Domain Knowledge Aided Explainable Artificial Intelligence for Intrusion Detection and Response

Sheikh Rabiul Islam, William Eberle, Sheikh K. Ghafoor, Ambareen Siraj, and Mike Rogers
Departmemt of Computer Science
Tennessee Technological University
Cookeville, U.S.
sislam42@students.tntech.edu, {weberle, sghafoor, asiraj, mrogers}@tntech.edu

Abstract

Artificial Intelligence (AI) has become an integral part of modern-day security solutions for its capability of learning very complex functions and handling ”Big Data”. However, the lack of explainability and interpretability of successful AI models is a key stumbling block when trust in a model’s prediction is critical. This leads to human intervention, which in turn results in a delayed response or decision. While there have been major advancements in the speed and performance of AI-based intrusion *detection* systems, the *response* is still at human speed when it comes to explaining and interpreting a specific prediction or decision. In this work, we infuse popular domain knowledge (i.e., CIA principles) in our model for better explainability and validate the approach on a network intrusion detection test case. Our experimental results suggest that the infusion of domain knowledge provides better explainability as well as a faster decision or response. In addition, the infused domain knowledge generalizes the model to work well with unknown attacks, as well as open the path to adapt to a large stream of network traffic from numerous IoT devices.

1 Introduction

Most of the recent advancements in Artificial Intelligence (AI), and more specifically Machine Learning (ML), have come from complex non-linear models such as Deep Neural Networks, Ensemble Methods, and Support Vector Machines. These models are also known as ”black box” models as they are complex to interpret and explain, which arises from their inherent non-linear capabilities, multiple parameters, and very complex transformations. In addition, some algorithms require a very large number of samples (i.e., large training sets) to work efficiently, where it is very difficult to figure out what the model learned from the dataset and which portion of the data set has more influence on the output (mle).

Due to these challenges, the black box models lacks explainability and interpretability, ultimately resulting in a lack of trust in the model and prediction, as well as possibly leading to a delayed human response/decision. This limitation also involves ethical issues in a few sensitive domains like finance (e.g., credit approval), health care (e.g., disease diagnosis), and security (e.g., identifying target). For instance, AI and ML are becoming an integral part of security solutions and defense. To mitigate the unethical use of AI and eliminate biases from AI systems, the various governments have started taking different precautionary initiatives. Recently, the European Union implemented the rule of ”right of explanation”, where a user can ask an explanation of algorithmic decision (Goodman and Flaxman 2017). In addition, more recently the US government introduced a new bill, the ”Algorithmic Accountability Act”, which would require companies to assess their machine learning systems for bias and discrimination, with a need to take corrective measures (Wyden). The U.S. Department of Defense (DoD) has identified explainability as a key stumbling block in the adoption of AI-based solutions in many of their projects. Their DARPA division has invested $2 billion on an Explainable Artificial Intelligence (XAI) program (Turek ; Rankin).

Network intrusions are a common cyber-crime activity, estimated to cost around $6 trillion annually in damages by 2021 (Doyle). In order to combat these attacks, an Intrusion Detection System (IDS) is a security system to monitor network and computer systems (Hodo et al. 2016). Research in AI-based IDS has shown promising results (Hodo et al. 2016),(Shone et al. 2018),(Kim et al. 2016),(Javaid et al. 2016),(Li, Sun, and Wang 2012), and has become an integral part of security solutions due to its capability of learning complex, nonlinear functions and analyzing large data streams from numerous connected devices. A recent survey by (Dong and Wang 2016) suggests that deep learning-based methods are accurate and robust to a wide range of attacks and sample sizes. However, there are concerns regarding the sustainability of current approaches (e.g., intrusion detection/prevention systems) when faced with the demands of modern networks and the increasing level of human interaction (Shone et al. 2018). In the age of IoT and Big Data, an increasing number of connected devices and associated streams of network traffic have exacerbated the problem. In addition, the delay in detection/response increases the chance of *zero day exploitation*, whereby a previously un-

Copyright © 2020 held by the author(s). In A. Martin, K. Hinkelmann, H.-G. Fill, A. Gerber, D. Lenat, R. Stolle, F. van Harmelen (Eds.), Proceedings of the AAAI 2020 Spring Symposium on Combining Machine Learning and Knowledge Engineering in Practice (AAAI-MAKE 2020). Stanford University, Palo Alto, California, USA, March 23-25, 2020. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

known vulnerability is just discovered by the attacker, and the attacker immediately initiates an attack. However, improved explainability of an AI model could quicken interpretation making it more feasible to accelerate the *response*.

Explainability is the extent to which the internal working mechanism of the machine or AI system can be explained in human terms. And interpretability is the extent to which a cause and effect (i.e., understanding of what's happening) can be observed within a system. In other words, interpretability is a form of abstract knowledge about what's happening and explainability is about the detailed step-by-step knowledge of what is happening (Montavon, Samek, and Müller 2018),(Turek). However, while some literature treat interpretability and explainability as the same, they are actually two different traits of a model. Just because a model can be interpreted does not mean that it can be explained, and explainability needs to go beyond the algorithm (Lipton 2016).

Explainability and interpretability of a model could be achieved before, during, and after modeling. From the literature, we find that interpretability in pre-modeling (i.e., before modeling) is under-focused. (Miller 2018) argue that explainability should incorporate knowledge from a different domain such as philosophy, psychology, and cognitive science so that the explanation is not just based on the researcher's intuition of what constitutes a good explanation. However, we also find that the use of domain knowledge for explainability is also under-focused. In this work, we introduce a novel approach for an AI-based explainable intrusion detection and response system, and demonstrate its effectiveness by infusing a popular network security principle (CIA principle) into the model for better explainability and interpretability of the decision.

We use a recent and comprehensive IDS dataset (CICIDS2017) which covers necessary criteria with common updated attacks such as DDoS, Brute Force, XSS, SQL Injection, Infiltration, Portscan, and Botnet. We infuse CIA principle in the model that provides a concise and interpretable set of important features. Computer security rest on CIA principles, *C* stands for *confidentiality*—concealment of information or resources, *I* stands for *integrity*—trustworthiness of data or resources, and *A* stands for *availability*—ability to use the information or resource desired (Matt and others 2006). For instance, security compromise in *confidentiality* could happen through eavesdropping unencrypted data, compromise in *integrity* could happen through an unauthorized attempt to change data, and the compromise in *availability* could happen through the deliberate arrangement of denial to access data or service. We also convert the domain knowledge infused features into three features C, I, and A by quantitatively computing compromises associated with each of those for each record. Output expressed as these generalized and newly constructed set of features provides better explainability with negligible compromises in performance. We also found that generalization provides more resiliency against unknown attacks.

In summary, our contributions in this work are as follows: (1) we demonstrate a method for the collection and use of domain knowledge in an intrusion detection/response system; (2) we introduce a way to bring popular security principles (e.g., CIA principles) to aid in interpretability and explainability; (3) our experimental results show that infusing domain knowledge into "black box" models can make them better explainable with little or no compromise in performance ; and (4) domain knowledge infusion increases generalizability, which leads to better resiliency against unknown attack.

We start with a background of related work (Section 2) followed by a description of our proposed approach, an intuitive description of standard supervised algorithms, and an overview of the dataset (Section 3) used in this work. In Section 4, we describe our experiments, followed by Section 5 which contains discussion on results from the experiments. We conclude with limitations and future work in Section 6.

