Microsoft Cybersecurity Incident Prediction Final Report for DSCI 632

David W. Blankenship Jr.

dwb65@drexel.edu

Table of Contents

Introduction

Motivation

The motivation for this project is the increasing relevance of cybersecurity in the modern world. As we live in an increasingly digital world, we have seen high profile cybersecurity incidents such as the 2014 Sony Pictures hack or the 2021 Colonial Pipeline Ransomware attack and the continued push for new technologies such as the Internet of Things which increases the surface area of attack for the average person. These trends and events highlight the importance of research into cybersecurity incidents to better understand the techniques used by bad actors and develop new methods of defeating them. This paper is intended to add to the research trying to understand and identify cyber incidents to better prevent them.

Project Structure

This paper is written for my graduate class Drexel 632 Applied Cloud Computing. The purpose of this paper is to show mastery of the skills we've developed throughout the class. As such this project is largely done in PySpark to showcase the technical ability we've developed. The major focus is on Exploratory Data Analysis (EDA) and Machine Learning (ML) components.

Infrastructure

For this project I used Google Colab to conduct my EDA and ML portions. The Notebooks for the project can be found at my GitHub page linked here: https://github.com/General-Cow/Cybersecurity-Incident-Prediction. Please be aware that the formatting in the notebooks is off whenever a .show() command is used due to the difference in how Colab and GitHub display the output.

Dataset Background

Dataset Acquisition

This dataset is publicly available on Kaggle at the following reference [1]. Additionally, the dataset is associated with a research paper [2] that dives into a deeper description of the dataset, potential use cases, and an analysis of the dataset by the original authors. This dataset was put together for Microsoft Copilot for Security Guided Response, an AI solution for cybersecurity. The dataset is called GUIDE and is meant to

provide a large public dataset to support the development of guided response systems by other groups.

Dataset Basic Characteristics

The dataset comes presplit by the authors into a train and test dataset. The split is a 70:30 split and is stratified by the authors according to stratified based on IncidentGrade (our target), OrgId, and DetectorId. The train dataset is 2.3 GB and has 9516837 rows of data and 45 columns: 44 features and 1 target. The test dataset is 1 GB and has 4147992 rows of data and 46 columns. The extra column in the test dataset is a Usage column and is dropped later as it does not exist in the train dataset. The table describing the features of the dataset from the original paper [2] is recreated below without alteration.

It should be noted that the authors did perform some techniques that altered the original dataset as part of an anonymization process to ensure that privacy is protected. These techniques are described in the original paper [2] and consist of introducing noise into the timestamps and some ID features. These techniques are justifiable based on privacy concerns and shouldn't overly affect the results.

The data required very little in terms of preprocessing thanks to this being a well-maintained dataset posted by Microsoft researchers on Kaggle.

Feature	Description
Id	Unique ID for each OrgId-IncidentId pair
OrgId	Organization identifier
IncidentId	Organizationally unique incident identifier
AlertId	Unique identifier for an alert
Timestamp	Time the alert was created
DetectorId	Unique ID for the alert generating detector
AlertTitle	Title of the alert
Category	Category of the alert
MitreTechniques	MITRE ATT&CK techniques involved in alert
IncidentGrade	SOC grade assigned to the incident
ActionGrouped	SOC alert remediation action (high level)
ActionGranular	SOC alert remediation action (fine-grain)
EntityType	Type of entity involved in the alert
EvidenceRole	Role of the evidence in the investigation
Roles	Additional metadata on evidence role in alert
DeviceId	Unique identifier for the device
DeviceName	Name of the device
Sha256	SHA-256 hash of the file
IpAddress	IP address involved
Url	URL involved
AccountSid	On-premises account identifier
AccountUpn	Email account identifier
AccountObjectId	Entra ID account identifier
AccountName	Name of the on-premises account
NetworkMessageId	Org-level identifier for email message
EmailClusterId	Unique identifier for the email cluster
RegistryKey	Registry key involved
RegistryValueName	Name of the registry value
RegistryValueData	Data of the registry value
ApplicationId	Unique identifier for the application
ApplicationName	Name of the application
OAuthApplicationId	OAuth application identifier
ThreatFamily	Malware family associated with a file
FileName	Name of the file
FolderPath	Path of the file folder
ResourceIdName	Name of the Azure resource
ResourceType	Type of Azure resource
OSFamily	Family of the operating system
OSVersion	Version of the operating system
AntispamDirection	Direction of the antispam filter
SuspicionLevel	Level of suspicion
LastVerdict	Final verdict of threat analysis
CountryCode	Country code evidence appears in
State	State of evidence appears in
City	City evidence appears in

