ABSTRACT

Title of Dissertation: AUTOMATED CALCULATION OF A RISK DECISION FOR A TEXTUAL DOCUMENT USING PROBABILISTIC NEURAL NETWORK

Garfield S. Jones, Doctor of Engineering, May 2019

Dissertation Chair: Tridip K. Bardhan, Ph.D.
Department of Engineering

Organizations have been struggling to make objective risk decisions concerning cyber since the dawn of the Internet. Most risk based decisions are made at the strategic level, where senior decision makers weigh subjective expert information to determine cyber risk. The Common Vulnerability Scoring System (CVSS) is one of the primary methods cyber risk is evaluated. The CVSS contains base, temporal, and environmental scoring approaches. Although, a quantitative score is produced, the score is determined largely by subjective means and does not allow for a quick objective determination by system administrators of whether a textual document is a threat. Developing an objective risk evaluation process at a tactical level will assist the senior decision makers with a more quantitative portion of their risk decision process. At the lowest level, risk decisions on whether a textual file should be accepted or not quickly based on a quantitative method is the primary objective for this paper. A search algorithm is used to detect the words or phrases that are possible threats to the system. The threat update will come through the Common Vulnerabilities and Exposures or a public database. A weight must

be added to the generated score to allow for the time that the vulnerability is in the database. Finally, the use of a Probabilistic Neural Network to classify the file quickly for acceptance, quarantine, or denial by the system administrator will be determined objectively and rapidly.

AUTOMATED CALCULATION OF A RISK DECISION FOR A TEXTUAL DOCUMENT USING PROBABILISTIC NEURAL NETWORK

by

Garfield S. Jones

A Dissertation Submitted in Partial Fulfillment
of the Requirements for the Degree
Doctor of Engineering

MORGAN STATE UNIVERSITY

February 2019

AUTOMATED CALCULATION OF A RISK DECISION FOR A TEXTUAL DOCUMENT USING PROBABILISTIC NEURAL NETWORK

by

Garfield S. Jones

has been approved

February 2019

DISSERTATION COMMITTEE APPROVAL:

_____, Chair
Tridip K. Bardhan, Ph.D.

_____
Gaungming Chen, Ph.D.

_____
Jessye Leigh Bemley Talley, Ph.D.

Acknowledgments

I would like to thank my brother Dr. Selwyn Jones for all his guidance and patience with helping to facilitate this dissertation. I would also like to thank my committee at Morgan State University, Dr. Tridip Bardhan, Dr. Gaungming Chen, Dr. Jessye Bemley Talley, and Dr. Bheem Kattel for their guidance on this paper. My wife Rebecca and my children, Jalen, Allyson, Keanna, and Natalia, I truly appreciate all the patience you had with me going through the years of fulfilling this milestone. I would also like to thank all my coworkers especially Dr. Arthur Fries, Mr. Kevin Cox, Mrs. Betsy Kulick, Mr. Ross Foard, and Mr. Chad Baer for all their encouragement to complete this paper. Lastly, I would like to thank my mother Sybil, my father Maurice Sr. (deceased), my brothers (Maurice, Gerald, and Menzies), my sisters (Desiree, Corinne, Camille, and my shining star Dawn [deceased]). Without this army of support, I would not have been able to complete this dissertation.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

The numbers of cyberattacks on networks are growing in number, and becoming more sophisticated, aggressive and dynamic in execution (Awan & Rana, 2016). The increasing numbers of devices on the networks have spawned an even more complex number of software applications on the networks. Measuring the risk of an intrusion or a compromise and identifying the most recent tactics of cyber criminals on large computer networks can be difficult (Awan & Rana, 2016). Primarily, this is due to the wide range of services and applications running within the network, the multiple vulnerabilities associated with each application, the severity associated with each vulnerability, and the ever-changing attack vector of cyber criminals (Awan & Rana, 2016; Lund, 2011). A vulnerability is defined as "a defect which enables an attacker to bypass security measures" (Alhazm, Sung-Whan, & Malaiya, 2006, p. 2). Software vulnerabilities are high priorities within any information technology IT system. Like hardware, software can also pose a significant risk to IT systems and networks. To assist in combating this increasing cyber threat within networks, organizations focused on identifying cyber risk and threats on the network. In this paper, the focus is to identify a quantitative method which facilitates a classification of a text document as a threat or not based on existing risk scoring and assessment mechanisms. For a system administrator to make an informed decision, the risk of acceptance, denial, or quarantine can be scored, so that a quick decision on the textual document can be obtained.

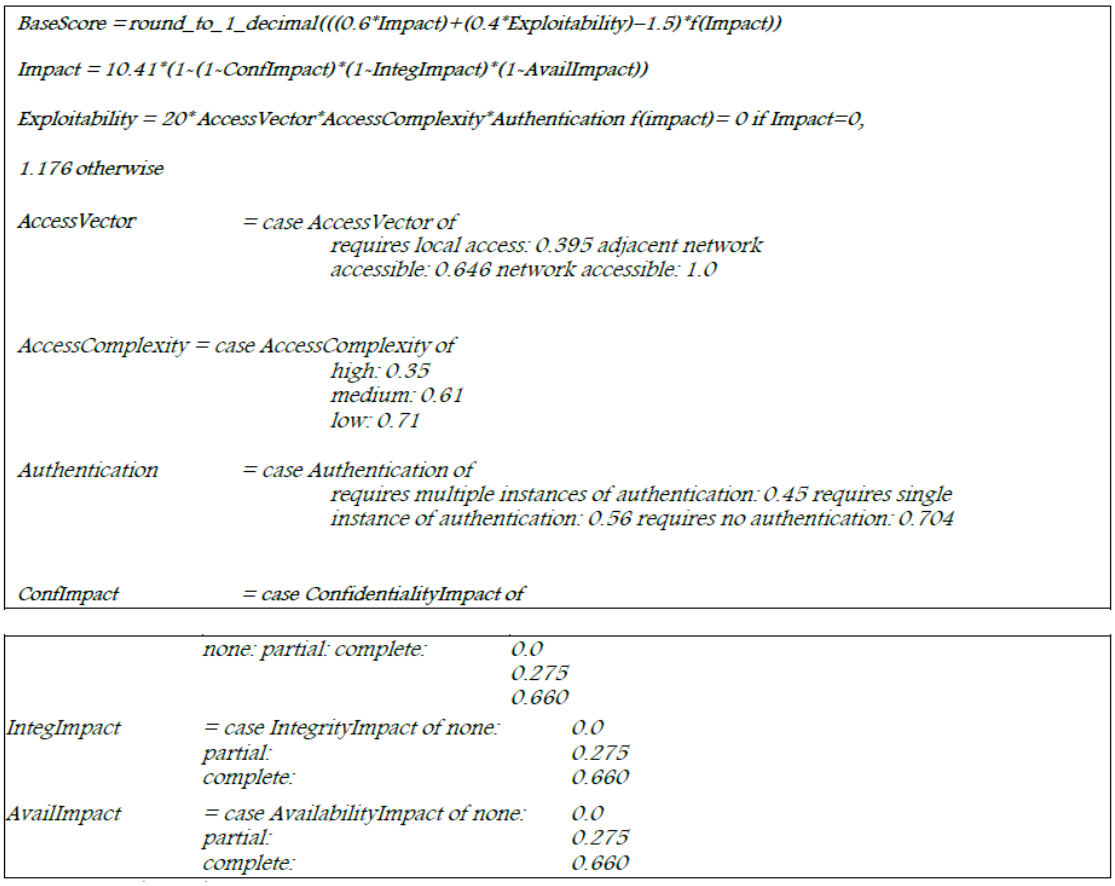
The acceptance of a document by a system administrator is a risk decision. Risk is defined as the net negative impact of the exercise of a vulnerability, considering both the