2 Background

Research in Explainable Artificial Intelligence (XAI) is a re-emerging field, after the earlier work of (Chandrasekaran, Tanner, and Josephson 1989), (Swartout and Moore 1993), and (Swartout 1985). Previous work focused on primarily explaining the decision process of knowledge-based systems and expert systems. The classical learning paradigm Explanation-Based Learning (EBL), introduced in the early '80s, can also be regarded as a precursor of explainability. EBL involves learning a problem-solving technique by observing and analyzing solutions to a specific problem (DeJong 1981), (Mitchell, Keller, and Kedar-Cabelli 1986). The main reason for the renewed interest in XAI research has stemmed from recent advancements in AI and ML and their application to a wide range of areas, as well as concerns over unethical use and undesired biases in the models. In addition, recent concerns and laws by different governments are necessitating more research in XAI.

(Yang and Shafto 2017), use *Bayesian Teaching*, where a smaller subset of examples is used to train the model instead of the whole dataset. The subset of examples is chosen by domain experts as the examples most relevant to the problem of interest. However, for this purpose, choosing the right subset of examples in the real-world is challenging.

(Lei, Barzilay, and Jaakkola 2016) propose an approach for sentiment analysis where a subset of text from the whole text is selected as the rationale for the prediction. In addition, the selected subset of text is concise and sufficient enough to act as a substitute for the original text, and still capable of making the correct prediction. Although their approach outperforms available attention-based models (from deep learning) with variable-length input (e.g., a model for document summarization) , it is limited to only text analysis.

When the explanation is based on feature importance, it is necessary to keep in mind that features that are globally important may not be important in the local context, and vice versa (Ribeiro, Singh, and Guestrin 2016). (Ribeiro, Singh, and Guestrin 2016) propose a novel explanation technique capable of explaining the prediction of any classifier (i.e., in the model agnostic way) in a locally interpretable model (i.e., in the vicinity of the instance being predicted) around the prediction. Their concern is on two issues: (1) whether the user should trust the prediction of the model and act on that, and (2) whether the user should trust a model to behave reasonably-well when deployed. In addition, they involve human judgment in their experiment (i.e., human in the loop) to decide whether to trust the model or not.

(Kim et al. 2017) propose a concept attribution-based approach (i.e., sensitivity to concept) that provides an interpretation of the neural network's internal state in terms of human-friendly concepts. Their approach, *Testing with CAV (TCAV)*, quantifies the prediction's sensitivity to the high dimensional concept. For example, a user-defined set of examples that defines the concept 'striped', TCAV can quantify the influence of 'striped' in the prediction of 'zebra' as a single number. To learn the high dimensional concepts they use a Concept Activation Vector (CAV) —CAVs are learned from training a linear classifier that can distinguish between the activations produced by a particular concept's examples and examples in any layer.

Most of these approaches try to find out how the prediction deviates from the base/average scenario. Lime (Ribeiro, Singh, and Guestrin 2016) tries to generate an explanation by locally (i.e., using local behavior) approximating the model with an interpretable model (e.g., decision trees, linear model). However, it is limited by the use of the only linear model to approximate the local behavior. (Lundberg and Lee 2017) propose "SHAP" which unifies seven previous approaches: LIME (Ribeiro, Singh, and Guestrin 2016), DeepLIFT (Shrikumar, Greenside, and Kundaje 2017), Tree Interpreter (Ando), QII (Datta, Sen, and Zick 2016), Shapley sampling values (Štrumbelj and Kononenko 2014), Shapley regression values (Lipovetsky and Conklin 2001), and Layer-wise relevance propagation (Bach et al. 2015) to make the explanation of prediction for any machine learning model. While SHAP comes with theoretical guarantees about consistency and local accuracy from game theory, it needs to run many evaluations of the original model to estimate a single vector of feature importance (Lundberg). ELI5 also uses the LIME algorithm internally for explanations. In addition, ELI5 is not truly model agnostic, mostly limited to tree-based and other parametric or linear models. Furthermore, Tree Interpreter is limited to only tree-based approaches (e.g., Random Forest, Decision Trees).

AI-based IDSs have continued to show promising performance (Hodo et al. 2016),(Shone et al. 2018),(Kim et al. 2016),(Javaid et al. 2016),(Li, Sun, and Wang 2012). (Shone et al. 2018) propose an approach in combination of both shallow (Random Forest) and deep learning (Auto Encoder), capable of analyzing a wide range of network traffic, outperforming mainstream Deep Belief Networks (DBN). In a literature survey on traditional IDS vs deep learning IDS by (Dong and Wang 2016), they suggest deep learning-based methods provide better accuracy for a wide range of samples sizes and a variety of network traffic or attacks (Dong and Wang 2016). However, in all of the previous work, there are still long training times and a reliance on a human operator (Shone et al. 2018).

However, incorporating domain knowledge for explainability has garnered little attention. Previously, we introduced the concept of infusing domain knowledge (Islam et al. 2019), albeit for bankruptcy prediction with a limited focus. (Miller 2018) have argued that incorporating knowledge from different domains will provide better explainability. In addition, (Kim et al. 2017) use the prediction's sensitivity to high dimensional concepts (e.g., the concept 'striped' to 'Zebra') for explaining the prediction. Furthermore, both LIME (Ribeiro, Singh, and Guestrin 2016) and SHAP (Lundberg and Lee 2017) use a simplified input mapping—mapping the original input to a simplified set of input. To the best of our knowledge, none of the models incorporate domain knowledge with a focus towards better explainability and interpretability. Although our proposed conceptual model comes with a negligible compromise in accuracy, it comes with better explainability and interpretibility, and scalability to big data problems.

3 Methodology

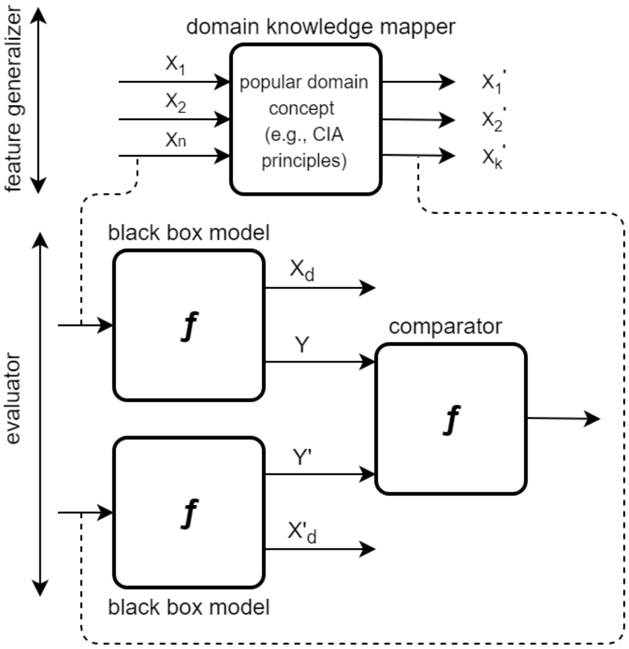

Figure 1: Proposed Technique

3.1 Proposed Approach

The proposed approach consists of two components: a *feature generalizer*, which gives a generalized feature set with the help of domain knowledge in two different ways; and an *evaluator* that produces and compares the results from the "black box" model for multiple configurations of features: domain knowledge infused features, newly constructed features from domain knowledge infused features, selected features, and all features.