Descriptions of columns from original paper [2]

Exploratory Data Analysis

Missing Data Analysis

I first look at which columns have missing values and find that MitreTechniques, ActionGrouped, ActionGranular, EmailClusterId, ThreatFamily, ResourceType, Roles, AntispamDirection, SuspicionLevel, LastVerdict, and our target IncidentGrade are all missing data. The raw missing count and the percentage of the total missing data are given in the table below.

Column Name	Missing Count	Missing Percent
MitreTechniques	5468386	57.46
ActionGrouped	9460773	99.41
ActionGranular	9460773	99.41
EmailClusterId	9420025	98.98
ThreatFamily	9441956	99.21
ResourceType	9509762	99.93
Roles	9298686	97.71
AntispamDirection	9339535	98.14
SuspicionLevel	8072708	84.83
LastVerdict	7282572	76.52
IncidentGrade	51340	0.54

I chose to drop the MitreTechniques, ActionGrouped, ActionGranular, EmailClusterId, ThreatFamily, ResourceType, Roles, AntispamDirection, SuspicionLevel, and LastVerdict due to the sheer amount missing. While MitreTechniques has significantly less missing than the rest, it is still a significant majority of the data. We also elect to dive deeper into analysis of the data we do have for this column later. For our target column IncidentGrade, due to the sheer volume of data we already have, we elect to simply drop the null values rather than attempt to impute the target values using something like a KNN imputer. It also avoids the possibility of inserting bias or data leakage.

Univariate Analysis

In this section I perform univariate analysis and look at the breakdown of the categorical and numerical features. For the categoricals we look at the counts of each feature. For the numericals, we look the the count, max, min, mean, standard deviation, and quartile ranges to characterize the distribution.

Categorical Features

Category	Count
InitialAccess	4293041
Exfiltration	1577965
SuspiciousActivity	1003933
CommandAndControl	826691
Impact	750885
CredentialAccess	300518
Execution	267594
Malware	144081
Discovery	129342
Persistence	72682
DefenseEvasion	46662
LateralMovement	41754
Ransomware	18974
UnwantedSoftware	18211
Collection	14753
PrivilegeEscalation	4671
Exploit	4648
CredentialStealing	388
WebExploit	38
Weaponization	6

Initial access is by far the most common label in the dataset. The following four are also sizable. Following labels rapidly fall off in count suggesting the dominance of the top 5.

IncidentGrade	Counts	Total %
BenignPositive	4110817	43.43
TruePositive	3322713	35.10
FalsePositive	2031967	21.47
NULL	51340	0.54

IncidentGrade is our target and as such it is going to be very important to our evaluation to account for the observed imbalance of labels. We will be looking at the F1 score, precision, and recall by each class because of this. From the sources we know that the data was stratified on IncidentGrade [1][2].

ActionGrouped	
NULL	9460773
ContainAccount	53760
IsolateDevice	2237
Stop Virtual Machines	67

Overwhelmingly null, but ContainAccount being the most common action taken by the SOC tracks with the most common MITRE Technique being related to attacks from compromised accounts. However, the amount of nulls makes it impossible to suggest that this is a clear trend for ActionGrouped within the dataset.

ActionGranular	
NULL	9460773
update stsrefreshtokenvalidfrom timestamp.	21393
account password changed	14059
change user password.	13623
isolateresponse	2043
account disabled	1991
disable account.	1143
reset user password.	886
forcepasswordresetremediation	234
quarantinefile	194
msecidentitiesconfirmusercompromised	146
disableuser	107
msecidentitiessuspenduser	81
delete virtualmachines	67
account deleted	55
delete user.	35
set force change user password.	7

We see that ActionGranular is paired with ActionGrouped. Can also see the breakdown by type of action