probability and the impact of occurrence (National Institute of Standards and Technology, 2012). Essentially, risk is a product of threat likelihood value multiplied by impact (Khambhammettu, 2013). Quantitively risk is difficult to assess, so a subjective element is generally part of the score. Some of the more widely used risk scoring mechanisms are credibility of risk assessment (CORAS), International Organization for Standardization/International Electrotechnical Commission ISO/IEC27001, and Common Vulnerability Scoring System (CVSS), require subjective, expert qualitative inputs, which could potentially lead to inaccurate (ambiguous)and inconclusive results (Awan & Rana, 2016; Oparaugo, 2015).

The CORA framework is designed for helping nonexperts in judging the credibility of risk assessments (Wiedemann, 2013).The CORA framework can be used by (a) stakeholders and policy makers, to make an educated judgment about the credibility of an assessment, and (b) the authors of a risk assessment, to improve the evaluability of their reports (Wiedemann, 2013) ISO/IEC27001 is the most used standard within the information security field (Talib, 2012). It is used by organizations that manage information on behalf of others and it is applied to assure the protection of critical client information (Talib, 2012). The application of ISO standards generally is costlier due to the expert human assistance required to apply the standard (Talib, 2012). CVSS is a scoring method used to quantify the severity of security vulnerability (Holm & Afridi, 2015).

This expert knowledge is not always available or accessible, so a more accurate and objective method is proposed in this paper. The National Vulnerability Database

(NVD) is the U.S. government repository of standards based vulnerability management data represented using the Security Content Automation Protocol (SCAP; DHS/NCCIC, 2018). This data enables automation of vulnerability management, security measurement, and compliance (DHS/NCCIC, 2018). The NVD includes databases of security checklist references, security-related software flaws, misconfigurations, product names, and impact metrics (DHS/NCCIC, 2018). The CVSS helps organizations prioritize and coordinate a joint response to security vulnerabilities by communicating the Base, Temporal, and environmental properties of a vulnerability (Zhang, Ou, & Caragea, 2015). The Base, Temporal, and Environmental are the three measures that quantify the severity of vulnerabilities using the CVSS (Holm & Afridi, 2015).The Base metric assumes the fundamental characteristics of vulnerability are constant over time and user environments (Mell, Scarfone, & Romanosky, 2007). The Temporal represents the characteristics of vulnerability that change over time but not among user environments (Mell et al., 2007). Moreover, the Environmental metric represents the characteristics of vulnerability that are relevant and unique to a particular user's environment (Mell et al., 2007). The formulas for each are represented in Figures 1, 2, and 3 for a each metric, respectively.



*BaseScore =round_to_1_decimal((((0.6*Impact)+(0.4*Exploitability)−1.5)*f(Impact))*

Impact = 10.41(1-(1-ConfImpact)*(1-IntegImpact)*(1-AvailImpact))*

*Exploitability = 20*AccessVector*AccessComplexity*Authentication f(impact)= 0 if Impact=0,*

1.176 otherwise

AccessVector = case AccessVector of
requires local access: 0.395 adjacent network
accessible: 0.646 network accessible: 1.0

AccessComplexity = case AccessComplexity of
high: 0.35
medium: 0.61
low: 0.71

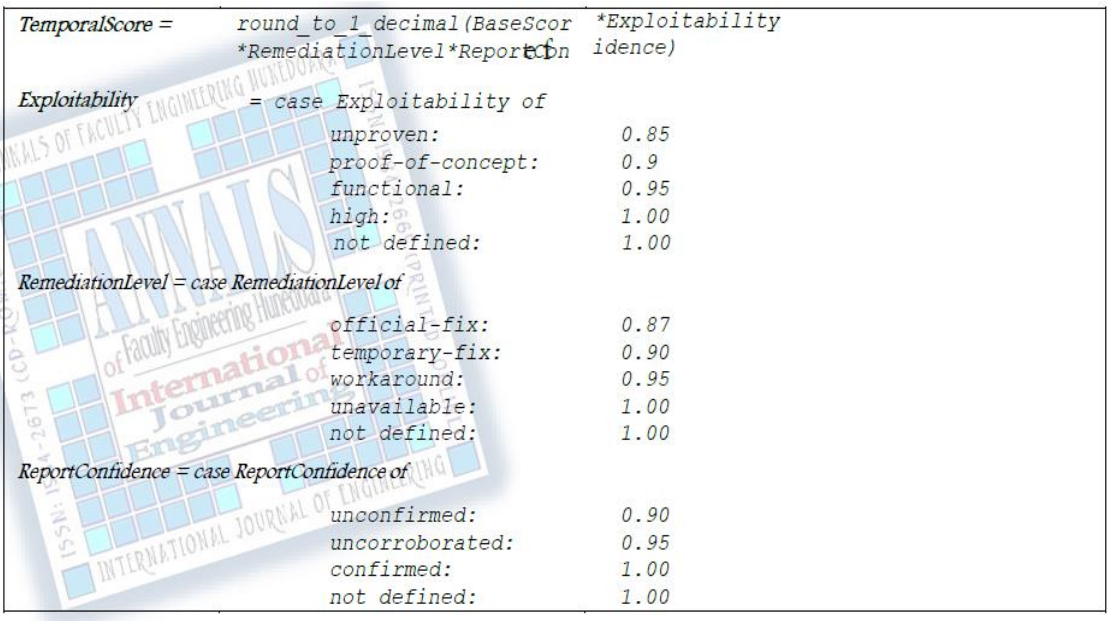
Authentication = case Authentication of
requires multiple instances of authentication: 0.45 requires single
instance of authentication: 0.56 requires no authentication: 0.704