3.2 Feature Generalizer

The *feature generalizer* (Figure 1, top portion), takes original features of the dataset (X_1, X_2, ... $X_\mathrm{n} \in X$ where X is the set of all features) and infuse domain knowledge to

Table 1: Mapping of network attack with related component of CIA principles

Attack	Related component of CIA
DoS GoldenEye	A
Heartbleed	C
DoS hulk	A
DoS Slowhttp	A
DoS slowloris	A
SSH-Patator	C
FTP-Patator	C
Web Attack	C, I, A
Inflitration	C
Bot	C, I, A
PortScan	C
DDoS	**A**

produce/re-construct a concise and better interpretable feature set (X_1', X_2', ... X_k' $\in$ *X'* where *X'* is the universal set of original/transformed/constructed features, but here *k* is much smaller than *n*) in two different ways:

Feature Mapping As stated earlier, we use CIA principles as domain knowledge, which stands for *confidentiality*, *integrity*, and *availability*. We analyze all types of attacks for associated compromises in each component of CIA principles (see Table 1). The *Heartbleed* vulnerability is related to a compromise in *confidentiality* as an attacker could gain access to the memory of the systems protected by the vulnerable version of the OpenSSL. A *Web attack* (e.g., Sql injection) is related to a compromise in *confidentiality* and *integrity* (e.g., read/write data using injected query), and *availability* (e.g., flooding the database server with injected complex queries like a cross join). *Infiltration* attack is related to a compromise in *confidentiality* as it normally exploits software vulnerability (e.g., Adobe Acrobat Reader) to create a backdoor and reveal information (e.g., IPs). *Port scan* attack is related to a compromise in *confidentiality* as the attacker sends packets with varying destination ports to learn the services and operating systems from the reply. All *DoS* and *DDoS* attacks are related to a compromise in *availability* as it aims to hamper the availability of service or data. Furthermore, *SSH patator* and *FTP patator* are brute force attacks and are usually responsible for a compromise in *confidentiality*. Botnet (i.e., robot network—a network of malware-infected computers) could provide a remote shell, file upload/download option, screenshot capture option, and key logging options which has potential for all of the *confidentiality, integrity, and availability* related compromises.

Furthermore, from the feature ranking of the original dataset provider (Sharafaldin, Lashkari, and Ghorbani 2018), for each type of attack, we take the top three features according to their importance (i.e., feature importance from Random Forest Regressor) and calculate the mapping (Table 2) with related compromises under CIA principles. For example, the feature *Average Packet Size* is renamed as *Avg Packet Size - A* where -A indicates that it is a key feature for the compromise of *availability* (see Table 2). To get this mapping between feature and associated compromises, we first find the mapping between an attack and related compromises (from Table 1, formulated as Equation 2). In other words, Formula 1 gives the name of the associated attack from the feature name and Formula 2 gives associated compromises from the attack name. Thus, with the help of domain knowledge, we keep 22 features (see Table 2) out of a total of 78 features. We will refer to these features as the *domain features*. The feature descriptions in Table 2 are taken from the data processing software's website (net).

$$f(feature) \rightarrow attack \tag{1}$$

$$f(attack) \rightarrow C, I, orA \tag{2}$$

Feature Construction We also construct three new features, C, I, and A, from the domain features by quantitatively calculating compromises associated with each of the domain features. For that purpose, we calculate the correlation coefficient vector of the dataset to understand whether the increase in the value of a feature has a positive or negative impact on the target variable. We then convert the correlation coefficient (a.k.a *coeff*) vector V in to a 1 or -1 based on whether the correlation coefficient is positive or negative accordingly. We also group the domain features and corresponding *coeff* tuple into three groups. Using formula 3, 4, and 5, we aggregate each group (from C, I, and A) of domain features into the three new features C, I, and A. We also scale all feature values from 0 to 1 before starting the aggregation process. During the aggregation for a particular group (e.g., C), if the correlation coefficient vector (e.g., V_i) for a feature (e.g., C_i) of that group has a negative value, then the value for that feature is deducted, and vice-versa if positive. In addition, when a feature is liable for more than one compromise, the feature value is split between the associated elements of CIA principles.

$$C = \sum_{i=0}^{n} C_i V_i \tag{3}$$

$$I = \sum_{i=0}^{n} I_i V_i \tag{4}$$

$$A = \sum_{i=0}^{n} A_i V_i \tag{5}$$

3.3 Evaluator

The task of the *evaluator* (Figure 1, bottom side) is to execute and compare the performance (in detecting malicious and benign records) of four different types of models, as follows: (1) using all features, (2) using selected features (selection is done by feature selection algorithm), (3) using domain knowledge infused features, and (4) using newly constructed features C, I, and A from domain knowledge infused features. In addition, the *evaluator* performs the following two tests:

1. Explainability Test: The purpose of this test is to discover the comparative advantages or disadvantages of incorporating domain knowledge in the experiment; and

Table 2: Mapping of feature with related component of CIA principles

Feature	Description	In top 3 features of attack	Renamed feature
ACK Flag Count	Number of packets with ACK	SSH-Patator	ACK Flag Count - C
Active Mean	Mean time a flow was active before becoming idle	DoS Slowhttp, Infiltration	Active Mean - AC
Active Min	Minimum time a flow was active before becoming idle	DoS Slowhttp	Active Min - A
Average Packet Size	**Average size of packet**	**DDoS**	**Avg Packet Size - A**
Bwd IAT Mean	Mean time between two packets sent in the backward direction	DoS slowloris	Bwd IAT Mean - A
Bwd Packet Length Std	Standard deviation size of packet in backward direction	DoS Hulk, DoS GoldenEye, DDoS, Heartbleed, DoS Hulk	Bwd Packet Length Std - AC
Bwd Packets/s	Number of backward packets per second	Bot, PortScan	Bwd Packets/s - CIA
Fwd IAT Mean	Mean time between two packets sent in the forward direction	DoS slowloris	Fwd IAT Mean - A
Fwd IAT Min	Minimum time between two packets sent in the forward direction	DoS slowloris, DoS GoldenEye	Fwd IAT Min - A
Fwd Packet Length Mean	Mean size of packet in forward direction	Benign, Bot	Fwd Packet Length Mean - CIA
Fwd Packets/s	Number of forward packets per second	FTP-Patator	Fwd Packets/s - C
Fwd PSH Flags	Number of times the PSH flag was set in packets travelling in the forward direction (0 for UDP)	FTP-Patator	Fwd PSH Flags - C
Flow Duration	Duration of the flow in Microsecond	DDoS, DoS slowloris, DoS Hulk, DoS Slowhttp, Infiltration, Heartbleed	Flow Duration - AC
Flow IAT Mean	Mean inter-arrival time of packet	DoS GoldenEye	Flow IAT Mean - A
Flow IAT Min	Minimum inter-arrival time of packet	DoS GoldenEye	Flow IAT Min - A
Flow IAT Std	Standard deviation inter-arrival time of packet	DDoS, DoS Slowhttp, DoS Hulk	Flow IAT Std - A
Init_Win_bytes_forward	The total number of bytes sent in initial window in the backward direction	Web Attack	Init Win Bytes Fwd - CIA
PSH Flag Count	Number of packets with PUSH	PortScan	PSH Flag Count - C
Subflow Fwd Bytes	The average number of packets in a sub flow in the forward direction	Benign, SSH-Patator, Web Attack, Bot, Heartbleed, Infiltration	Subflow Fwd Bytes - CIA
SYN Flag Count	Number of packets with SYN	FTP-Patator	SYN Flag Count - C
Total Length of Fwd Packets	Total size of packet in forward direction	Benign, SSH-Patator, Web Attack, Bot, Heartbleed, Infiltration	Total Length of Fwd Packets - CIA

2. Generalizability Test: The purpose of this test is to analyze how different approaches perform in unknown or unseen attack detection. We delete all training records for a particular attack one at a time and investigate the performance of the model on the same test set, which includes records from unknown or unseen attacks. Details of these tests are described in Section 4.