EntityType	
Ip	2181194
User	1932416
MailMessage	1173154
Machine	699208
File	688402
Url	682578
CloudLogonRequest	638565
Mailbox	483158
Process	345732
MailCluster	224684
CloudApplication	216811
CloudLogonSession	212382
RegistryValue	11209
AzureResource	8166
RegistryKey	7277
GenericEntity	4294
OAuthApplication	2595
Malware	2580
SecurityGroup	1518
BlobContainer	306
Blob	263
MailboxConfiguration	226
Nic	49
IoTDevice	31
ActiveDirectoryDomain	11
GoogleCloudResource	10
AmazonResource	6
Container	2
KubernetesCluster	2
ContainerImage	2
KubernetesPod	2
KubernetesNamespace	2
ContainerRegistry	2

We see several relevant EntityType labels related to common MITRE techniques below. Particularly those related to valid accounts and cloud accounts.

EvidenceRole	
Related	5208644
Impacted	4308193

We see a slight imbalance toward the evidence being related to and not impacted by the incident.

Roles	
NULL	9298686
Contextual	130528
Destination	34324
Suspicious	28952
Attacker	13096
Source	7171
Compromised	3219
PolicyViolator	666
Attacked	188
Edited	6
Added	1

Unfortunately, Roles is mostly null values which is very unfortunate as it may have been able to provide additional information for the evidence role and would have potentially been a good choice to pair with EvidenceRole for additional feature engineering.

Top 10 ThreatFamily	
NULL	9441956
Phish	3526
Malgent	2636
Donoff	2519
CustomEnterpriseBlock	2145
CustomEnterpriseBlockOnly	2139
CymRan	1648
CustomCertEnterpriseBlock	1595
Phonzy	1493
Wacatac	1185

The overwhelming presence of nulls makes this impossible to comment on.

ResourceType	
NULL	9509762
Virtual Machine	4146
Key Vault	480
App Service	404
Azure Arc machine	373
Storage Account	366
Subscription	271
SQL Database	226
Azure Database for MySQL Server	182
Azure Resource	157
Virtual Machine Scale Set	112
API Management Service	94
Key Vault Secret	77
Azure Database for PostgreSQL Server	45
SQL Server	41
Azure Cosmos DB Account	30
Networking	14
Synapse SQL Pool	11
GCP Compute Instance	10
Kubernetes Service	9
SQL Managed Instance	9
Key Vault Certificate	7
AWS EC2 Instance	6
Key Vault Key	2
Synapse Workspace	2
Cognitive Service	1

The overwhelming presence of nulls makes this impossible to comment on.

OSFamily		Top 10 OSVersion	
0	190036	**66**	9322572
1	2732	**0**	187405
2	1496	**2**	1892
3	7	**1**	1652
4	1	**3**	1125
		4	732
		6	362
		5	266
		8	132
		9	109

For both OSFamily and OSVersion we see that one variable dominates. It is hard to tell if this reflects a sampling bias in the data or is actually representative of preferred targets. It is not stated within the paper or Kaggle dataset if there is a temporal meaning to the values.

AntispamDirection	
NULL	9339535
Inbound	161111
Intraorg	15308
Outbound	868
DomainPII_50d8b4a941c26b89482c94ab324b5a274f9ced66	10
DomainPII_df80ab894e01e375bf55d12ba315c04029d3e32d	5

The overwhelming presence of nulls makes this impossible to comment on.

SuspicionLevel	
Null	8072708
Suspicious	1442614
Incriminated	1515

While dominated by Nulls we see a tendency towards Suspicious for the data with this feature as well as a relatively sizable amount that have this label.

LastVerdict	
NULL	7282572
Suspicious	1402997
Malicious	433359
NoThreatsFound	397718
DomainPII_50d8b4a941c26b89482c94ab324b5a274f9ced66	128
DomainPII_9207384283ce115db5a590dd9ca5de21e5e99df2	63

Similarly to SuspicionLevel we see mostly nulls with the top 3 labels being relatively sizable in absolute terms and Suspicious again dominating.

Numerical Features

Rather than comment on each table, I will simply note that we see two trends within the numerical data. First, we see a sizable amount of our numerical features are heavily skewed. It is very common for our features to have a minimum of 0 and a remainder of all the quartiles and max be the exact same value. For those that don't exactly fit the previous trend we still see heavily skewed data.

