IA645: Data Analysis for Cybersecurity

Linux syslog Data Analysis and Machine Learning

Team: Aaron Liske, Mia Jones, William Smith III, Yogesh Chavarkar
Faculty Guidance: Dr Omar Darwish

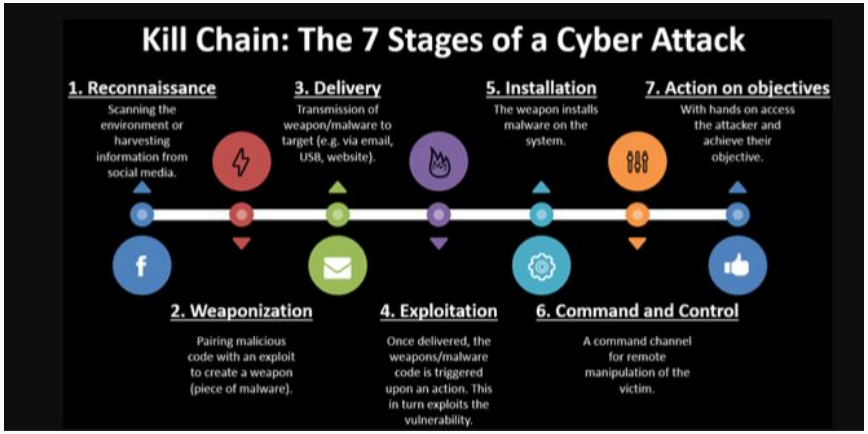
Game Above College of Engineering and Technology
Eastern Michigan University

Agenda

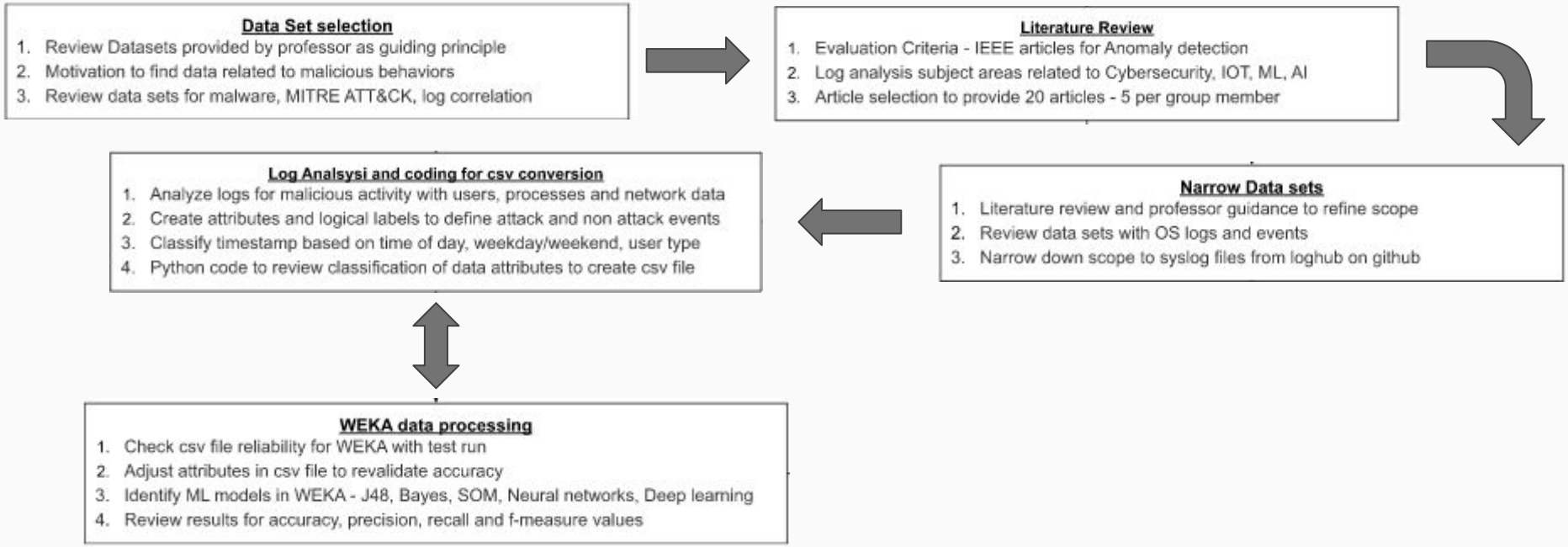
- Introduction
- Proposed Work
- Results
- Limitation
- Conclusion
- Future Work

Introduction

- Goal is to aid detection and classification of malware using log file analysis using ML.
- Classification of outbound connections from compromised systems



Workflow


Data Set selection
1. Review Datasets provided by professor as guiding principle
2. Motivation to find data related to malicious behaviors
3. Review data sets for malware, MITRE ATT&CK, log correlation
Literature Review
1. Evaluation Criteria - IEEE articles for Anomaly detection
2. Log analysis subject areas related to Cybersecurity, IOT, ML, AI
3. Article selection to provide 20 articles - 5 per group member
Narrow Data sets
1. Literature review and professor guidance to refine scope
2. Review data sets with OS logs and events
3. Narrow down scope to syslog files from loghub on github
Log Analsysi and coding for csv conversion
1. Analyze logs for malicious activity with users, processes and network data
2. Create attributes and logical labels to define attack and non attack events
3. Classify timestamp based on time of day, weekday/weekend, user type
4. Python code to review classification of data attributes to create csv file
WEKA data processing
1. Check csv file reliability for WEKA with test run
2. Adjust attributes in csv file to revalidate accuracy
3. Identify ML models in WEKA - J48, Bayes, SOM, Neural networks, Deep learning
4. Review results for accuracy, precision, recall and f-measure values

Dataset Selection

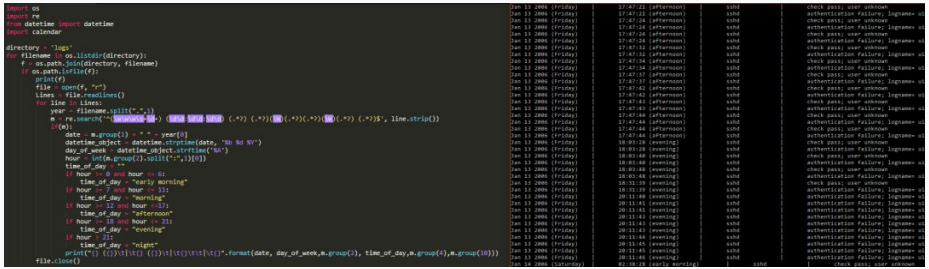
- Objective: identify malicious patterns found in authentication failures, ftp, https and nfs-rpc connections.
- Scope of the project is to focus on a single area compromising of operating system
- Dataset used was linux syslog data from var/log/messages.