ConfImpact = case ConfidentialityImpact of
none: partial: complete: *0.0*
0.275
0.660

IntegImpact = case IntegrityImpact of none: *0.0*
partial: *0.275*
complete: *0.660*

AvailImpact = case AvailabilityImpact of none: *0.0*
partial: *0.275*
complete: *0.660*

Figure 1. CVSS base metric (Cisar, Rajnai, Cisar, & Pinter, 2016).



```
TemporalScore =      round_to_1_decimal(BaseScor  *Exploitability
                     *RemediationLevel*Reportcon  idence)

Exploitability       = case Exploitability of
                       unproven:              0.85
                       proof-of-concept:      0.9
                       functional:            0.95
                       high:                  1.00
                       not defined:           1.00

RemediationLevel = case RemediationLevel of
                       official-fix:          0.87
                       temporary-fix:         0.90
                       workaround:            0.95
                       unavailable:           1.00
                       not defined:           1.00

ReportConfidence = case ReportConfidence of
                       unconfirmed:           0.90
                       uncorroborated:        0.95
                       confirmed:             1.00
                       not defined:           1.00
```

Figure 2. CVSS temporal metric (Cisar et al., 2016).

EnvironmentalScore = round_to_1_decimal((AdjustedTemporal+
*(10-AdjustedTemporal)*CollateralDamagePotential)*TargetDistribution)*

AdjustedTemporal = TemporalScore recomputed with the BaseScore's Impact sub- equation replaced with the AdjustedImpact equation

AdjustedImpact = min(10,10.41(1-(1-ConfImpact*ConfReq)*(1-IntegImpact*IntegReq)*(1-AvailImpact*AvailReq)))*

CollateralDamagePotential = case CollateralDama
none: low: *gePotential of 0*
low-medium: medium-high: high: *0.1*
not defined: *0.3*
0.4
TargetDistribution = case TargetDistribu *0.5*
none: low: medium: high: *0*
not defined:
tion of 0
0.25
0.75
1.00
1.00

ConfReq = case ConfReq of low: medium: high:
not defined:
0.5
1.0
1.51
1.0

IntegReq = case IntegReq of low: medium: high:
not defined:
0.5
1.0
1.51
1.0

AvailReq = case AvailReq of low: medium: high:
not defined:
0.5
1.0
1.51
1.0

Figure 3. CVSS environmental metric (Cisar et al., 2016).

Additionally the metrics are composed of the elements in Figure 4.

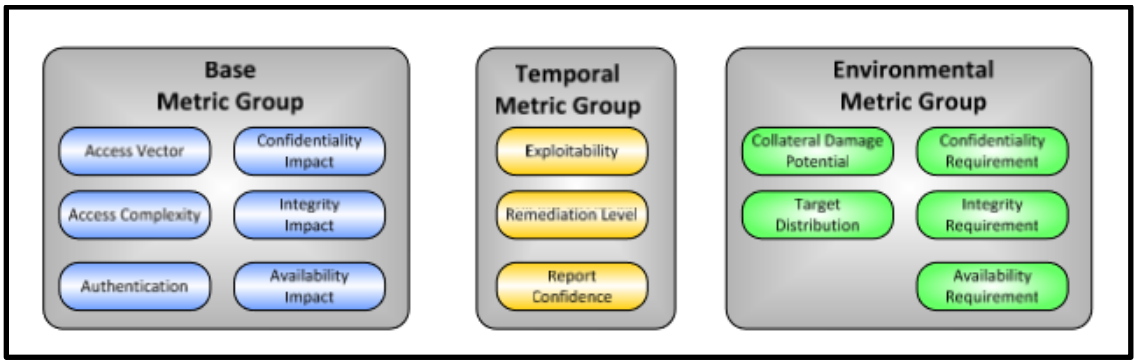


Figure 4. CVSS metric groups (Mell et al., 2007).

The CVSS provides a way to capture the major features of a vulnerability resulting in a numerical score equating to a severity, as well as a textual representation of the score (FIRST.Org, 2018).

Probabilistic Neural Network (PNN) is a classifier algorithm and can be used as an approximator mapping any input pattern to a number of classifications (Cheung & Cannons, 2002). By replacing the sigmoid activation function often used in neural networks with an exponential function, a PNN that can computed nonlinear decision boundaries which approach the Bayes optimal is formed (Specht, 1990). The PNN is an implementation of a statistical algorithm called kernel discriminant analysis in which the operations are organized into a multilayered feedforward network with four layers (Cheung & Cannons, 2002):

- Input layer
- Pattern layer
- Summation layer
- Output layer

The Input layer supplies the extracted features from the dataset (A. K. Singh, 2011.). In the pattern layer, the total number of neurons is equal to the sum of the numbers of neurons used to represent the patterns for each class. Each class can contain a large number of training patterns (training vectors) of which dimension is the same as the number of input factors, while it is taking a set of specific values of input factors (Modaresi & Araghinejad, 2014).