	ID	OrgId	IncidentId
Count	9516837	9516837	9516837
Mean	8.43E11	181.58	70663.49
Standard Deviation	4.96E11	386.78	120836.85
Minimum	0	0	0
25%	412316863917	10	504
50%	841813590289	45	10335
75%	1271310322153	171	84315
Max	1709396988938	6147	599706

	AlertId	DetectorId	AlertTitle
Count	9516837	9516837	9516837
Mean	406518.83	110.6724262483428	2947.32
Standard Deviation	459282.70	435.1037900929531	11461.50
Minimum	0	0	0
25%	23232	2	2
50%	216619	9	11
75%	671536	45	180
Max	1721456	9522	113174

	DeviceId	Sha256	IpAddress
Count	9516837	9516837	9516837
Mean	95664.76	128719.06	285750.61
Standard Deviation	16352.88	33992.08	141224.00
Minimum	0	0	0
25%	98799	138268	360606
50%	98799	138268	360606
75%	98799	138268	360606
Max	98799	138268	360606

	Url	AccountSid	AccountUpn
Count	9516837	9516837	9516837
Mean	150331.69	352446.57	464377.26
Standard Deviation	37507.95	166496.50	290227.49
Minimum	0	0	0
25%	160396	441377	92558
50%	160396	441377	673934
75%	160396	441377	673934
Max	160396	441377	673934

	AccountObjectId	AccountName	DeviceName
Count	9516837	9516837	9516837
Mean	340962.24	356966.41	143229.83
Standard Deviation	159937.73	174446.56	36070.56
Minimum	0	0	0
25%	425863	453297	153085
50%	425863	453297	153085
75%	425863	453297	153085
Max	425863	453297	153085

	NetworkMessageId	RegistryKey	RegistryValueName
Count	9516837	9516837	9516837
Mean	480046.69	1628.17	634.73
Standard Deviation	141758.38	66.75	12.58
Minimum	0	0	0
25%	529644	1631	635
50%	529644	1631	635
75%	529644	1631	635
Max	529644	1631	635

	RegistryValueData	ApplicationId	EmailClusterId
Count	9516837	9516837	96812
Mean	859.56	2200.91	3.24E9
Standard Deviation	18.86	331.55	1.03E9
Minimum	0	0	192708.0
25%	860	2251	2.76E9
50%	860	2251	3.50E9
75%	860	2251	4.11E9
Max	860	2251	4.29E9

	ApplicationName	OAuthApplicationId	FileName
Count	9516837	9516837	9516837
Mean	3342.79	880.80	262262.09
Standard Deviation	510.34	12.91	81529.56
Minimum	0	0	0
25%	3421	881	289573
50%	3421	881	289573
75%	3421	881	289573
Max	3421	881	289573

	FolderPath	ResourceIdName
Count	9516837	9516837
Mean	107617.15	3583.48
Standard Deviation	32208.35	90.20
Minimum	0	0
25%	117668	3586
50%	117668	3586
75%	117668	3586
Max	117668	3586

	CountryCode	State	City
Count	9516837	9516837	9516837
Mean	223.67	1351.49	9936.18
Standard Deviation	62.80	350.98	2606.81
Minimum	0	0	0
25%	242	1445	10630
50%	242	1445	10630
75%	242	1445	10630
Max	242	1445	10630

Time Feature Analysis

Each row of data has an associated timestamp for the incident in question. As there may be a time component relevant to identifying cybersecurity incidents, I elect to perform feature engineering on the Timestamp column and break the original timestamp column into 8 columns: Year, Month, Day, Hour, Minute, Second, Day of Week, and Week of Year. We analyze the variables as well.

The data is overwhelmingly from 2024 with all but 179 counts of data from 2023. All other entries, over 9.5 million of them, are from 2024 with no missing values. The month of the data is overwhelmingly from June with May being the only other month of significance. For the week of the year, we see a band of weeks between weeks 21-25 with high volumes of activity. Week 21 in 2024 started on May 20th and week 25 ended on June 23rd. Events in this timespan account for over 99.97% of the entire dataset. This could either be extreme selection bias in the data or reflective of a serious cyber-attack campaign. We also looked at the events by hours. While there is some variation there is no clear trend, and the hours are nowhere near as biased as the previous features.