Tools Used

- Visual Studio Code
- EMU Azure Tenant
- Log analytics workspace
  - Custom tables linuxlogs_CL and ia645nixlogs_CL
- SQL database
- Python

Codes and Scripts Used

- Extract, Transform, and Load (ETL) methods were used in a Python script to transform the data from the logs into a usable CSV that Weka could readily use.



```
#TODO: move user level to appropriate function for output instead of dual outputs for brevity
def message_parse(svc, msg, timestamp):
    #gain access to the variables needed
    global prev_service
    global prev_host
    global prev_timestamp

    #set variable defaults
    suspicious = 'NON-ATTACK'
    user_level = 'no_euid'

    #because sshd and samba have such similar messages, processing is identical
    if svc == 'sshd' or svc == 'samba':
        m = re.search('^(.*?) (euid=)(\d+) (.*?)$', msg)
        if(m):
            user_level = user_type(m.group(3))
            if user_level == 'root':
                suspicious = 'ATTACK'
            else:
                suspicious = 'NON-ATTACK'
        else:
            suspicious = 'NON-ATTACK'

    #process log for ftp(d).  mark as suspicious if line matches previous entry's remote host or is within 2 seconds per timestamp
    if svc == 'ftpd' or svc == 'ftp':
        m = re.search(r"^(.*?)(\d+\.\d+\.\d+\.\d+)(.*?)$", msg.replace('\.','.'))
        curr_host = ''
        if(m):
            curr_host = m.group(2)
        if prev_service == 'ftpd' or prev_service == 'ftp':
            if prev_timestamp != '':
                #parse out the seconds for the timestamp
                #TODO: add logic for seconds under 2 to loop back to 59 for timestamp checking
                prev_seconds = int(prev_timestamp.split(":")[2])
                curr_seconds = int(timestamp.split(":")[2])
                delta_time = curr_seconds - prev_seconds
                if (curr_seconds - prev_seconds < 2 or curr_host == prev_host) and (msg.strip() != "FTP session closed"):
                    suspicious = 'ATTACK'
                else:
                    suspicious = 'NON-ATTACK'
            #set host and timestamp variables for comparison later
            prev_host = curr_host
            prev_timestamp = timestamp
        else:
            suspicious = 'NON-ATTACK'
```

Log Analysis

Python Used

- Utilized regex to pull each log message apart to find the relevant sections for processing

```
^(\w\w\w\s+\d+) (\d\d:\d\d:\d\d) (.*?)
(.*?)(\W)(.*?)(.*?)(\W)(.*?) (.*?)$
```