3.4 Algorithms

We use six different supervised algorithms for predicting malicious records: one of those is a probabilistic classifier based on Naive Bayes theorem, and the remaining five are supervised ”black box” models. The algorithm descriptions are taken from our previous work (Islam et al. 2019).

Artificial Neural Network (ANN) An Artificial Neural Network is a non-linear model, capable of mimicking human brain functions to some extent. It consists of an input layer, one or multiple hidden layer(s), and the output layer. Each layer consists of multiple neurons that help to learn the complex pattern. Each subsequent layer learns more abstract concepts before it finally merges into the output layer.

Support Vector Machine (SVM) The Support Vector Machine (SVM) was first introduced by (Boser, Guyon, and Vapnik 1992) and has been used for many supervised classification tasks. In addition to linear classification, the model can learn an optimal hyperplane that separates instances of different classes using a highly non-linear implicit mapping of input vectors in high dimensional feature space (i.e., kernel trick) (Hooman et al. 2016). When the number of samples is too high (i.e., millions) then it is very costly in terms of computation time.

Random Forest (RF) A Random Forest is a tree-based ensemble technique developed by (Breiman 2001) for the supervised classification task. In RF, many trees are generated from the bootstrapped subsamples (i.e., random sample drawn with replacement) of the training data. In each tree, the splitting attribute is chosen from a smaller random subset of attributes of that tree (i.e., the chosen split attribute that is the best among that random subset). This randomness helps to make trees less correlated as correlated trees make the same kinds of prediction errors and can overfit the model. In less correlated trees, a few trees may be wrong but many others will be right and as a group the trees can move in the right direction as the output from all the trees are averaged for the final prediction.

Extra Trees (ET) Extremely Randomized Trees or Extra Trees (ET) is a tree-based ensemble technique simialr to RF. The only difference is in the process of splitting attribute selections and determining the threshold (cutoff) value, both are chosen in an extremely random fashion (Islam, Ghafoor, and Eberle 2018). Similar to RF, a random subset of features are taken into consideration for the split selection, but instead of choosing the most discriminative cut off threshold, ET cut off thresholds are set to random values. Thus, the best of these randomly chosen values is set as the threshold for the splitting rule (ens) on a particular node. Unlike DT, RF has multiple trees which leads to a reduced variance. However, bias is introduced, as a subset of the whole feature set is chosen for each tree instead of all features. ET was proposed by (Geurts, Ernst, and Wehenkel 2006), and has achieved a state of the art performance in some anomaly/intrusion detection research (Islam 2018), (Islam, Eberle, and Ghafoor 2018),(Islam, Ghafoor, and Eberle 2018).

Gradient Boosting (GB) (Friedman 2001), generalized Adaboost to a Gradient Boosting algorithm to allow a variety of loss function. Here the shortcoming of weak learners is identified using the gradient instead of highly weighted data points as in Adaboost. Gradient Boosting (GB) is a classifier/regression model in the form of an ensemble of weak prediction models, such as Decision Trees. It works sequentially like the AdaBoost algorithm, in that each subsequent model tries to minimize the loss function (i.e., Mean Squared Error) by paying special focus on instances that were hard to get right in the previous model.

Naive Bayes (NB) Naive Bayes algorithm is based on Bayes Theorem, which was formulated in the seventeenth century. It is a supervised, simple, and comparatively fast algorithm based on statistics. In a real-world problem, it is unusual that all features are independent. However, Naive Bayes assumes conditional independence among features and surprisingly works well in many cases. It also requires a small amount of training data to estimate the necessary parameters (nai). This assumption of Naive Bayes helps to avoid lots of computations (e.g., computing the conditional probability for each feature with others) and makes it a faster algorithm. Besides, the avoidance of a conditional probability calculation helps (the class conditional feature distribution can be independently estimated as one-dimensional distribution) in Big Data problems where the curse of dimensionality is a concern. However, NB is a bad estimator of a probabilty (Zhang 2004). We use the Bernoulli Naive Bayes (Manning, Raghavan, and 2010) for our experiments where each feature is assumed to be binary-valued.

3.5 Data

In this work, we use a recent and comprehensive IDS dataset namely CICIDS2017, published in 2017, covers necessary criteria with common updated attacks such as DoS, DDoS, Brute Force, XSS, SQL Injection, Infiltration, Portscan, and Botnet. In fact, this dataset is created to eliminates the shortcomings (e.g., lack of traffic diversity and volume, lack of variety of attacks, anonymized packet information, and out of date) of previous well known IDS dataset such as DARPA98, KDD99, ISC2012, ADFA13, DEFCON, CAIDA, LBNL, CDX, Kyoto, Twente, and UMASS since 1998. This is a labeled dataset containing 78 network traffic features (some features are listed in Table 2) extracted and calculated from pcap file using CICFlowMeter software (Lashkari et al. 2017) for all benign and intrusive flows (Sharafaldin, Lashkari, and Ghorbani 2018) . This new IDS dataset includes seven common updated family of attacks satisfying real-world criteria, also publicly available at here: https://www.unb.ca/cic/datasets/ids-2017.html .

Each record of the dataset is labeled by the particular type of attack. We make a new feature 'Class', which is the target feature. We set the value of the 'Class' attribute to 1 for all records labeled as any of 14 types of attacks, as those are malicious/intrusive, and set the value to 0 for the remaining records as those are benign. Following that, in the whole dataset, there are total 2,830,743 records for 14 different attacks, 2,273,097 are benign and 557,646 are malicious. Approximately 24.5% of the records are malicious, giving us an imbalanced dataset which impacts the performance (e.g., bias to the class of majority samples) of some machine learning algorithms. To overcome this problem, we use the well-known oversampling technique SMOTE (Chawla et al. 2002) to oversample the minority class. In (Dong and Wang 2016), the author uses SMOTE to overcome the issue in their empirical study on the comparison of traditional vs deep learning-based IDS. SMOTE creates synthetic samples rather than just oversampling with replacement. The minority class is oversampled by creating new examples along with the line segments joining any or all of k nearest minority samples, where k is chosen based on the percentage of oversampling required (i.e., hyperparameter to the algorithm) (Chawla et al. 2002).