A search for 2024 cybersecurity incidents did not reveal any particular incident or campaign that may correlate with the week 21-25 activity. I also reviewed the CrowdStrike 2025 Global Threat Report [3] which highlights major cybersecurity threat trends of the previous year and predictions for the coming year. While a very informative read, there was no clearly correlated incident. According to the report Microsoft did disclose 5 vulnerabilities related to CVE CVE-2023-29360 on 11 June 2024 [3]. However, reviewing its entry on NIST [4], it is not clearly related to anything in this dataset.

MITRE Techniques Analysis

MITRE Techniques come from the MITRE ATT&CK knowledge base. The MITRE techniques are specific adversarial techniques systematically categorized to allow for the better development and understanding of cybersecurity practices. The knowledge base can be found at https://attack.mitre.org/ [5]. While we will be dropping the column due to the volume of missing values, inspecting it will be revealing as to the nature of the most common types of attacks seen in the train dataset. There are 1194 unique values and exploring them all is beyond the scope of this paper. However, we show the top 10 most common non-null values below and we will see that this covers a large fraction of our total non-null incidents.

MITRE Techniques	Counts	Benign Positive	False Positive	True Positive	Null
T1078 T1078.004	1354904	35897	308347	1009935	725
T1566.002	814308	391993	128955	292731	629
T1566	659591	346566	167763	141964	3298
T1133	145579	19124	126455	0	0
T1566.001	136892	90698	7398	38792	4
T1110 T1110.003 T1110.001	88661	1549	3102	84002	8
T1087 T1087.002	54564	23935	21831	8782	16
T1110	41768	24047	7081	10604	36
T1078 T1098	40342	35268	3721	1341	12
T1559 T1106 T1059.005	37869	37869	0	0	0

We can also look at the contingency table comparing the top 10 MITRE techniques to the Incident Grade and we see several interesting patterns. We will break down these down by technique and explain the meaning of them. Note that the nulls are a small fraction of all the different techniques we look at here.

- T1078; T1078.004
    - T1078 is an attack that uses a valid account or compromised credentials.
    - T1078.004 is a sub-technique related to cloud accounts which is very relevant to this course.
    - These types of attacks tend to be True Positives in this dataset
- T1087; T1087.002
    - T1087 is an attack that involves Account Discovery or attempting to get a list of valid accounts, emails, or other credentials.
    - T1087.002 is a sub-technique involving getting a list of Domain Accounts
    - These tend to mostly be Benign and False Positives
- T1566
    - T1566 is a Phishing attack which is when an adversary sends messages to gain access to a system.
    - Mostly Benign Positives but a significant fraction are False and True Positives

- T1110; T1110.003; T1110.001
    - T1110 is a Brute Force technique to gain access to accounts.
    - T1110.001 is a sub-technique of brute forcing involving password guessing
    - T1110.003 is a sub-technique of brute forcing involving password spraying, which is using a list of commonly used passwords.
    - Significant balance towards True Positives
- T1133
    - T1133 is related to External Remote Services such as VPN, Citrix, and similar applications.
    - 0 True Positives and mostly False Positives
- T1559; T1106; T1059.005
    - T1559 is related to Inter-Process Communication mechanisms for local code or command execution
    - T1059.005 is related to abuse of Command and Scripting Interpreter: Visual Basic. VB is a commonly used Microsoft product
    - T1106 is related to adversaries exploiting the Native API.
    - Only Benign Positives
- T1566.001
    - T1566.001 is a sub-technique of phishing known as spear phishing with an attachment that is targeted to a specific individual.
    - Mostly Benign Positives with a sizable fraction of True Positives
- T1566.002
    - T1566.002 is a sub-technique of phishing known as spear phishing with a link that is targeted to a specific individual.
    - Sizable distribution across all values but mostly tending toward Benign Positives
- T1078; T1098
    - T1078 is an attack that uses a valid account or compromised credentials.
    - T1098 is related to Account Manipulation. This can involve modifying permissions or credentials of compromised accounts.
    - Mostly Benign Positives
- T1110
    - T1110 is a Brute Force technique to gain access to accounts.
    - Relatively sizable across all, but mostly Benign Positives

It should be recalled that the Incident Grade classes are imbalanced with majority Benign Positives, followed by True Positives, and False Positives as the least common. Some of these techniques are distributed in a similar manner to the imbalance and as such

it can be hard to disentangle real trends from distributions of the target. As such they should be considered with the difference to the overall dataset's imbalance.