Main Regex for Processing Log Line

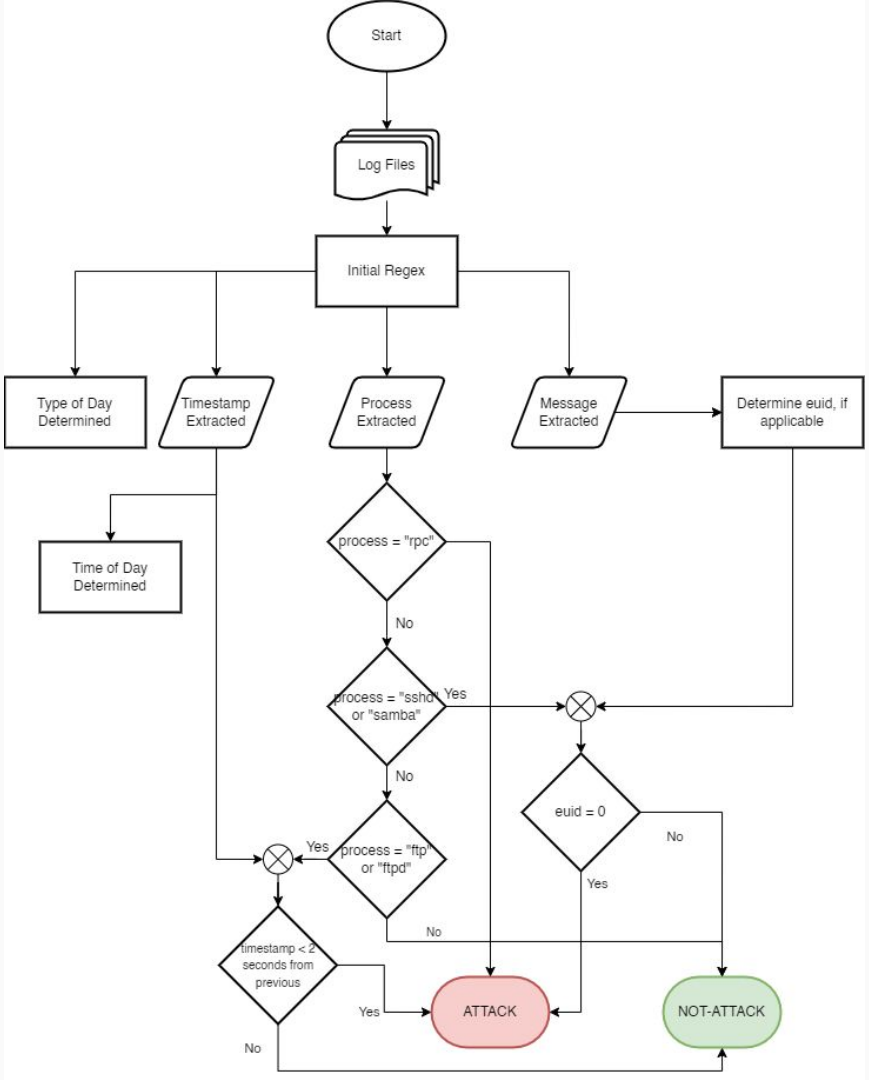
Regex Used

- Match groups used are the date, time, process, and the message

```
if hour >= 0 and hour <= 6:
      time_of_day = 'early morning'
if hour >= 7 and hour <= 11:
      time_of_day = 'morning'
if hour >= 12 and hour <=17:
      time_of_day = 'afternoon'
if hour >= 18 and hour <= 21:
      time_of_day = 'evening'
if hour > 21:
      time_of_day = 'night'
```

Determination of Time of Day

Individual Log Entry Flowchart



Weka Data Processing

- CSV below shows accuracy in a range of 93.65-90.03% running classifiers J48, BayesNet, and Naive Bayes. .

TYPE_OF_DAY	TIME_OF_DAY	PROCESS	USER_LEVEL	SUSPICIOUS
NON_WORKING	early morning	syslogd	no_euid	FALSE
NON_WORKING	early morning	syslogd	no_euid	FALSE
WORKING	morning	syslogd	no_euid	FALSE
WORKING	morning	syslog	no_euid	FALSE
WORKING	morning	syslog	no_euid	FALSE
WORKING	morning	kernel	no_euid	FALSE

- Data set further refined in the CSV below to higher consistency 90.02-90.03%.
- User level category was removed and suspicious changed to attack or non-attack.

TYPE_OF_DAY	TIME_OF_DAY	PROCESS	SUSPICIOUS
NON_WORKING	early morning	syslogd	NON-ATTACK
NON_WORKING	early morning	syslogd	NON-ATTACK
WORKING	morning	syslogd	NON-ATTACK
WORKING	morning	syslog	NON-ATTACK
WORKING	morning	syslog	NON-ATTACK
WORKING	morning	kernel	NON-ATTACK

Dataset Summary

Data Type: Linux syslog files labeled for suspicious activity
File Type: CSV
Name: WEKA Dataset
Instances: 71704
Attributes: 4

- TYPE_OF_DAY
- TIME_OF_DAY
- PROCESS
- SUSPICIOUS

Algorithms

Decision Tree - J48

Accuracy	Precision	Recall	F1-Measure
90.03	1.0, 0.77, 0.923	0.850, 1.0, 0.9	0.92, 0.87, 0.90
Inference: Folds 2, 5, and 10 with the same accuracy. Split 66% is slightly higher with 93.9% accuracy.			

Naive Bayes

Accuracy	Precision	Recall	F1-Measure
90.02	1.0, 77, .923	0.850, 1.0, 090	0.919, 0.870, 0.903
Inference: Folds 2, 5, and 10 with the same accuracy. Split 66% is slightly higher with 90.32% accuracy			

Algorithms

Bayes Net

Accuracy	Precision	Recall	F1-Measure
90.02	1.0, 0.77, 0.92	0.85, 1.0, 0.90	0.92, 0.87, 0.90
Inference: Folds 2, 5, and 10 with the same accuracy. Split 66% is slightly higher with 90.32% accuracy			

Random Tree

Accuracy	Precision	Recall	F1-Measure
90.03	1.0, 0.770, 0.92	0.85, 1.0, 0.900	0.92, 0.87, 0.90
Inference: Folds 2, 5, and 10 with the same accuracy. Split 66% is slightly higher with 90.32% accuracy			

SMO Support Vector Machine

Accuracy	Precision	Recall	F1-Measure
90.03	1.0, 0.77, 0.92	0.85, 1.0, 0.9	0.92, 0.87, 0.90
Inference: Folds 2, 5, and 10 with the same accuracy. Split 66% is slightly higher with 90.32% accuracy			

Multilayer Perceptron (Deep Learning Neural Network)

Accuracy	Precision	Recall	F1-Measure
90.03	1.0, 0.77, .92	0.95, 1.0, 0.9	0.92, 0.87, 0.90
Inference: Similar results using the “Use training set” test option.			

Limitation

- Services in a Linux System
- Retaining User Data

Conclusion

- “Stepping stone" in predicting attacks in near-real time
- Learns from behaviors and patterns, and not dependant on hard coded signatures
- Classifies outbound connections from compromised systems

Future Work

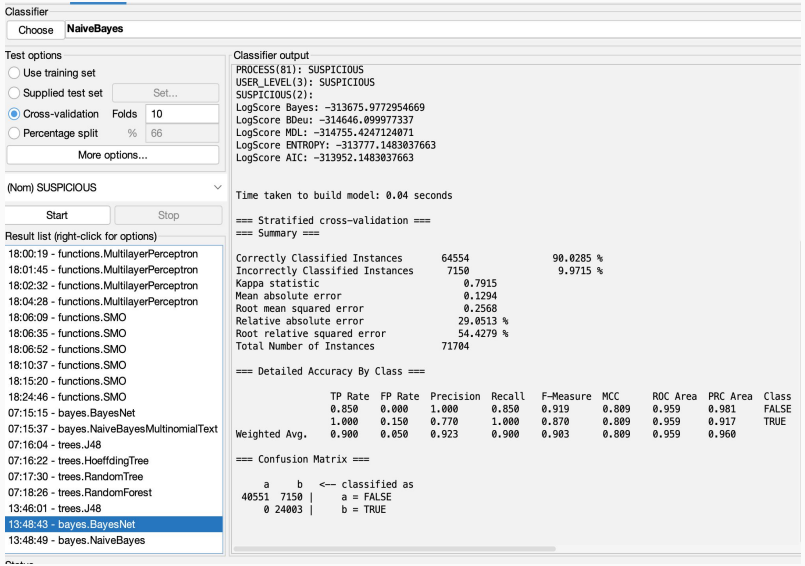
This Machine Learning tool can effectively work in two different cybersecurity functions