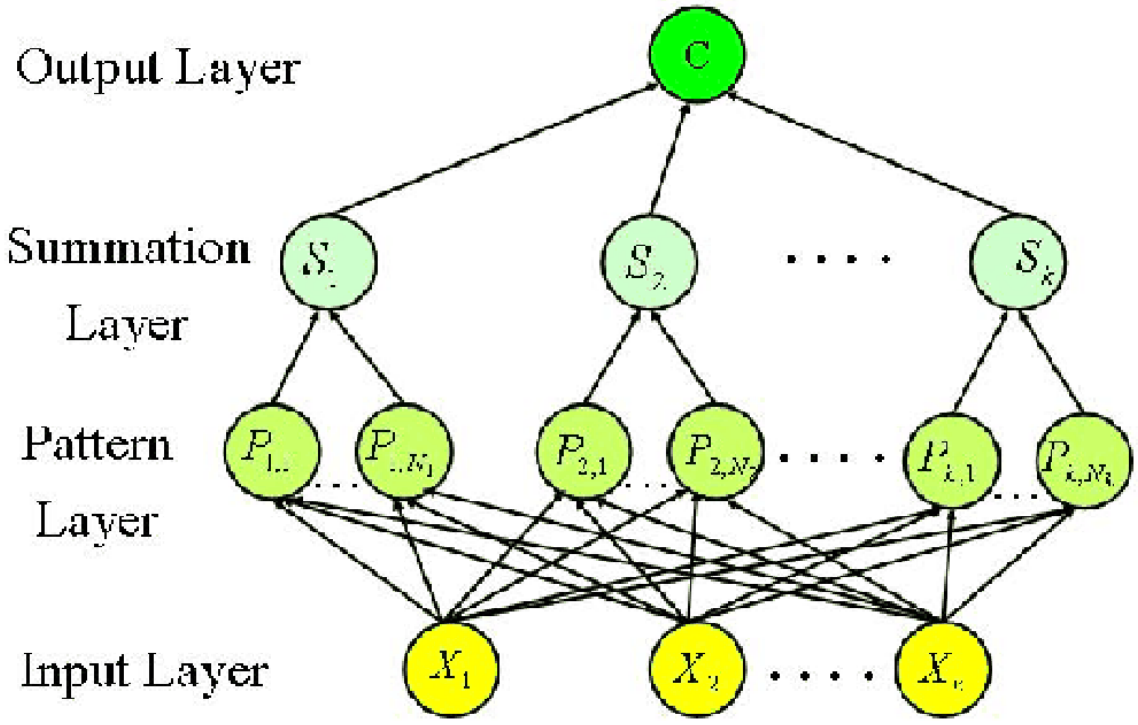


Figure 5. PNN layout (A. K. Singh, 2011).

The PNN is this paper is used to assist the system administrator or designated user in prompt classification of a textual document whether it is a threat or not (A. K. Singh, 2011).The Hidden or Summation layer is where the total "n" artificial neurons take in a set of weighted inputs and produce an output through an activation function (Cheung & Cannons, 2002). Parameters differ when a sample population is known verses unknown (Cheung & Cannons, 2002). In this paper, the estimator used assesses that the sample population of the textual document is known. The Pattern layer sometimes called the Summation layer is where all neurons are totaled within this layer (A. K. Singh, 2011). The final layer is the Output layer; it decides in which class test sample belongs by

comparing the values of the pattern layer (A. K. Singh, 2011). In this paper the PNN is modelled using the Python language to produce a graphical output.

The textual data mining begins with identifying a document that comes into network through an e-mail attachment, thumb drive, downloaded by an end user, or other electronic means. In this paper, the bag-of-words approach is engaged to allow for the possibility that tens of thousands of different words occur within a set of documents (Bramer, 2013). The elimination of stop words or filler words is a technique that is used within this paper to allow for reduction of common words within the text document (Bramer, 2013). Additionally, an algorithm for stemming is used to identify and reduce amount the words that have the same root word, but may have variants based on the prefixes or suffixes that the root word may carry (Bramer, 2013; McEwan, 2018).

The training set of data in a PNN should consist of typical samples and patterns; must be sufficiently representative so hyperspace of problem is covered well, especially near decision surfaces (Nimbhorkar, 2014).

The datasets used in this paper was provided by the University of California Irvine (UCI) database (Dua, 2017). The UCI provides over 452 datasets for the machine learning community (Dua, 2017). The datasets used in this paper are the IRIS and Balance Scale datasets (Dua, 2017). The driving purpose for the selection of these datasets was based principally on the dataset containing 3 classes which would align with the method outlined in this paper (Dua, 2017).

Background of the Problem

Cybersecurity is defined as information security aimed at averting cyberattacks, which are among the main issues caused by the extensive use of networks within organizations and personally (de Gusmão, 2018). Various models have been developed to lower the risk of cybersecurity focusing on the strategic level, but the problem can also be approached from the user/client level. This paper focuses on how to lower the risk from the client level systems with an emphasis on allowing or not allowing text documents within a network based on user decision. Additionally, integrating the use of CVSS metrics to assist in the decision analysis provides a more reliable metric of if to accept the text document or not.

Statement of the Problem

Given a text document or possibly a set of textual documents sent to a user or a system administrator within a computer network. A decision must be made whether to accept, deny, or quarantine such document or set of documents. At times, antivirus software has not been updated and/or the organization has identified a vulnerability that has not made it to the NVD or posted as a CVE, which is part of the calculation in CVSS metrics. Additionally, there may be time lag between the identification a vulnerability and when an organization is notified of this defect. These time gaps reflect the escalation of risk to an organizational network if a vulnerability is not identified for remediation in time. Essentially, the larger the time gap to identify the vulnerability, the higher the possible risk to the organizational network. Additionally, there is a possible increase in cost associated with the increase in organizational risk. Therefore, the assumption of this

paper is the risk strategy can be implemented at the senior level, but must be diligently applied from the lowest level to be effective.

Purpose of the Study

The purpose of this study is multidisciplinary software engineering and development research paper of a process and application for evaluating a textual document using the PNN to classify the document as a threat or a possible threat. The purpose is to develop an effective method to predict the CVSS score that a text document would receive if evaluated by the CVEs stored in the NVD. Therefore, a user can now make an informed decision of whether to accept the document into an organizational or user network environment. Finally, the author will develop python scripts to parse and import data from text documents into the PNN for proper classification of the document.

Significance of the Study

The significance of the paper allows the user or system administrator to weigh in on weather a text document should be accepted or not. This allows for a more updated and informed approach from the user perspective. The antivirus software may not be updated as such, where this method allows for input from the user with a weighted metric and scoring process to accept, deny, or quarantine a text document.

Chapter 2: Literature Review

As stated by Khazaei, not very much attention or research has been directed towards identifying textual information hidden vulnerabilities databases, but in this paper, the objective is to find vulnerabilities in textual documents to assist the system administrator in the determination of a document efficiently and effectively (Khazaei, Ghasemzadeh, & Derhami, 2016).

