).todense() )
print('-----------------------------------------------------')
print( vectorizer.transform(test).toarray() )
print('-----------------------------------------------------')

clf = tree.DecisionTreeClassifier()
clf = clf.fit(X, Y)

output=clf.predict(train)



print("Predict Model :")
print(output)

x=0

df_output = pd.DataFrame(columns=['label', 'content'])
for i in range(len(output)):
    label=''
    if output[i]==1:
            label='error'
            x=x+1
    else:
            label='notice'
    df_output.loc[i] = [label] + [test[i]]




df_output.to_csv('output_learning.csv')
print('-----------------------------------------------------')
print("output dataframe :")
print(df_output)
print('-----------------------------------------------------')
print('total errors')
print(1000-x)
#print(accuracy_score(Y,output))
```

This is my supervised learning algorithm for Apache log, let's see a different result:

Result for 1000 logs in the training set and 1000 logs for the test set :

```
Predict Model :
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
-----------------------------------------------------
output dataframe :
      label                                            content
0    notice                       LDAP: SSL support unavailable
1    notice  suEXEC mechanism enabled (wrapper: /usr/sbin/s...
2    notice  Digest: generating secret for digest authentic...
3    notice                                       Digest: done
4    notice                  LDAP: Built with OpenLDAP LDAP SDK
..      ...                                                ...
976   error  [client 24.163.159.161] File does not exist: /...
977   error  [client 24.163.159.161] File does not exist: /...
978   error  [client 24.163.159.161] File does not exist: /...
979   error  [client 24.163.159.161] File does not exist: /...
980   error  [client 24.163.159.161] File does not exist: /...

[981 rows x 2 columns]
-----------------------------------------------------
accurary score :
1.0
```

Result for 1000 logs in the training set and 2000 logs for the test set :

```
--------------------------------------------------------
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]
--------------------------------------------------------
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]
--------------------------------------------------------
Predict Model :
[0 0 0 ... 1 1 1]
--------------------------------------------------------
output dataframe :
       label                                              content
0     notice                          LDAP: SSL support unavailable
1     notice  suEXEC mechanism enabled (wrapper: /usr/sbin/s...
2     notice  Digest: generating secret for digest authentic...
3     notice                                          Digest: done
4     notice                  LDAP: Built with OpenLDAP LDAP SDK
...      ...                                                 ...
1995   error  [client 66.219.55.2] Directory index forbidden...
1996   error  [client 218.89.93.99] Directory index forbidde...
1997   error  [client 222.245.99.74] Directory index forbidd...
1998   error  [client 61.138.111.18] Directory index forbidd...
1999   error  [client 66.30.172.178] Directory index forbidd...

[2000 rows x 2 columns]
--------------------------------------------------------
total errors
16
```

Result for 2000 logs in the training set and 1000 logs for the test set :

```
-----------------------------------------------------
Predict Model :
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]
-----------------------------------------------------
output dataframe :
      label                                         content
0    notice                      LDAP: SSL support unavailable
1    notice  suEXEC mechanism enabled (wrapper: /usr/sbin/s...
2    notice  Digest: generating secret for digest authentic...
3    notice                                       Digest: done
4    notice                 LDAP: Built with OpenLDAP LDAP SDK
..      ...                                                ...
976   error  [client 24.163.159.161] File does not exist: /...
977   error  [client 24.163.159.161] File does not exist: /...
978   error  [client 24.163.159.161] File does not exist: /...
979   error  [client 24.163.159.161] File does not exist: /...
980   error  [client 24.163.159.161] File does not exist: /...

[981 rows x 2 columns]
```

For this run, the accuracy score is false because some lines in the CSV as corrupt.

I make a python script to extract other features like words occurrences and the string length:

```
# -*- coding: utf-8 -*-
"""
Created on Mon Jun 14 09:43:38 2021

@author: Yann Kerdoncuff
"""

import pandas as pd

def word_count(str):
    counts = dict()
    words = str.split()

    for word in words:
        if word in counts:
                counts[word] += 1
        else:
                counts[word] = 1

    return counts

df=pd.read_csv('error_1k.csv')

df1=df[0:1000]

#word=dict()
#df1.columns=['index','error','content']



print(df1)


for i in range(0,1000):
    if len(df1.loc[i]['index'])<6 : #somtimes line can be corrupt
        ligne=df1.loc[i]['content']
        counter=word_count(ligne)
        nb_words=sum(counter.values())
        length=len(ligne)
        df1.loc[i, 'nb_words']=nb_words
        df1.loc[i, 'length']=length




#print(word)
```

```
#df1['nb_word']=word.values()

print(df1)

df1.to_csv('feature.csv',index=False,header=True)
```

I don’t use these features in my supervised algorithm but if I want to have a better accuracy this can be useful.

Abstract

Dalhousie University welcomed me as an intern from April 12, 2021, to July 2, 2021. 12 weeks of internship to validate my DUT networks and telecommunications. she allowed me to integrate an international research group: the NIMS lab. I learned to use many useful tools for anyone who wants to work in the field of data sciences.

The NIMS lab is focused on autonomous systems. They study the learning capacity of the systems and also monitor their behavioral changes according to their environment. The lab aims it's to make more and more aware system to their environment.

In the context of networks, we can use an autonomous monitoring system to observe the traffic as well as security holes on the network. This allows to have efficient monitoring in real-time and limits the intervention of a human. This tool is also very efficient to detect failures to intervene as soon as possible. NIMS has developed a system called HADES (Holistic Anomaly Detection for Enterprise Security) to observe network service logs. The system will evaluate and make statistics to monitor a network for an enterprise.

In my internship, I will be interested in the part of a system allowing to detect if a log has an anomaly or not. To do this I will first analyze logs, study their structures for different network services. Then I will study the functioning of this type of system which uses machine learning. Finally, I will realize my machine learning algorithm.

This internship allowed me to develop my technical skills and knowledge but also my professional skills in more than just its international aspect.