- Compliance using syslogs showing exposure of misconfigured assets
- Attack prediction for threat hunting and collecting Cyber threat Intelligence from observed attack campaigns

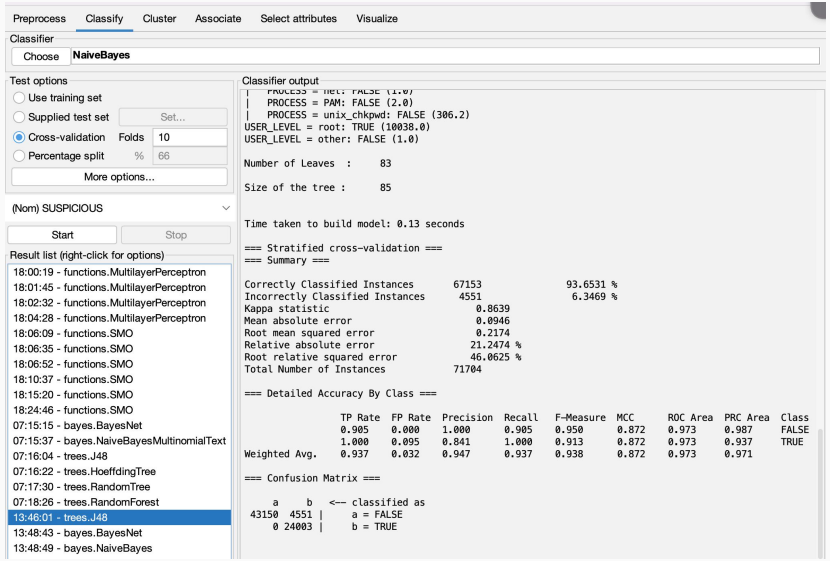
Appendix