The most striking feature of these most common attack vectors is how many are related to having, or attempting to get access to, valid accounts/credentials. T1078 and T1078.004 are explicitly attacks involving compromised accounts. T1087 and T1087.002 are related to acquiring valid accounts to exploit. T1566, T1566.001, and T1566.002 all phishing techniques which are typically done to get access to a user's account or credentials. T1110, T1110.001, and T1110.003 all involve attempting to gain access to an account. T1098 is related to a situation where an account is compromised. These account for 8 of the top 10 most common techniques.

Returning to the CrowdStrike Global Threat Report [reference] there were some trends throughout the year that are interesting when considered with our dataset. First, there was a 26% increase in cloud intrusions of which 35% in the front half were able to achieve initial access using valid accounts. This is highly relevant to the trends we see with valid account related activities I just described and the most common MITRE technique category involving cloud accounts. The timing also corresponds with our weeks of highest activity discussed in the Time Analysis subsection. CrowdStrike also reports a 50% increase in the use of stolen credentials once again corroborating with what we see in this dataset. Finally, CrowdStrike reports that 52% of the vulnerabilities they observed involved initial access once again highlighting the trend we see in this dataset.

Model Development & Evaluation

The models I elected to use for this project are Naïve Bayes, Logistic Regression, Decision Tree and Random Forest Classifiers. Initially I had hoped to use the Gradient Boosted Tree Classifier as gradient boosted trees have very high performance for similar problems, however the PySpark implementation is incapable of handling multiclass problems. It can only handle binary classification problems.

Another major issue I ran into was related to Google Colab's free user limits constantly crashing the runs. This appeared to be related to going over the allotted RAM and showed up consistently when attempting Grid Search Cross-Validation. As such I elected to conduct manual tuning to particularly great effect for the Decision Tree models. The selected parameters are given in the Model Parameters subsection.

Model Choices

Naïve Bayes

As a baseline model we elect to use Multinomial Naïve Bayes and default parameters. As we will see this is probably our weakest model and I suspect it has to do with the assumption of feature independence inherent in Naïve Bayes. Cybersecurity data tends to have highly correlated features, and this dataset is no exception. Many of the features explicitly mention their relationship with other features. For example, Role's description mentions its association with EvidenceRole. There's also ID which mentions its related to OrgId and IncidentId pairs. However, its purpose is to serve as an easily implemented baseline with which to compare our other models against.

Logistic Regression

We also elect to use Logistic Regression as a relatively simple model to implement with relatively fast training time. We use SoftMax regression to determine classes. One-vs-Rest was also attempted but constantly failed and so was discarded. We used default parameters.

Decision Tree

Decision Trees are an excellent choice of model for this dataset. They require relatively little data preparation, make few assumptions of the data, work with a mix of feature types, and are excellent at finding complex patterns. Additionally, they are well suited to data as heavily skewed as ours.

In general, the biggest downsides to Decision Trees are that they can be computationally expensive for training and prone to overfitting. We will see however that we did not actually run into these issues. The biggest issue we ran into was that in PySpark there is a limit to the max depth of the tree. As we will see the max depth of 30 ended up being what we selected through hyperparameter tuning.

Random Forest

One run of Random Forest was attempted to leverage the powerful predictive power of ensemble methods, however practical concerns plagued this model with several crashes of the Colab notebook. As such a decision was made to reduce the number of trees. Like Decision Tree, PySpark limited the depth to 30.