4 Experiments

4.1 Experimental Setup

We execute the experiments in a GPU enabled Linux machine with 12GB of RAM and core i7 processor. All supervised machine learning algorithms are implemented using the Python-based *Scikit-learn* (sci) library. In addition, we use *Tensorflow* (ten) for the Artificial Neural Network. Due to resource limitations, instead of using the whole dataset, we take a stratified sample of the data which is big enough (i.e., 300K records) for a single GPU enabled commodity machine. We make the sampled dataset available to the research community at (sam). Furthermore, we use 70% of the data for training the models and kept 30% of the data as a holdout set to test the model. We confirm the target class had the same ratio in both sets. To avoid the adverse effect of class imbalance in classification performance, we re-sample the minority class of the training set using SMOTE (Chawla et al. 2002) to balance the dataset. However, we do not resample the test set, as real-world data is skewed and oversampling the test set could exhibit an overoptimistic performance.

We run all supervised machine learning algorithm using four different approaches:

1. With all features: using all 78 features of the dataset without discarding any features.
2. With selected features: using *Random Forest Regressor* (adapting with the work of (Sharafaldin, Lashkari, and Ghorbani 2018)) to select important features of the dataset, giving us 50 important features having a nonzero influence on the target variable;
3. With domain knowledge infused features: using infused domain knowledge features (see Section 3.2), we will use the term *domain features* interchangeably to express it in short form; and
4. With newly constructed features from domain knowledge infused features: using newly constructed features C, I, and A (see Section 3.2) from domain knowledge infused features, we will use the term *domain features-constructed* interchangeably to express it in short form.

Furthermore, we perform two types of experiments using each of the four feature settings:

4.2 Explainability Test

For this test, we run six supervised algorithms RF, ET, SVM, GB, ANN, and NB using the four described feature settings

Table 3: Performance using all features vs selected features

Alg.	Acc.	Prec.	Rec.	F-score	AUC
RF-A	**0.9987**	**0.9965**	**0.9971**	**0.9968**	**0.9997**
RF-S	0.9986	0.9962	0.9966	0.9964	**0.9997**
	0.0002	0.0003	0.0006	0.0005	0.0000
ET-A	0.9981	0.9951	0.9951	0.9951	0.9994
ET-S	0.9980	0.9950	0.9950	0.9950	0.9994
	0.0001	0.0002	0.0001	0.0001	0.0000
ANN-A	0.9802	0.9155	0.9908	0.9516	0.9984
ANN-S	0.9740	0.8929	0.9860	0.9372	0.9968
	0.0062	0.0226	0.0047	0.0145	0.0017
SVM-A	0.9109	0.6996	0.9595	0.8092	0.9780
SVM-S	0.8869	0.6433	0.9565	0.7692	0.9746
	0.0239	0.0563	0.0030	0.0400	0.0034
GB-A	0.9960	0.9854	0.9944	0.9899	0.9995
GB-S	0.9957	0.9840	0.9945	0.9892	0.9996
	0.0003	0.0014	(0.0001)	0.0007	(0.0001)
NB-A	0.7753	0.4371	0.4888	0.4615	0.8601
NB-S	0.7621	0.4144	0.5019	0.4539	0.8508
	0.0132	0.0228	(0.0131)	0.0076	0.0093

and report the results Section 5. Unlike NB, other classifiers are ”black box” in nature. NB is a probabilistic classifier based on Bayes Theorem with strong conditional independence assumption among features. The main purpose to include NB in the experiment is the generalizability test.

4.3 Generalizability Test

For testing the generalizability of the approach, we train the classifier without the representative of a particular attack, and test it with the presence of the representative of that particular attack, in order to classify it malicious/benign. To be more specific, we delete all records of a particular attack from the training set, train the classifier with the records of the remaining 13 attacks, and test the classifier with all 14 attacks. We report the percentage of deleted attacks that are correctly detected as malicious. We repeat this one by one for all 14 attacks. We make the source code available to the research community to replicate the experiments at (pro).

5 Results

The results from our experiments can be broadly categorized into two categories:

5.1 Findings from Explainability Test

Comparing the performance using *all features* vs *selected features*, (Table 3) shows that models using *all features* (denoted with an appended -A, for instance RF-A) tend to show better results in terms of all performance metrics. However, while the difference with the *selected features* setting is negligible ($<.0007$ for RF) for any performance metric, that might be a result of the elimination of features with little significance. In addition, Random Forest outperforms other algorithms SVM, ET, GB, ANN, and NB under this feature setting (i.e., using all features). So we consider the results using all features as a baseline to compare with our proposed approach.

Before starting the comparison of results from our approach with all features (i.e., baseline), we seek the best feature setting among two domain related feature settings of our proposed approach. In other words, in our attempt to find the better approach among using domain knowledge infused features vs newly constructed features (C, I, and A) from domain knowledge infused features, we find that, in almost all cases, the model with domain knowledge infused features (denoted with an appended -D1, for instance RF-D1) performs better than the counterpart (see Table 4). Although for RF, the maximum performance gap is .2 in the recall, for ET that gap is .048 with a similar precision. As the domain features (22 features) contain a lot more detail than the newly constructed features C, I, and A (3 features), it loses few details. In terms of individual algorithms, RF is again a clear winner this time using domain features. Although NB and ANN exhibit better recall using constructed features, it comes with compromises in precision. So, overall we consider the domain features setting as the best over the constructed features.

Table 4: Performance using domain features vs constructed features

Alg.	Acc.	Prec.	Rec.	F-score	AUC
RF-D1	**0.9973**	**0.9920**	**0.9945**	**0.9932**	**0.9993**
RF-D2	0.9511	0.9446	0.7985	0.8654	0.9572
	0.0463	0.0475	0.1960	0.1278	0.0421
ET-D1	0.9969	0.9913	0.9932	0.9923	0.9989
ET-D2	0.9756	0.9321	0.9448	0.9384	0.9954
	0.0214	0.0592	0.0483	0.0538	0.0036
ANN-D1	0.9497	0.8300	0.9362	0.8799	0.9865
ANN-D2	0.5952	0.3241	0.9721	0.4862	0.7921
	0.3544	0.5059	(0.0359)	0.3937	0.1945
SVM-D1	0.8489	0.5747	0.8968	0.7005	0.9252
SVM-D2	0.7195	0.3739	0.6281	0.4687	0.7886
	0.1294	0.2008	0.2687	0.2318	0.1366
GB-D1	0.9881	0.9513	0.9904	0.9705	0.9986
GB-D2	0.9230	0.7692	0.8701	0.8165	0.9789
	0.0652	0.1821	0.1204	0.1539	0.0198
NB-D1	0.7982	0.4881	0.5028	0.4953	0.8553
NB-D2	0.5591	0.2687	0.7195	0.3913	0.6591
	0.2391	0.2194	(0.2167)	0.1040	0.1962

While we know the best feature setting is the *all features*, as shown in the comparison of *all features* vs *selected features* in the Table 3), we also know the best feature setting *domain features* from *domain features* vs *constructed features* (see Table 4). So we further compare the performance of models using the two best settings *all features* (i.e., baseline) vs *domain features*. We find that, among all models, RF using all features (denoted with an appended -A, for instance RF-A) performs better than all other algorithms (see Table 5 and Figure 3). Interestingly, RF using domain knowledge infused features (denoted with an appended -D1, for instance RF-D1) also shows promising performance. The difference between these two in terms of any performance metrics is negligible ($<.005$). In fact, the result of RF using the domain knowledge infused feature settings is better than what (Sharafaldin, Lashkari, and Ghorbani 2018) reports using the same dataset. The slight improvement might stem from the experimental settings (e.g., training test set split, re-sampling techniques). Furthermore, in the domain knowledge infused feature setting we are using only 22 features out of 78 total, where each feature indicates the associated compromises (e.g., confidentiality, integrity, or availability), capable of producing better explainable and interpretable results compared to the counterpart. The prediction