Weka Data Processing

BayesNet Classifier

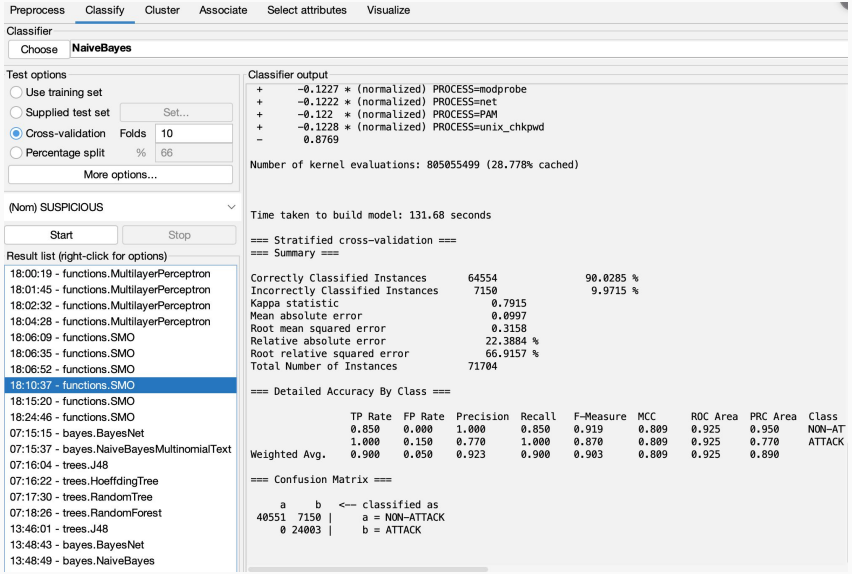


J48 Classifier

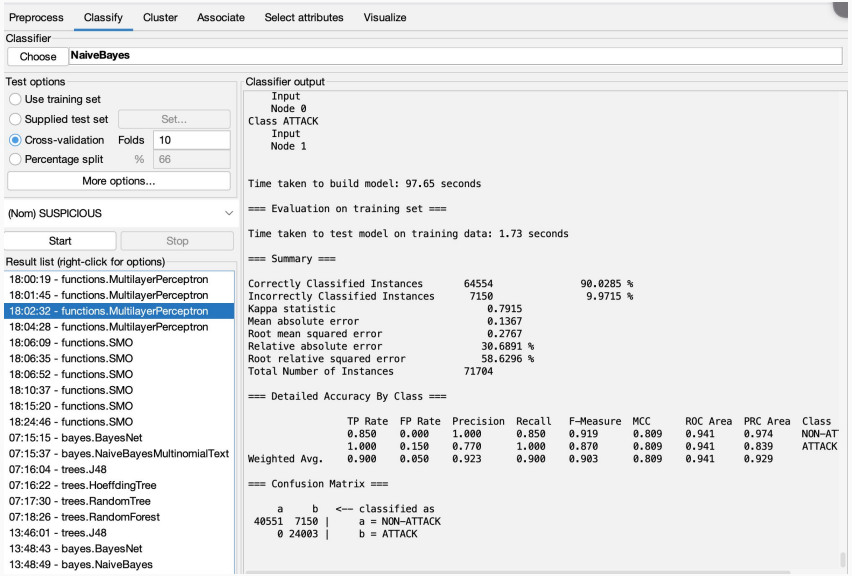


Weka Data Processing

- SMO- Support Vector Machine Classifier

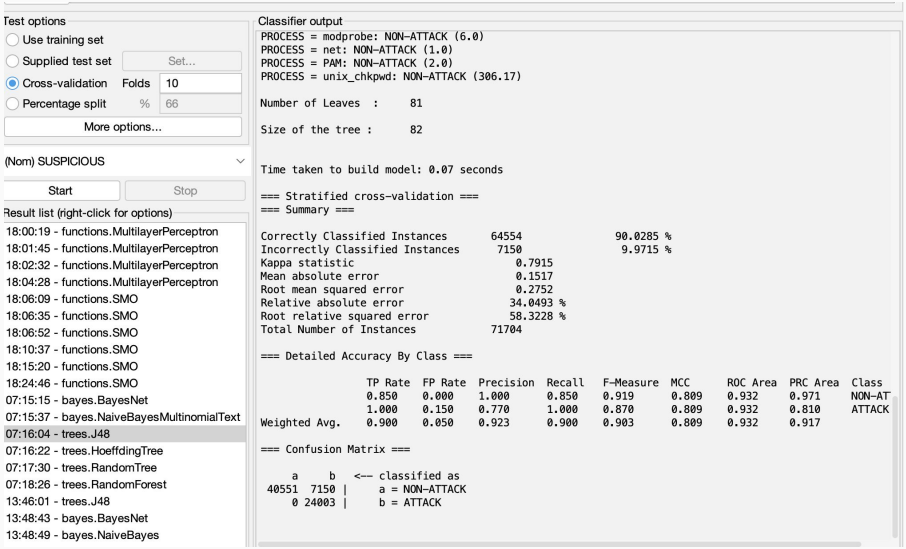