Textual Data Mining

Khazaei, Ghasemzadeh, and Derhami discuss an objective method for CVSS score calculation (Khazaei et al., 2016). The article details a method giving priority to software vulnerabilities (Khazaei et al., 2016). The data that is used is from the Open Sourced Vulnerability Database (OSVDB) database, and is extracted using text mining tools and techniques (Khazaei et al., 2016). The OSVDB is a large public database containing reports on over 100,000 vulnerabilities until the end of 2013 (Khazaei et al., 2016). The OSVDB is used in this article, but the database was shut down in 2016 due to excessive vulnerabilities (Gold, 2016). In addition, the article details the use of Support Vector Machines (SVM) and Random-Forest algorithms as well as fuzzy systems are examined to predict the concerned CVSS scores (Khazaei et al., 2016). The article defines vulnerability and discusses how organizations such as CERT/CC (Computer Emergency Readiness Team/Coordination Center) calculate numeric scores from 0 to 180 based on factors that concern the exploitation on the vulnerability related to the internet infrastructure (Khazaei et al., 2016). The article uses the Base CVSS metric to compare vulnerabilities with each other. The use of the Base metric was used in the article and the

Temporal metric was emphasized in this paper. The goal of the study was to predict the CVSS base scores based on vulnerability description using text mining (Khazaei et al., 2016). The article details the process of data and feature extraction from an additional public database such as Mozilla Foundation Security Advisories (MFSA; Khazaei et al., 2016). The process started with removing the stop words and stemming, which is very similar to the process in this paper (Khazaei et al., 2016). Next the article detailed the importance of a word in a document by using the following formula:

$$f_{ij}\ log \frac{number\ of\ documents}{number\ of\ documents\ that\ include\ word\ i}$$

f_{ij} is the frequencies for work *i* in document *j* (Khazaei et al., 2016).

Table 1

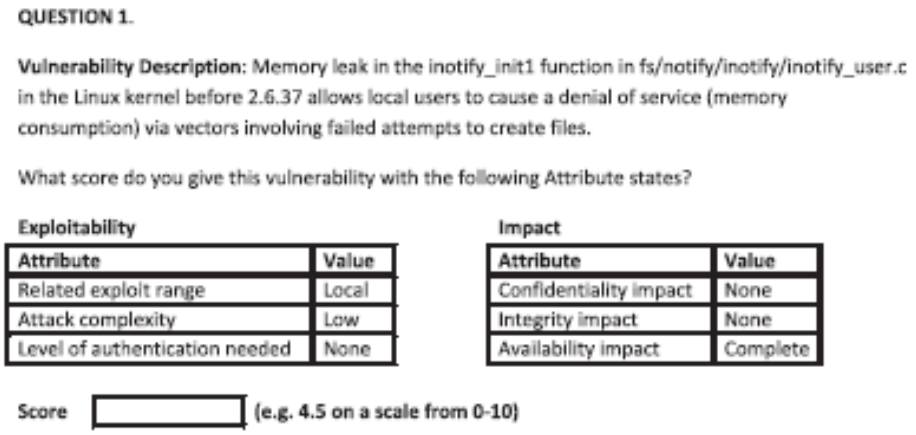
SVM and Random-Forest Results (Khazaei et al., 2016)

		SVM	Random-Forest
Without data dimension reduction	Training	57.8%	86.92%
	Testing	55.89%	66.09%
The LDA dimension reduction	Training	86.46%	99.1%
	Testing	86.23%	85.78%
The LDA dimension reduction on the PCA features	Training	86.45%	99.14%
	Testing	86.11%	85.3%

To create the CVSS predictors the SVM and Random-Forest algorithms and fuzzy systems were examined as mentioned earlier. Using of these automatic predictors reduces the human errors and increases the CVSS calculation speed (Khazaei et al., 2016). The best predictor was assessed to be a fuzzy system with a CVSS score accuracy predictor of

88% accuracy. This article is the most closely related article to this paper due to its use of text mining methods to evaluate the most accurate method for predicting a CVSS score.

The next related work to this paper by Holm and Afridi is a study of the accuracy of CVSS scores (Holm & Afridi, 2015). The article studies the accuracy through a survey with opinions by 384 experts, covering more than 3000 vulnerabilities (Holm & Afridi, 2015). The results showed the mean disagreement between the experts and the CVSS Base Score to be about -0.38, with a variance of 4.46 (Holm & Afridi, 2015). This article as detailed used the Base metric versus the Temporal metric used in this paper. This article was mainly qualitative with virtually no use of mathematical assertions to claims detailed in the article. Holm explains the Base metric from CVSS version 2, which at the time of the article was the only version available at the time (Holm & Afridi, 2015). The method used in the article began with population sampling of the experts, consensus, the survey and operationalization of research questions (Holm & Afridi, 2015). An example of a question was given:



QUESTION 1.

Vulnerability Description: Memory leak in the inotify_init1 function in fs/notify/inotify/inotify_user.c in the Linux kernel before 2.6.37 allows local users to cause a denial of service (memory consumption) via vectors involving failed attempts to create files.

What score do you give this vulnerability with the following Attribute states?

Exploitability

Attribute	Value
Related exploit range	Local
Attack complexity	Low
Level of authentication needed	None

Impact

Attribute	Value
Confidentiality impact	None
Integrity impact	None
Availability impact	Complete

Score [ ] **(e.g. 4.5 on a scale from 0-10)**

Figure 6. Example survey question (for CVE 2010-4250; Holm & Afridi, 2015).

Finally, the selection of experts was determined from the three questions given to the 384 respondents by the correlation of consensus scores for each respondent (Holm & Afridi, 2015). The validity and reliability of this method was addressed within the article, but not with a mathematical approach as detailed in this paper. The results of the article showed there is a significant difference between the scores provided by experts and the Base Score. The differences in specific vulnerability types were rated by experts higher than other vulnerabilities lower than the Base Score such as XSS and information exposure lower, and code and SQL injection too high on the Base Score (Holm & Afridi, 2015).