Table 5: Performance using all features vs domain features

Alg.	Acc.	Prec.	Rec.	F-score	AUC
RF-A	**0.9987**	**0.9965**	**0.9971**	**0.9968**	**0.9997**
RF-D1	0.9973	0.9920	0.9945	0.9932	0.9993
	0.0014	0.0045	0.0027	0.0036	0.0004
ET-A	0.9981	0.9951	0.9951	0.9951	0.9994
ET-D1	0.9969	0.9913	0.9932	0.9923	0.9989
	0.0011	0.0038	0.0020	0.0029	0.0004
ANN-A	0.9802	0.9155	0.9908	0.9516	0.9984
ANN-D1	0.9497	0.8300	0.9362	0.8799	0.9865
	0.0305	0.0855	0.0546	0.0717	0.0119
SVM-A	0.9109	0.6996	0.9595	0.8092	0.9780
SVM-D1	0.8489	0.5747	0.8968	0.7005	0.9252
	0.0619	0.1249	0.0627	0.1087	0.0528
GB-A	0.9960	0.9854	0.9944	0.9899	0.9995
GB-D1	0.9881	0.9513	0.9904	0.9705	0.9986
	0.0079	0.0341	0.0039	0.0194	0.0009
NB-A	0.7753	0.4371	0.4888	0.4615	0.8601
NB-D1	0.7982	0.4881	0.5028	0.4953	0.8553
	(0.0229)	(0.0510)	(0.0140)	(0.0338)	0.0048

for a particular sample can be represented as:

$$P(D) = \sum_{g=0}^{G} contribution(g) \tag{6}$$

where g is the generalized domain feature (e.g., ACK Flag Count - C), P(D) is the probability value of the decision. Instead of using contributions from each of the domain features, we can express the output in terms of the contribution from each element of the domain concept. For that, we need to aggregate contributions from all features into three groups (C,I, and A). This will enable an analyst to understand the nature of the attack more quickly (Figure 2). For instance, when the greater portion of a feature contribution for a sample is from features tagged with -A (i.e., *Availability*) then it might be a DDoS attack, which usually comes with very high compromises in *Availability* of data or service. We use R's iml package to generate the sample wise breakdown of features contribution (Figure 2).

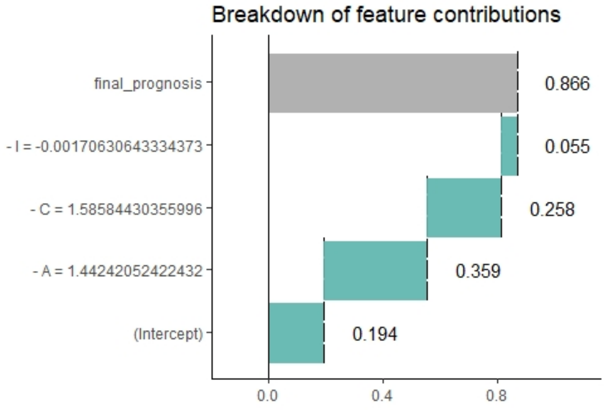

Figure 2: Breakdown of the prediction for a random sample in terms of C, I, and A.

5.2 Findings from Generalizability Test

Recall that the purpose of this test is to test the resiliency against unknown attacks. First, we use Random Forest (RF), the best performing algorithm so far using all four settings of features. As shown in Table 6 and Figure 4, we see that except for the constructed feature settings (denoted by Cons.), the performances of other feature settings (all, selected, and domain) are similar. The constructed features fail to provide comparable performance for RF as it has only three features and loses lots of detail of data (too much generalization). Surprisingly, a few unknown attacks are only detectable using the domain knowledge infused features. For instance, *Web Attack Sql Injection* is detected as suspicious only by domain knowledge infused features. Overall, although the *domain knowledge infused* feature setting perform slightly worse than the *all feature* setting, it comes with an explainable features set with the added capability of identifying a few unknown attacks.

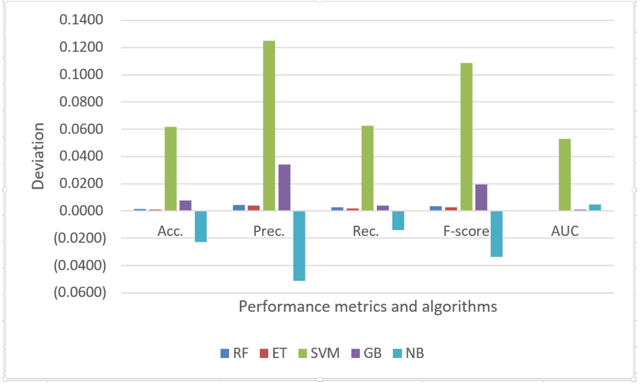

Figure 3: Performance deviations of using domain knowledge infused features from baseline

Table 6: Performance of unseen attack detection using RF

Attack	Count	All(%)	Sel.(%)	Dom.(%)	Cons.(%)
Ddos	4184	99.90	99.90	99.90	62.86
PortScan	4973	99.90	99.94	99.94	66.28
Bot	54	77.78	77.78	75.93	22.22
Infiltration	1	100	100	100	0.00
Web Attack-BF	49	95.92	95.92	91.84	75.51
Web Attack-XSS	23	95.65	95.65	91.30	65.22
Web Attack-Sql	1	**0.00**	**0.00**	**100**	**0.00**
FTP-Patator	251	99.20	100	99.20	81.67
SSH-Patator	198	98.99	99.49	96.97	75.76
DoS slowloris	188	99.47	99.47	98.94	61.70
DoS Slowloris	174	99.43	99.43	96.55	31.61
Dos Hulk	7319	99.71	99.73	99.34	96.19
DoS GoldenEye	314	99.36	99.68	98.41	85.03
Heartbleed	1	100	100	100	100

To reiterate, the constructed features set consists of only three features (C, I, and A) constructed from aggregating domain knowledge infused features. As this feature setting is composed of only three features, it is an extreme generalization of features and it loses a lot of details of data. However, this time it comes with an exceptional capability which we realize after applying a statistical approach (Naive Bayes) on the dataset. We find that (see Table 7), for NB, the newly constructed feature setting is best as NB is also able to detect unknown attacks with similar accuracy compared to other feature settings by RF in Table 6. The most interesting thing about this capability is that this feature set is composed of