- Multilayer Perceptron

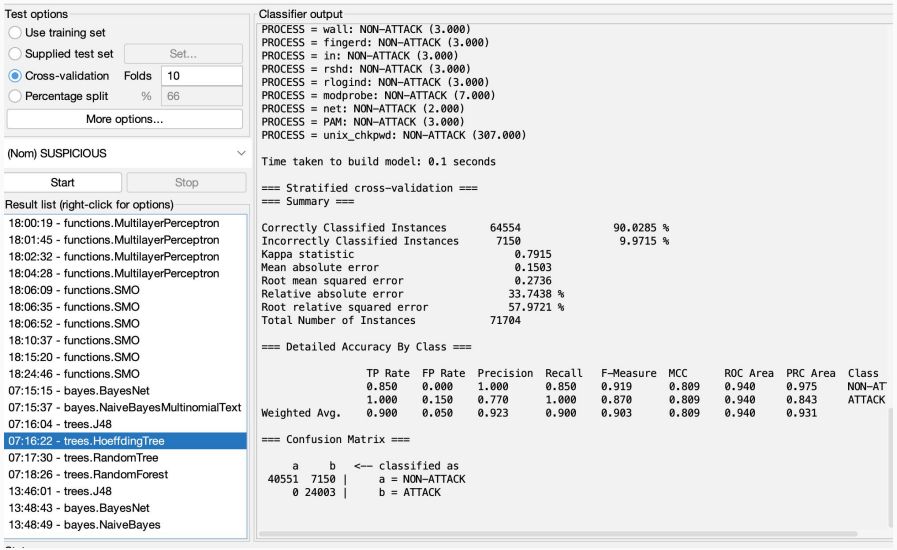


Weka Data Processing

- J48

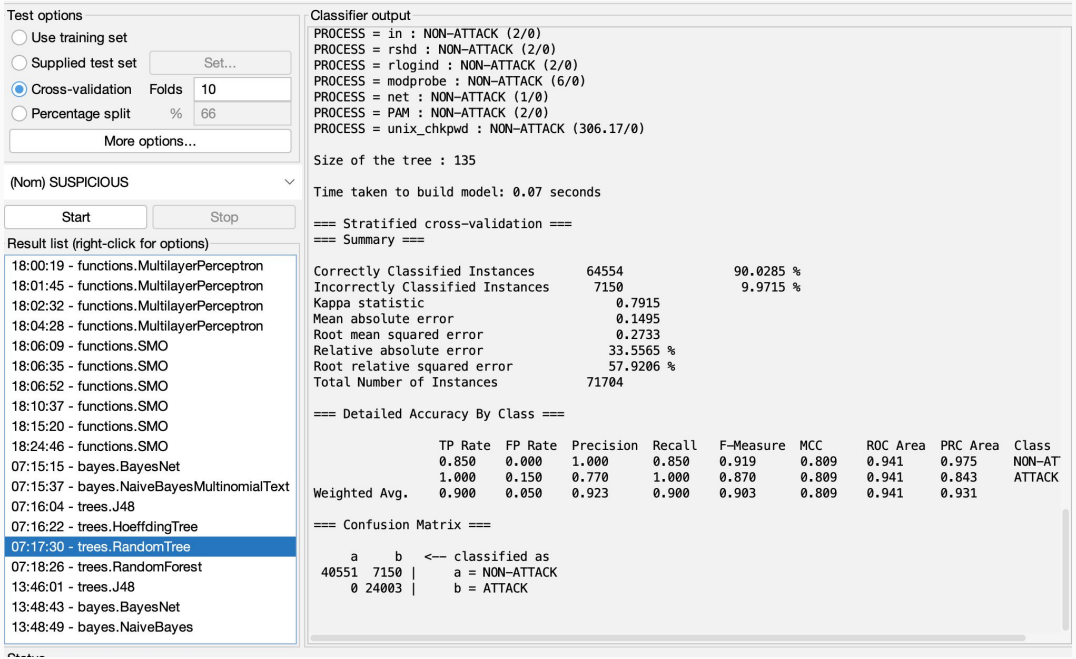


- HoeffdingTree Classifier



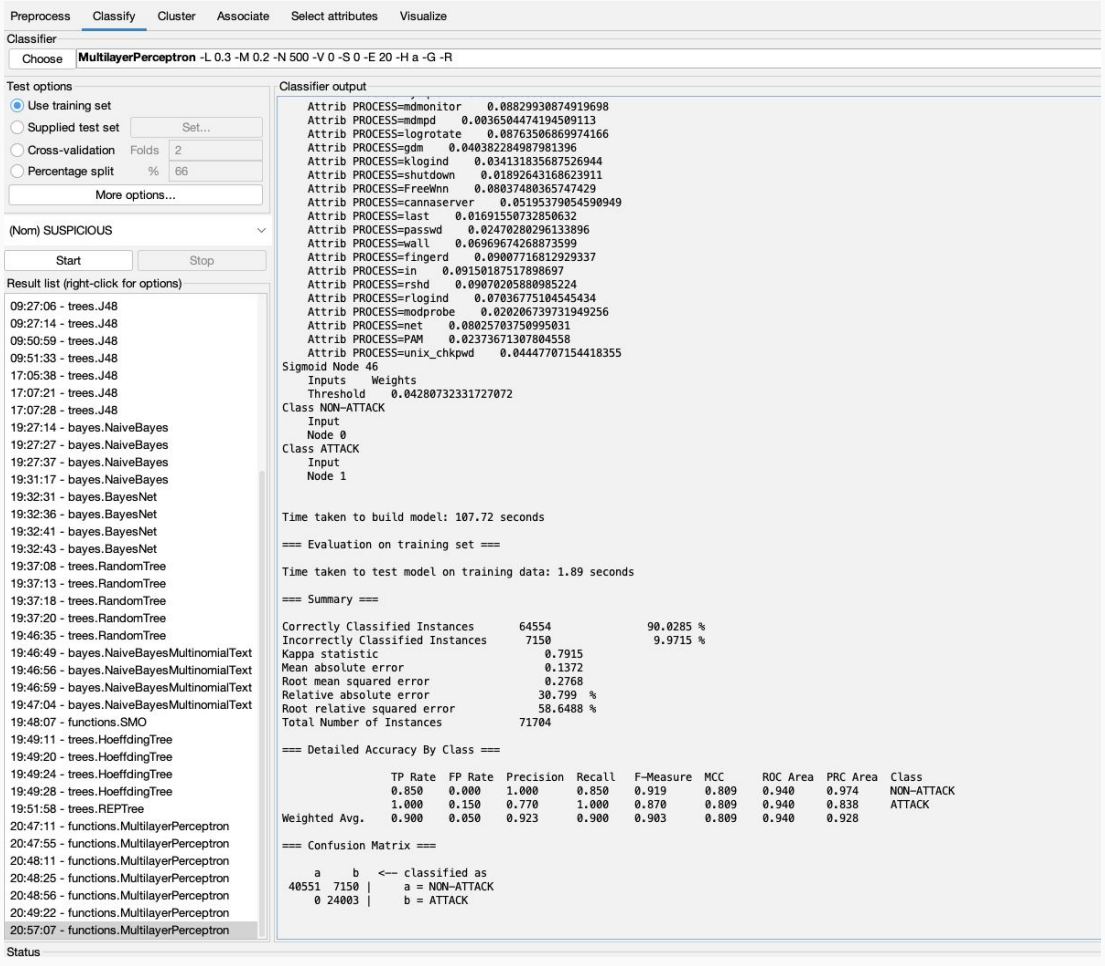
Weka Data Processing