Feature Engineering & Pipeline Development

All Models

- We drop the following columns with high percentages of nulls:
    - MitreTechniques, ActionGrouped, ActionGranular, EmailClusterId, ThreatFamily, ResourceType, Roles, AntispamDirection, SuspicionLevel, LastVerdict
    - We also drop the Usage column in the test dataset.
- We drop target nulls in the train set as they are a very small portion of our dataset and do not consider it wise to impute the target. The test set has no nulls in the target.
- We develop a custom timestamp transformer to convert our timestamp column into the following new features:
    - Year, Month, Day, Hour, Minute, Second, Day of Week, and Week of Year
- For the Category, EntityType and EvidenceRole features we use the StringIndexer function to convert the strings column of labels into a ML column of labels that can be used in both models.
    - We also use the string indexer on the target IncidentGrade to transform it into labels that can be predicted against.
- We use the VectorAssembler function to transform our data into a vector column that is convenient for the PySpark models.
- We create a Pipeline that allows us to easily stage our full dataset and transformations in a manner that is easy to prepare our data for model training and fit
    - The individual components will be shown for each model as they differ from model to model.

Naïve Bayes

- No feature scaling required for Naïve Bayes
- For the Naïve Bayes model, we also use the OneHotEncoder function to encode these StringIndexer transformed Category, EntityType and EvidenceRole features.

Logistic Regression

- Due to the high tendency of skewed data within this dataset, I elected to use the MinMaxScaler for all features.
    - StandardScaler assumes a normal distribution in our dataset. This is decidedly not the case.
    - We also tried RobustScaler but found worse results.

- For the Logistic Regression model, we also use the OneHotEncoder function to encode these StringIndexer transformed Category, EntityType and EvidenceRole features.

Decision Tree Classifier

- No feature scaling required due to the nature of decision trees.
- Otherwise, the same as mentioned in the All subsection.

Random Forest Classifier

- No feature scaling required due to the nature of random forests.
- Otherwise, the same as mentioned in the All subsection.

Model Parameters

- Naïve Bayes
    - We use default parameters. This includes the choice for Multinomial Naïve Bayes vice other options
- Logistic Regression
    - We use default Parameters
- Decision Tree Classifier
    - I used maxBins of 33. This was required as the maxBins must be at least as large as the largest categorical feature.
    - I tuned the maxDepth with the following values [5, 10, 15, 30]
        - The best value was found to be 30
    - I tuned minInstancesPerNode with the following values [1, 500, 1000]
        - The best value was found to be 500
- Random Forest Classifier
    - Reduced from default to 5 numTrees used to prevent crashing the notebook.
    - maxDepth of 30
    - minInstancesPerNode os 500
    - maxBins of 33. This was required as the maxBins must be at least as large as the largest categorical feature.

Results

For our evaluation we use the F1 scores of the classes, the Macro-F1 score, and related measures of the precision and recall. The Macro-F1 score is the average of the different classes F1 scores. These are the recommended metrics from the original authors [reference]. Additionally, we provide the Confusion Matrix for the top model as given by the crosstab function in PySpark.

Our top model as seen in the tables below was the Decision Tree. We used the Macro F1 score to determine the best model amongst different runs. It should be noted that amongst the various Decision Tree runs there was no model with clearly better results in every category. In general, it seems that there is some sacrifice in model performance for certain categories to increase the general performance. For example, in our first Decision Tree run we obtained a precision of 0.96 for TruePositive and a recall of 0.95 for BenignPositive. But this is at the expense of TruePositive and FalsePositive recalls which are 0.56 and 0.35 and the precision for BenignPositive which is 0.59. We see below in the top Decision Tree model significantly better generalized performance. The results of the Random Forest model suggest it they may be a powerful tool if the number of trees can be increased and run in an environment less prone to crashing than Colab.

If our goal is to maximize detections precision or recall of certain classes, then I would consider recommending different models. We see this for example in medical research where recall is very important due to the severity of false negatives for the health of the patient. Considering the cost of cybersecurity incidents, we may have something similar here. However, false positives likewise have a cost associated with them making this a hard problem requiring a serious risk analysis likely tailored to a specific use case that is well beyond the scope of this paper. As such we choose to prefer the generalized models.