Holm also published another article related to scoring of vulnerabilities. This article presents a statistical analysis of how eighteen security estimation metrics based on CVSS data correlate with the time-to-compromise of 34 collected by studying network traffic logs, attacker logs, observer logs, and network vulnerabilities (Holm, Ekstedt, & Andersson, 2012). Holm et al. (2012) study the quality of the multiple vulnerabilities aggregated into the individual score that is used today. The article stated that by itself, CVSS data does not accurately portray the time-to-compromise of a system (Holm et al., 2012). The models used were six weakest link models, a vulnerability exposure model, eight metrics comprising the number of existing vulnerabilities, VEA-ability, Lai and Hsia's model, and the model by McQueen (Holm et al., 2012). The most relevant model to this paper from the article was the vulnerability exposure model:

$$T = \sum_{i=1}^{N} (t - Ti),$$

where *N* is the number of open known vulnerabilities that apply to a system, *Ti* is the discovery date of the vulnerability *i*, *t* is the current date, and *T* is the total number of vulnerability days (Holm et al., 2012). In Holm et al.'s article the time to compromise (TTC) of a system was the primary object of the comparing the six models, but most relevant to this paper is the use of temporal metrics to prioritize vulnerabilities and give some vulnerabilities due to time of detection a higher CVSS score.

Another work with a focus on data mining, clustering and classification was authored by Li and Ye (2006). This paper used a data mining algorithm based on supervised clustering to lean data patterns for data classification (Li & Ye, 2006). The article addressed the issues with a few clustering techniques such as K-means, incremental, density based, and grid-based clustering algorithms versus the clustering and classification algorithm-supervised (CCAS) that the authors proposed (Li & Ye, 2006). The CCAS address the both the distance measurement issue with K-means and the problem with the other clustering algorithm of only using the last data cluster as the starting point (Li & Ye, 2006). The contribution from Li of the classification of data points into multiple target classes relates to this paper (Li & Ye, 2006). Although, classification is detailed in this article, the use of KNN separates Li's article from this paper.

Alguliev work on classification textual e-mail spam using data mining offered the use of the genetic algorithm and KNN for clustering of e-mail documents (Alguliev, Aliguliyev, & Nazirova, 2011). The article documented the impact spam has on productivity, mailbox space, viruses, and the time to delete the unwanted

correspondences (Alguliev et al., 2011). Before applying any clustering method, analysis of the messages was completed using the following weighting equation:

$$w_{ij} = n_{ij} log\left(\frac{n}{n_j}\right),$$

where $s_i = \{w_{i1}, \ldots, w_{im}\}$, $s = \{s_1, \ldots, s_n\}$ = collection of spam messages in vector space $= \{t_1, \ldots, t_m\}$ = set of terms (spam keywords), m = number of terms, w_{ij} = weight of term t_j in spam message i, $i=1 \ldots n$, $j = 1, \ldots m$, n = number of spam messages in collection (Alguliev et al., 2011).

In the aforementioned equation, n_{ij} = frequency of appearance of term t_j in spam message s_i and nj is the number of spam messages containing the term t_j. This type of weighting and frequency determination is similar to what is used in this paper, although this paper did not use a clustering algorithm. Additionally, the article moves into the classification of the e-mails by using KNN, followed by a knowledge extraction from classes using summarization technique (Alguliev et al., 2011). Multidocument summarization method is applied for knowledge extraction from clusters (Alguliev et al., 2011). The representativeness of a sentence is defined by similarity measure between them and corresponding cluster centroid, that is, the less Euclidean distance between the sentence and corresponding cluster centroid means the sentence is more representative (Alguliev et al., 2011). This article particularly assisted in providing a frequency formula for this paper.

Neural Networks and Cybersecurity Scoring

The work by Abdelmoula (2015) discusses how credit risk is evaluated by using k-nearest neighbor classifier in the case of Tunisian banks. The article defines credit risks

and how in the time the article was written, the new methods banks are determining if a borrower is a risk to not pay back their obligation (Abdelmoula, 2015). The article details a study to answer the question of default prediction of short term loans for a Tunisian commercial bank (Abdelmoula, 2015). The data used in the study was from Tunisian firms during the timeframe of 2003 to 2006 (Abdelmoula, 2015). K-nearest neighbor algorithm was used in the results to obtain an 88.63% (for k = 3; Abdelmoula, 2015). The KNN used in this model is similar to this paper in that a classifier algorithm was used, but the PNN was used as a classifier for text mining purposes in this paper. Additionally, this article did not discuss text mining and CVSS scoring, but was useful in establishing a risk methodology. To set up the experiment, Abdelmoula built three types of KNN classifiers:

1. Uses data on financial ratios (cash-flows excluded)—Non cash flow model
2. Uses data on all ratios—Cash Flow model
3. Uses all indicators of the study—Full information model

The conclusion assessed that commercial banks that grant borrower loans need consistent models that can correctly detect and predict defaults (Abdelmoula, 2015). The overall impetus of the paper was to assist the banks in making consistent decisions using a validated model.

An article published by Awan proposes a framework for measuring temporal variance in computer network risk. The approach is validated using data from a University network, with a collection of 462,787 instances representing threats measured over a 144 hour period (Awan & Rana, 2016). According to the article, the framework enables risk scores to be produced at various levels within the network over time,

underpinning a dynamic probabilistic risk assessment model, which is then combined to represent the relative risks within a network (Awan & Rana, 2016). A hotspot map is then created to show the risk areas (Awan & Rana, 2016). The work details 5 main contributions:

1. Proposing risk metrics which could be calculated more objectively, rather than using subjective, human-based qualitative inputs to identify cyber risk hotspots in a computer network
2. Modelling temporal risk behavior of software applications and understanding the effectiveness of security policies
3. Devising mechanism for alerting network administrator to take precautionary measures before the risk score significantly increases
4. Identifying cyber risk hotspots emerging over a period of time in a computer network
5. Investigating causes of emerging cyber risk hotspots associated with a particular software application (Awan & Rana, 2016).

The risk scoring framework in this work is made up of the elements in Figure 7.

1. Software, $S = \{\alpha_1, \alpha_2, \alpha_3, \cdots, \alpha_i\}$
2. Threat Category, $H = \{\Lambda_1, \Lambda_2, \Lambda_3, \cdots, \Lambda_j\}$
3. Threat, $\Lambda = \{\lambda_1, \lambda_2, \lambda_3, \cdots, \lambda_k\}$
4. Severity, $R = \{sev_1, sev_2, sev_3, \cdots, sev_l\}$

Figure 7. Elements of risk scoring (Awan & Rana, 2016).