- RandomTree Classifier



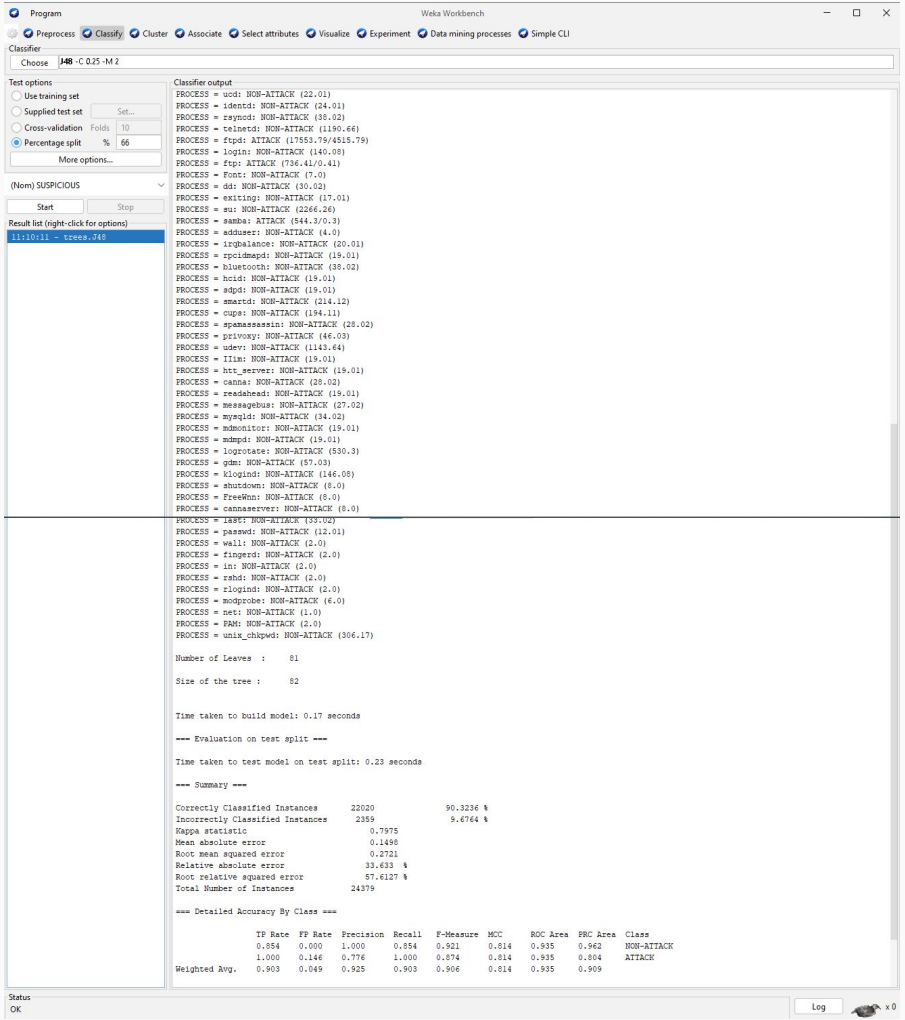
Results

Deep learning multilayer perceptron



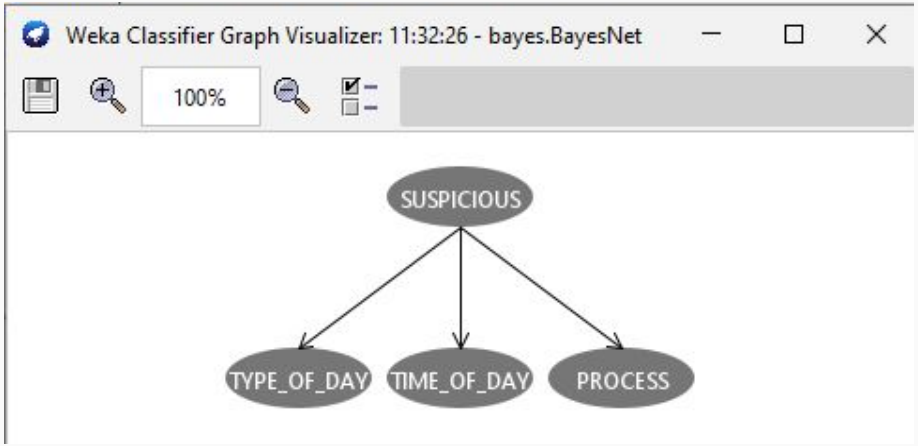
Results

Decision Tree - J48



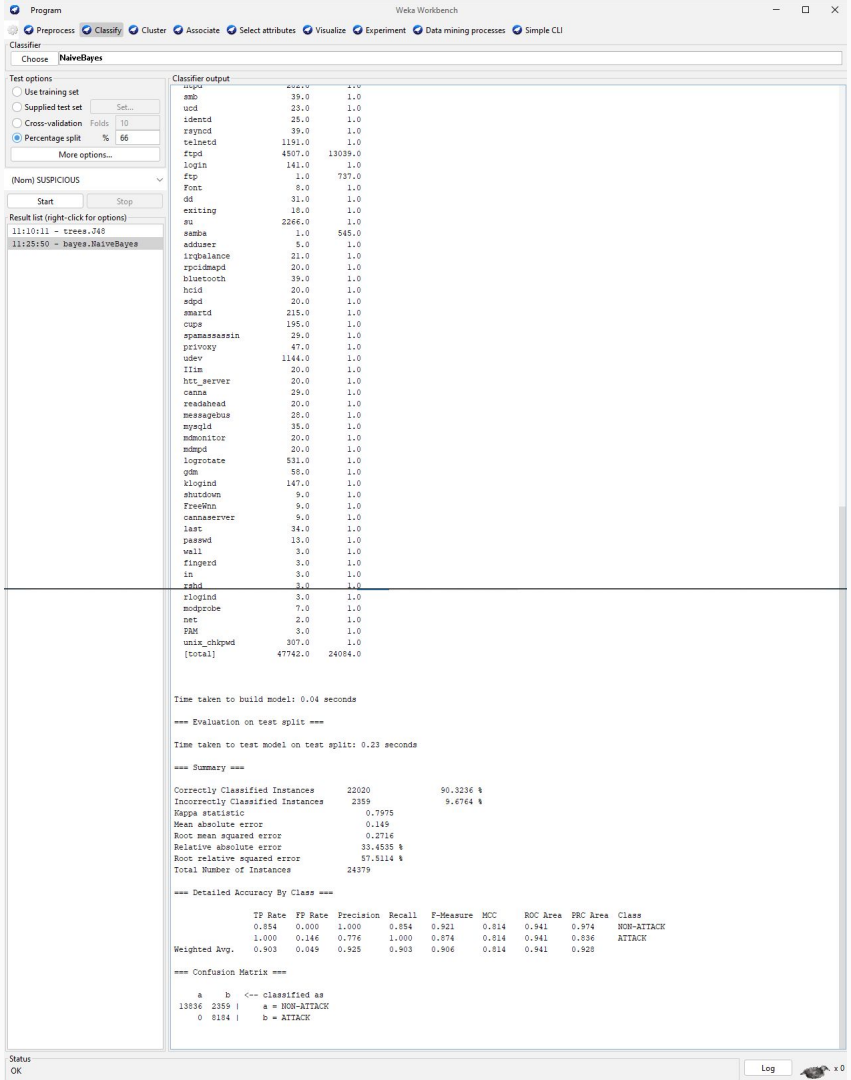