Naïve Bayes Test	**Precision**	**Recall**	**F1**	**Macro-F1**
Benign Positive	0.4898	0.5061	0.4978	0.4697
False Positive	0.2831	0.3803	0.3246	
True Positive	0.6103	0.4597	0.5244	

Logistic Regression Test	**Precision**	**Recall**	**F1**	**Macro-F1**
Benign Positive	0.6051	0.8393	0.7032	0.6276
False Positive	0.6469	0.2562	0.3670	
True Positive	0.7306	0.6654	0.6964	

Decision Tree Test	Precision	Recall	F1	Macro-F1
Benign Positive	**0.8589**	0.8792	**0.8690**	**0.8592**
False Positive	0.8128	**0.7951**	**0.8038**	
True Positive	0.8878	**0.8748**	**0.8813**	

Random Forest Test	Precision	Recall	F1	Macro-F1
Benign Positive	0.7859	**0.9168**	0.8463	0.8389
False Positive	**0.8459**	0.7135	0.7741	
True Positive	**0.9182**	0.8254	0.8693	

The highest value of each category is bold. Macro F1 considers the full test set.

Decision Tree Test Confusion Matrix	Benign Positive Prediction	False Positive Prediction	True Positive Prediction
Benign Positive Target	1541268	113829	97843
False Positive Target	117836	717699	67163
True Positive Target	135348	51448	1305558

Conclusion & Future Scope

Conclusion

We have displayed the ability of this dataset to create a model that is able to accurately predict the Incident Grade of a cybersecurity threat. While there is still work to be done to continue to improve detection the results are very promising particularly if more developed and high-performance machine learning models and libraries are used. Freitas et al [2] suggest that a greater than 99% confidence threshold must be passed to be confident in automated systems to make decisions based on these predictions due to the severity of consequences of bad predictions. While this analysis does not achieve this it is a step in the right direction. This project was also able to observe and corroborate important trends in cybersecurity incidents within the year 2024. We saw this in the MITRE

Technique analysis when we compared it with the CrowdStrike 2025 Global Threat Report [3].

Personally, I am very satisfied with the results of this project. Within the scope of the class I was able to develop and display my skills with PySpark on a very professionally interesting and relevant project while achieving respectable results.

Future Scope

While I am very satisfied with the results of this analysis there are likely several ways the results could be even further improved.

First, this project was done for DSCI632 Applied Cloud Computing and the requirement for the class was that all work be done in the PySpark library to prove mastery of cloud computing techniques we developed throughout the class. Typically, I would have tended towards the use of Scikit-Learn for the well-developed model library and XGBoost for efficient and high performing gradient boosted tree model. I also suspect a Neural Network model in PyTorch or Keras for deep learning methods would be a very interesting project. Trying these other libraries could potentially improve upon the respectable results we've obtained in PySpark.

Second, we saw that the MITRE Techniques include a wealth of information to analyze and could potentially offer the same to the analysis. One way to incorporate the MITRE techniques into the dataset would be to break up the entries and turn them into features with binary encoded features. For example, the T1078; T1078.004 entries would be broken into two columns with 1's in the two columns corresponding to these two MITRE techniques and 0s in all others. One concern is that this may cause dominance of these features as there 1194 unique labels for MitreTechniques and breaking it up further would cause an explosion of features. It would still be relatively small compared to the size of the dataset. Alternatively, feature engineering could be done to group the different techniques into even broader categories, helping to minimize the feature explosion while still capturing relevant information. This would require a deeper literature review and manual effort to create these features unfortunately beyond the current scope of this project.

I would also like a deeper investigation into why the cybersecurity incidents were so overwhelmingly concentrated into a narrow band of weeks as we saw in the Time Analysis. This would almost certainly require a much deeper search or communication with the original authors or other cybersecurity experts that is beyond the scope of this report.

References

1. : Scott Freitas, Jovan Kalajdjieski, Amir Gharib, and Rob McCann. (2024). Microsoft Security Incident Prediction. Kaggle. https://doi.org/10.34740/KAGGLE/DSV/8929038
2. Scott Freitas, Jovan Kalajdjieski, Amir Gharib and Rob McCann. "AI-Driven Guided Response for Security Operation Centers with Microsoft Copilot for Security." arXiv preprint arXiv:2407.09017 (2024).
3. CrowdStrike 2025 Global Threat Report. (2025). https://www.crowdstrike.com/en-us/. Accessed 06 March 2025.
4. NIST National Vulnerability Database. https://nvd.nist.gov/vuln/detail/CVE-2023-29360. Accessed 06 March 2025.
5. MITRE Corporation. “MITRE ATT&CK.” https://attack.mitre.org/. Accessed 06 March 2025.