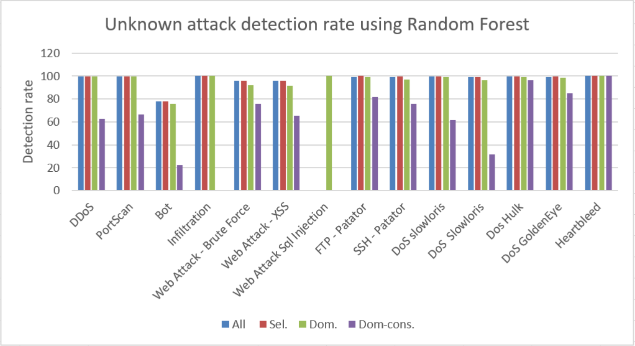

Figure 4: Unknown attack detection rate using Random Forest

Table 7: Performance of unseen attack detection using NB

Attack	Count	All(%)	Sel.(%)	Dom.(%)	Cons.(%)
Ddos	4184	76.94	74.59	83.22	**100**
PortScan	4973	0.18	0.18	6.64	**100**
Bot	54	0.00		0.00	**100**
Infiltration	1	0.00	0.00	0.00	**100**
Web Attack-BF	49	6.12	6.12	83.67	**100**
Web Attack-XSS	23	0.00	0.00	95.65	**100**
Web Attack-Sql	1	**100**	**100**	0.00	**100**
FTP-Patator	251	0.00	0.00	0.00	48.61
SSH-Patator	198	0.00	0.00	0.00	**100**
DoS slowloris	188	25.53	30.85	38.30	**66.49**
DoS Slowloris	174	71.26	**79.89**	75.29	78.16
Dos Hulk	7319	**67.78**	**67.78**	**67.03**	35.37
DoS GoldenEye	314	50.32	65.29	47.13	**96.82**
Heartbleed	1	**100**	**100**	0.00	0.00

only three features (C, I, and A), takes comparatively less time to execute, and comes with the added benefit of very good explainability. Once the prediction is expressed as a percentage of influence from each of C, I, and A, the analyst would be able to perceive the level of compromise more intuitively from the hints about the type of attack (e.g., DDoS will show a high percentage of A—compromise in Availability).

However, from Table 3, 4, and 5, we can see that NB's performance comes at a cost of precision and recall (i.e., produces comparatively more false positives and false negatives). In addition, NB is a bad probability estimator of the predicted output (Zhang 2004). However, NB with constructed features setting could be recommended as an additional IDS for quick interpretation of huge traffic data given the decision is treated as tentative with the requirement of a further sanity check. We also calculate the average time taken by each algorithm for all four feature settings and found that NB is the fastest algorithm. RF, ET, GB, ANN, and SVM take 2.80, 9.27, 77.06, 15.07, and 444.50 times more execution time compared to NB. Besides, the best algorithm, RF (1^{st} in terms of the performance metric and 2^{nd} in terms of execution time), can be executed in parallel using an *Apache Spark* for a far better run-time (Chen et al. 2016) making it highly scalable to big data problems.

Overall, domain knowledge infusion provides better explainability with negligible compromises in performance. In addition, the generalization provides better execution time and resiliency with unknown attacks.

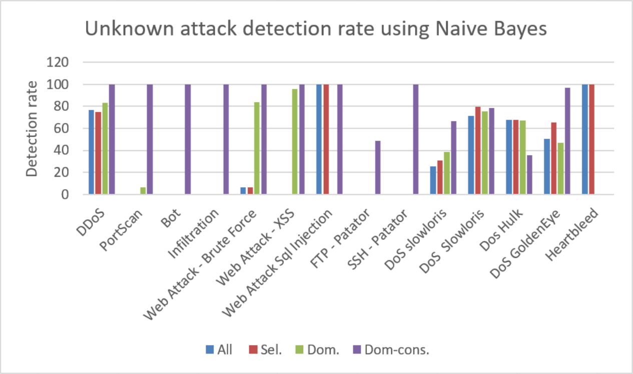

Figure 5: Unknown attack detection rate using Naive Bayes

6 Conclusion and Future Work

AI-based approaches have become an integral part of security solutions due to the potential for handling ”big data” and handling diverse network traffic data. Cybercrime related damages continue to rise, and network intrusions are a key tactic. Although AI-based IDS provides accelerated speeds in intrusion *detection*, *response* is still at a human speed where there is human in the loop. The lack of explainability of an AI-based model is a key reason for this bottleneck. To mitigate this problem, we infuse the CIA principle (i.e., domain knowledge) in the AI-based black box model for better explainability and generalizability of the model. Our experimental results show realizable successes in better explainability with a comprehensive, up to date, and real-world network intrusion dataset. In addition, the infused domain knowledge helps in detecting an unknown attack as it generalizes the problem, which ultimately opens the door to accommodate big data.

Going forward, finding an optimal solution to segregate the contribution of each participating feature (sample wise) considering interactions (i.e., correlations among features complicate explanations) among features will aid in better explainability of an individual prediction (i.e., per sample). Besides, to ensure trust, estimating the level of uncertainty in the model will be another extension of this work. There are some open challenges surrounding explainability and interpretability such as an agreement of what an explanation is and to whom, a formalism for the explanation, and quantifying the human comprehensibility of the explanation.

Acknowledgment

Thanks to Tennessee Tech Cyber-security Education, Research and Outreach Center (CEROC) for funding this research.

References

[Ando] Ando, S. Interpreting random forests. http://blog.datadive.net/interpreting-random-forests/.

[Bach et al. 2015] Bach, S.; Binder, A.; Montavon, G.; Klauschen, F.; Müller, K.-R.; and Samek, W. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one* 10(7):e0130140.

[Boser, Guyon, and Vapnik 1992] Boser, B. E.; Guyon, I. M.; and Vapnik, V. N. 1992. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, 144–152. ACM.

[Breiman 2001] Breiman, L. 2001. Random forests. *Machine learning* 45(1):5–32.

[Chandrasekaran, Tanner, and Josephson 1989] Chandrasekaran, B.; Tanner, M. C.; and Josephson, J. R. 1989. Explaining control strategies in problem solving. *IEEE Intelligent Systems* (1):9–15.

[Chawla et al. 2002] Chawla, N. V.; Bowyer, K. W.; Hall, L. O.; and Kegelmeyer, W. P. 2002. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research* 16:321–357.

[Chen et al. 2016] Chen, J.; Li, K.; Tang, Z.; Bilal, K.; Yu, S.; Weng, C.; and Li, K. 2016. A parallel random forest algorithm for big data in a spark cloud computing environment. *IEEE Transactions on Parallel and Distributed Systems* 28(4):919–933.

[Datta, Sen, and Zick 2016] Datta, A.; Sen, S.; and Zick, Y. 2016. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *2016 IEEE symposium on security and privacy (SP)*, 598–617. IEEE.

[DeJong 1981] DeJong, G. 1981. Generalizations based on explanations. In *IJCAI*, volume 81, 67–69.

[Dong and Wang 2016] Dong, B., and Wang, X. 2016. Comparison deep learning method to traditional methods using for network intrusion detection. In *2016 8th IEEE International Conference on Communication Software and Networks (ICCSN)*, 581–585. IEEE.

[Doyle] Doyle, M. Dont be lulled into a false sense of security. https://www.securityroundtable.org/dont-lulled-false-sense-cybersecurity/.

[ens] Ensemble methods. https://scikit-learn.org/stable/modules/ensemble.html.

[Friedman 2001] Friedman, J. H. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* 1189–1232.

[Geurts, Ernst, and Wehenkel 2006] Geurts, P.; Ernst, D.; and Wehenkel, L. 2006. Extremely randomized trees. *Machine learning* 63(1):3–42.

[Goodman and Flaxman 2017] Goodman, B., and Flaxman, S. 2017. European union regulations on algorithmic decision-making and a right to explanation. *AI Magazine* 38(3):50–57.