Results

Graph



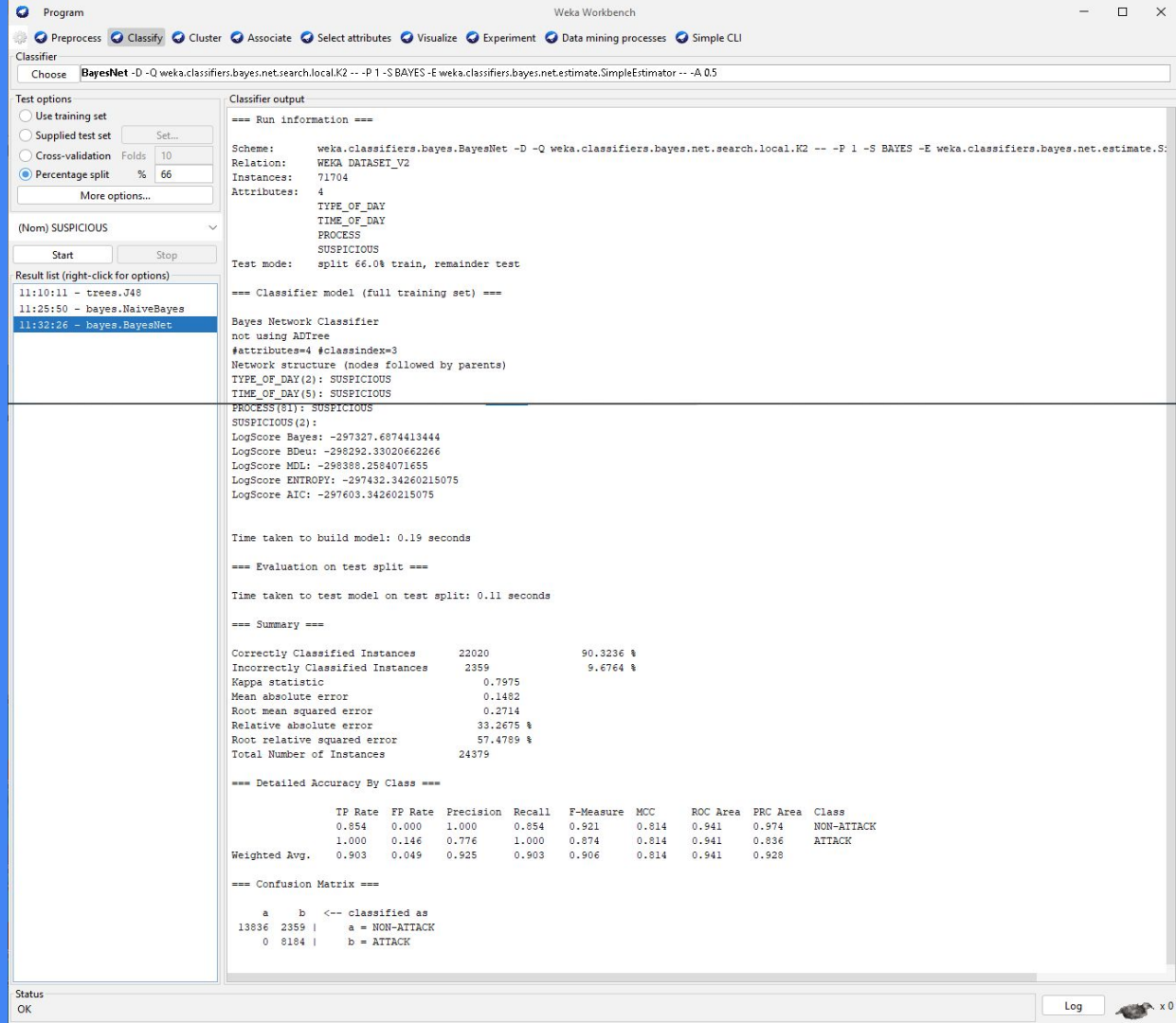
Results

Naive Bayes results



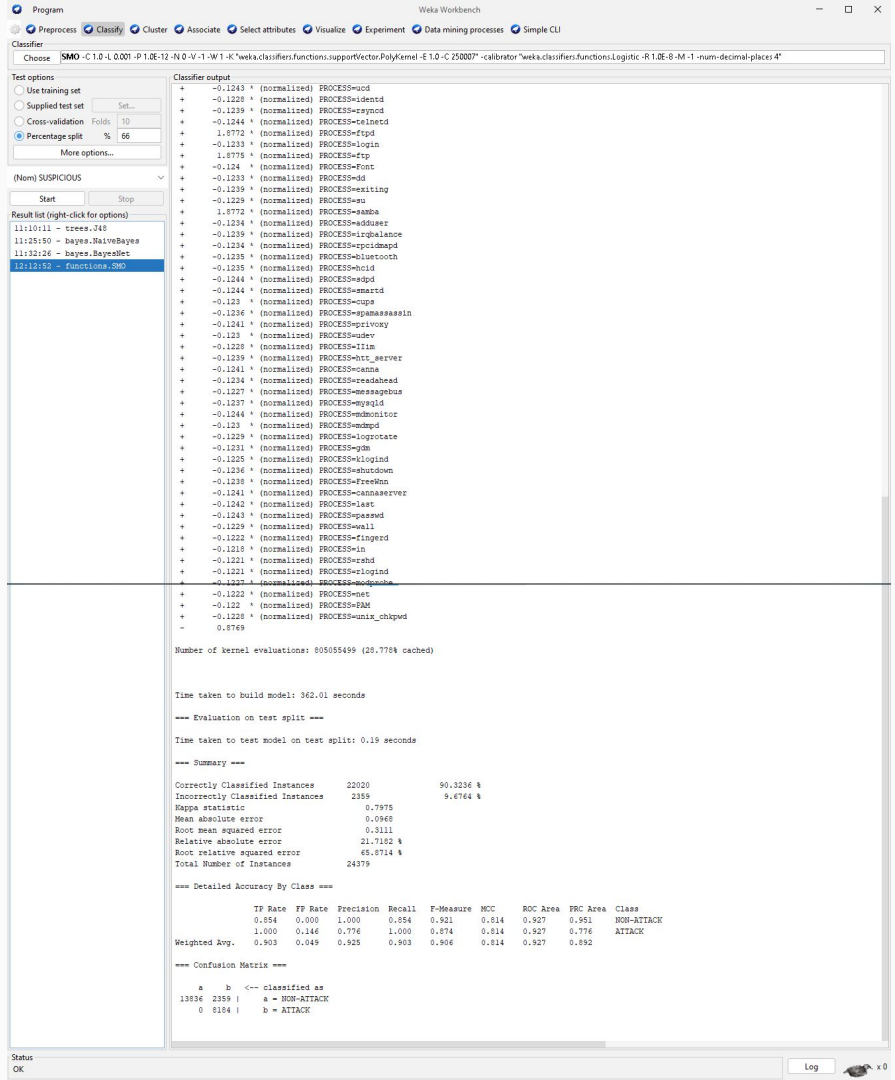
Results

Bayes Net results



Results

SMO Results



Results

Deep learning - Multilayer Perceptron results