There are related works identified that discuss a Bayesian network risk management framework, another about attack graphs, a survey of risk assessment and network Intrusion Response Systems (Awan & Rana, 2016). The article details the results of the approach by stating that externally hosted software applications, such as Steam and Google-play are leading to threats from malicious files, which require a different type of threat management from the internally hosted software applications, such as DNS, where the software application itself is proving to the key vulnerability (Awan & Rana, 2016). The article provided a snap shot of several days and showed how most pertinent threats fluctuate and alter over time. The results of the data showed 99.45% of malicious traffic targeting 14 software applications within the identified University network. This work contains the risk framework, but it did not detail the textual data mining and PNN established in this paper.

The next related work focused more on the use of the genetic algorithm (GA) to assist in decision making for credit scoring. Khanbabaei and Alborzi (2013) layout decision tree modelling that propose to classify customers optimally using a combination of clustering, feature selection, decision trees, and genetic algorithm techniques. The authors use GA to select appropriate features and build optimum decision trees in credit scoring of bank customers (Khanbabaei & Alborzi, 2013). The dataset from the article is collected from Bank Mellat of Iran, which contains about 5173 cases of individual consumers' credit data for the first three months of 2003 (Khanbabaei & Alborzi, 2013). There are three categories of customers in the target feature:

1. Customer (1): They have paid back all of their credit facilities.
2. Customer (2): Three months have passed from the maturity date of their credit facilities.
3. Customer (3): They have nonperforming credit facilities of more than six month (Khanbabaei & Alborzi, 2013).

The proposed method in the article is close to the method in this paper due to the thinning of data in the early steps, the proposed method for classification is outline in Figure 8.

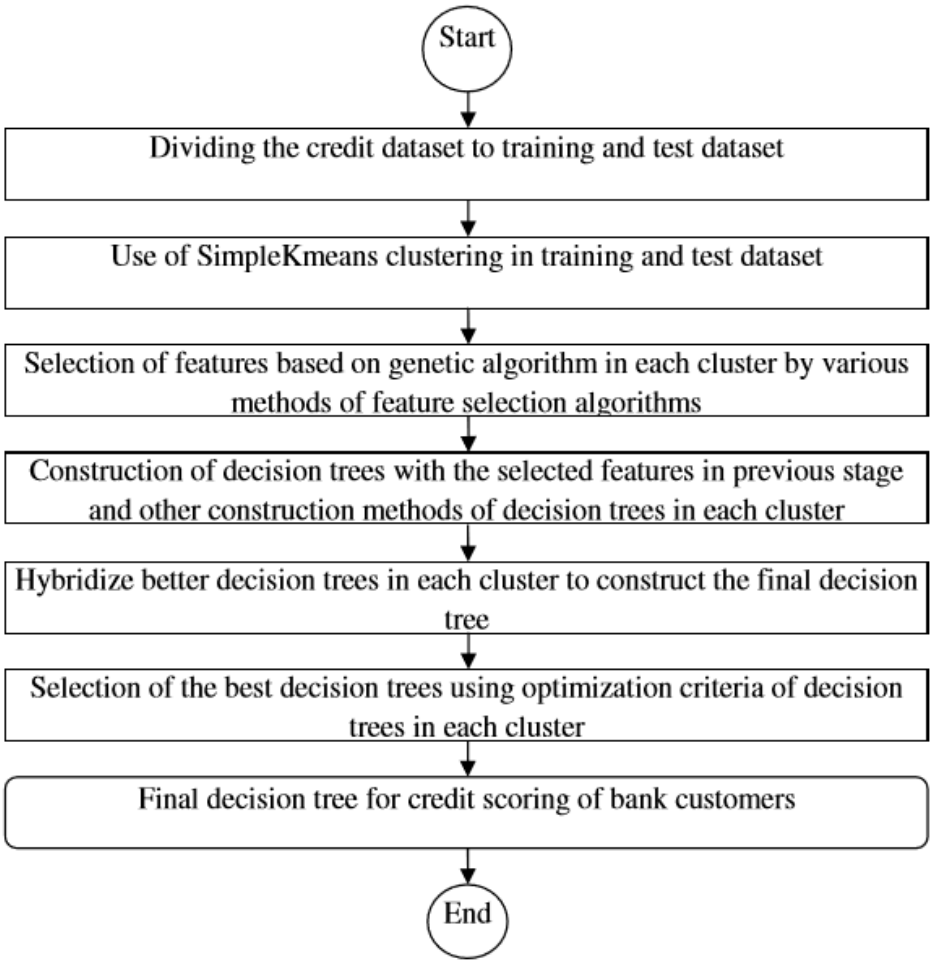


Figure 8. Steps for the proposed hybrid decision model (Khanbabaei & Alborzi, 2013).

The results from the article are in Figure 9.

Classification Models	Naive Bayes	KNN classifier (K=2)	CHAID tree	Random Forest classifier	multilayer perceptron (MLP)	sequential minimal optimization (SMO)	logistic regression
Classification Accuracy (Correctly Classified Instances)	84.99%	95.5%	80.5%	95.8%	93.6%	82.25%	83.32%

Figure 9. Results of the hybrid decision model (Khanbabaei & Alborzi, 2013).

The review of this article points to more reliable decision models for banks to possibly impose. This paper does emphasize the use to a decision algorithm to assess risk decisions at a lower level similar to the article reviewed, but with use of the PNN and text mining approaches.