[Hodo et al. 2016] Hodo, E.; Bellekens, X.; Hamilton, A.; Dubouilh, P.-L.; Iorkyase, E.; Tachtatzis, C.; and Atkinson, R. 2016. Threat analysis of iot networks using artificial neural network intrusion detection system. In *2016 International Symposium on Networks, Computers and Communications (ISNCC)*, 1–6. IEEE.

[Hooman et al. 2016] Hooman, A.; Marthandan, G.; Yusoff, W. F. W.; Omid, M.; and Karamizadeh, S. 2016. Statistical and data mining methods in credit scoring. *The Journal of Developing Areas* 50(5):371–381.

[Islam et al. 2019] Islam, S. R.; Eberle, W.; Bundy, S.; and Ghafoor, S. K. 2019. Infusing domain knowledge in ai-based" black box" models for better explainability with application in bankruptcy prediction. *arXiv preprint arXiv:1905.11474*.

[Islam, Eberle, and Ghafoor 2018] Islam, S. R.; Eberle, W.; and Ghafoor, S. K. 2018. Credit default mining using combined machine learning and heuristic approach. *arXiv preprint arXiv:1807.01176*.

[Islam, Ghafoor, and Eberle 2018] Islam, S. R.; Ghafoor, S. K.; and Eberle, W. 2018. Mining illegal insider trading of stocks: A proactive approach. In *2018 IEEE International Conference on Big Data (Big Data)*, 1397–1406. IEEE.

[Islam 2018] Islam, S. R. 2018. *An efficient technique for mining bad credit accounts from both olap and oltp*. Ph.D. Dissertation, Tennessee Technological University.

[Javaid et al. 2016] Javaid, A.; Niyaz, Q.; Sun, W.; and Alam, M. 2016. A deep learning approach for network intrusion detection system. In *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*, 21–26. ICST (Institute for Computer Sciences, Social-Informatics and .

[Kim et al. 2016] Kim, J.; Kim, J.; Thu, H. L. T.; and Kim, H. 2016. Long short term memory recurrent neural network classifier for intrusion detection. In *2016 International Conference on Platform Technology and Service (PlatCon)*, 1–5. IEEE.

[Kim et al. 2017] Kim, B.; Wattenberg, M.; Gilmer, J.; Cai, C.; Wexler, J.; Viegas, F.; and Sayres, R. 2017. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). *arXiv preprint arXiv:1711.11279*.

[Lashkari et al. 2017] Lashkari, A. H.; Draper-Gil, G.; Mamun, M. S. I.; and Ghorbani, A. A. 2017. Characterization of tor traffic using time based features. In *ICISSP*, 253–262.

[Lei, Barzilay, and Jaakkola 2016] Lei, T.; Barzilay, R.; and Jaakkola, T. 2016. Rationalizing neural predictions. *arXiv preprint arXiv:1606.04155*.

[Li, Sun, and Wang 2012] Li, Z.; Sun, W.; and Wang, L. 2012. A neural network based distributed intrusion detection system on cloud platform. In *2012 IEEE 2nd international conference on Cloud Computing and Intelligence Systems*, volume 1, 75–79. IEEE.

[Lipovetsky and Conklin 2001] Lipovetsky, S., and Conklin, M. 2001. Analysis of regression in game theory approach. *Applied Stochastic Models in Business and Industry* 17(4):319–330.

[Lipton 2016] Lipton, Z. C. 2016. The mythos of model interpretability. *arXiv preprint arXiv:1606.03490*.

[Lundberg and Lee 2017] Lundberg, S. M., and Lee, S.-I. 2017. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, 4765–4774.

[Lundberg] Lundberg, S. Shap vs lime. https://github.com/slundberg/shap/issues/19.

[Manning, Raghavan, and 2010] Manning, C.; Raghavan, P.; and , H. 2010. Introduction to information retrieval. *Natural Language Engineering* 16(1):100–103.

[Matt and others 2006] Matt, B., et al. 2006. *Introduction to computer security*. Pearson Education India.

[Miller 2018] Miller, T. 2018. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*.

[Mitchell, Keller, and Kedar-Cabelli 1986] Mitchell, T. M.; Keller, R. M.; and Kedar-Cabelli, S. T. 1986. Explanation-based generalization: A unifying view. *Machine learning* 1(1):47–80.

[mle] Machine learning explainability. https://www.kdnuggets.com/2018/12/machine-learning-explainability-interpretability-ai.html.

[Montavon, Samek, and Müller 2018] Montavon, G.; Samek, W.; and Müller, K.-R. 2018. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing* 73:1–15.

[nai] Naive bayes. https://scikit-learn.org/stable/modules/naive_bayes.html.

[net] Netflow meter. http://netflowmeter.ca/netflowmeter.html.

[pro] domain-knowledge-aided code. https://github.com/SheikhRabiul/domain-knowledge-aided-explainable-ai-for-intrusion-detection-and-response.

[Rankin] Rankin, K. The dark secret at the heart of ai. https://www.technologyreview.com/s/604087/the-dark-secret-at-the-heart-of-ai/.

[Ribeiro, Singh, and Guestrin 2016] Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2016. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 1135–1144. ACM.

[sam] domain-knowledge-aided dataset. https://github.com/SheikhRabiul/domain-knowledge-aided-explainable-ai-for-intrusion-detection-and-response/tree/master/data/combined_sampled.zip.

[sci] Scikit-learn: Machine learning in python. https://scikit-learn.org/stable/.

[Sharafaldin, Lashkari, and Ghorbani 2018] Sharafaldin, I.; Lashkari, A. H.; and Ghorbani, A. A. 2018. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *ICISSP*, 108–116.

[Shone et al. 2018] Shone, N.; Ngoc, T. N.; Phai, V. D.; and Shi, Q. 2018. A deep learning approach to network intrusion detection. *IEEE Transactions on Emerging Topics in Computational Intelligence* 2(1):41–50.

[Shrikumar, Greenside, and Kundaje 2017] Shrikumar, A.; Greenside, P.; and Kundaje, A. 2017. Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 3145–3153. JMLR. org.

[Štrumbelj and Kononenko 2014] Štrumbelj, E., and Kononenko, I. 2014. Explaining prediction models and individual predictions with feature contributions. *Knowledge and information systems* 41(3):647–665.

[Swartout and Moore 1993] Swartout, W. R., and Moore, J. D. 1993. Explanation in second generation expert systems. In *Second generation expert systems*. Springer. 543–585.

[Swartout 1985] Swartout, W. R. 1985. Rule-based expert systems: The mycin experiments of the stanford heuristic programming project: Bg buchanan and eh shortliffe,(addison-wesley, reading, ma, 1984); 702 pages.

[ten] Tensorflow. https://www.tensorflow.org/.

[Turek] Turek, M. Explainable ai. https://www.darpa.mil/program/explainable-artificial-intelligence.

[Wyden] Wyden, B. Algorithmic accountability. https://www.wyden.senate.gov/imo/media/doc/Algorithmic%20Accountability%20Act%20of%202019%20Bill%20Text.pdf.

[Yang and Shafto 2017] Yang, S. C.-H., and Shafto, P. 2017. Explainable artificial intelligence via bayesian teaching. In *NIPS 2017 workshop on Teaching Machines, Robots, and Humans*.

[Zhang 2004] Zhang, H. 2004. The optimality of naive bayes. *AA* 1(2):3.