Another related work was similar to the previous work discussing credit scoring, but using Branch and Bound (B&B) method along with an SVM classification. The authors of this work focused on dynamic incremental modeling for credit scoring versus a static model (Sun, Li, Chang, & Huang, 2015). Sun and Li introduce the concept of drift in credit scoring that highlight many actual tasks change over time (Sun et al., 2015). This is an important concept in this dissertation, because of the temporal aspects discussed in the article that task will likely change over time. The article states that it uses SVM due to the small sample size and the better performance, generalization, and modeling than traditional neural networks (Sun et al., 2015). Ensemble modeling was also detailed in the article to show how integrating multiple models can eventually lead to one overall model (Sun et al., 2015). The model proposed in the article followed these steps:

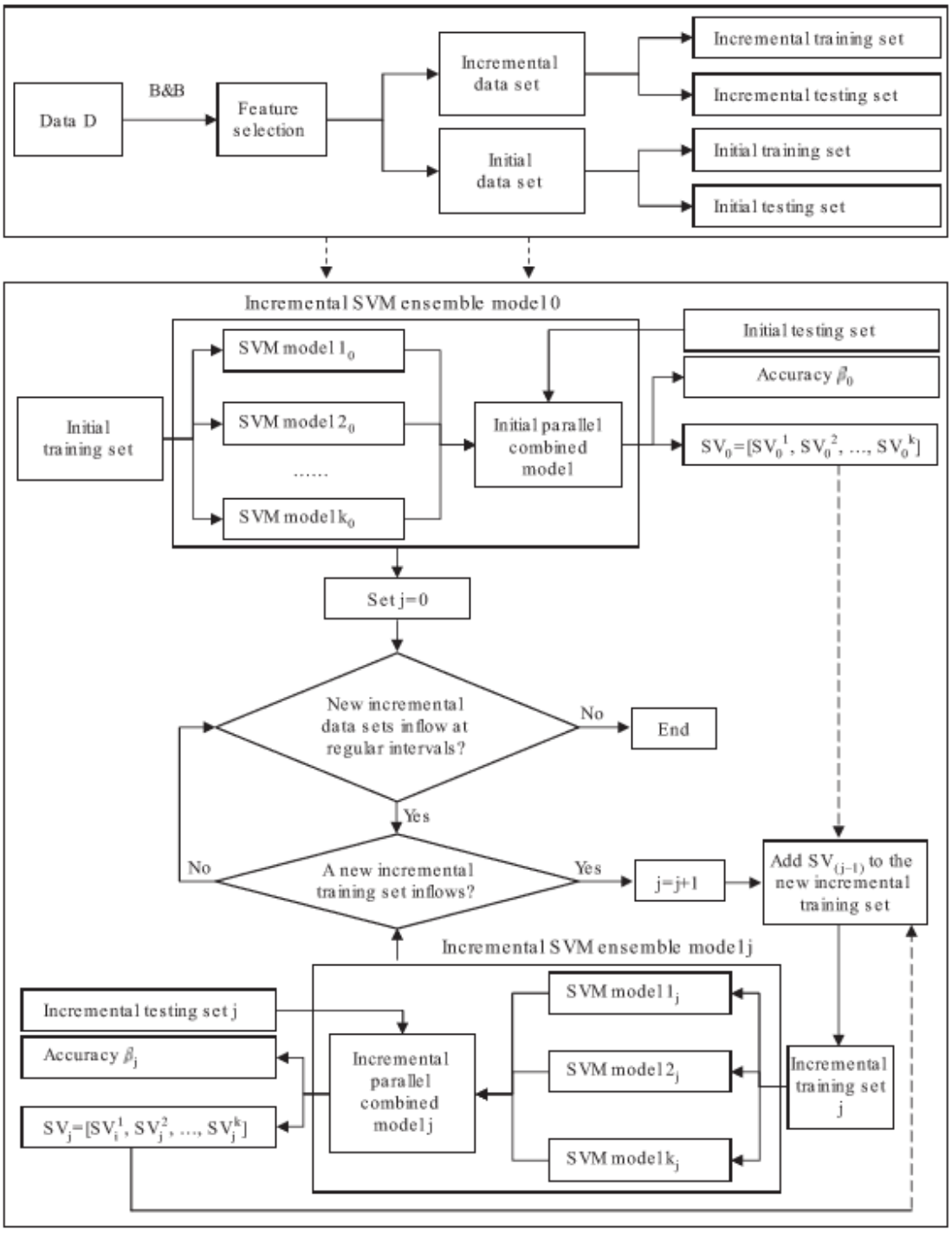


Figure 10. Incremental SVM modeling for credit scoring (Sun et al., 2015).

The results of the article's study are showed in Figure 11.

	Averageevaluationaccuracy(%)			
	ThenewincrementalSVM ensemblemodel	Thenon-incrementalSVM ensemblemodel	Absolutedifference values(%)	Differencepercentages
TestingsamplessetIIITotalIIITotalIIITotalIIITotal				
TE_1	83.3384.6784.0083.3384.6784.000.000.000.000.000.000.00			
TE_2	78.5785.2481.9172.8680.0076.435.715.245.4821.0526.1923.23			
TE_3	70.8385.0077.9267.5083.7575.633.331.252.2910.267.699.40			
TE_4	78.3388.7583.5470.0089.5879.798.33		Š0.833.7527.78	Š8.0018.56

Figure 11. Performance results of proposed model and the standard model (Sun et al., 2015).

The related article by Sun addresses the temporal elements that should be addressed in cybersecurity modeling which is proposed in this paper.

The next related work discusses KNN and the use of it to address power analysis attacks. This article by Martinasek focuses on comparing machine learning algorithms for power analysis (Martinasek, Zeman, Malina, & Martinasek, 2016). The example of successful attacks used in the article was against trusted cryptographic devices such as RFID (Radio-Frequency Identifications) and contact smart cards (Martinasek et al., 2016). The datasets used were unprotected AES (Advanced Encryption Standard) and independently created public available power traces corresponding to a masked AES implementation (Martinasek et al., 2016). The machine learning models compared in this paper were SVM, RF (Random Forest), Multi-Layer Perceptron (MLP), and KNN. The main questions addressed in this work were

- Which ML algorithm is the most suitable for profiling PA attacks?
- Are there any generally appropriate settings of the ML algorithms that can be used by the potential attacker for PA attacks?

- How big is the influence of the number of power traces and interesting points on the classification results of individual ML algorithms? (Martinasek et al., 2016).

The questions do not pertain to this paper, but the use of SVM, RF, and KNN was an approach that was considered for this paper, but not implemented within the algorithm.

Specht (1988) work on PNN laid the foundation for understanding how a PNN can be applied within this paper. Although the related work did not pertain to cybersecurity overall, the work did state that a four-layer neural network can map any input pattern to any number of classifications (Specht, 1988). The article details the Bayes strategy for pattern classification, where Specht states that "an accepted norm for decision rules or strategies used to classify patterns is that they do so in such a way to minimize the 'expected risk'" (p. 1). The work talks about training of the network by setting each X pattern in the training set equal to the W_i weight vector in one the pattern units, and then connecting the pattern unit's output to the appropriate summation unit as shown in Figure 12.

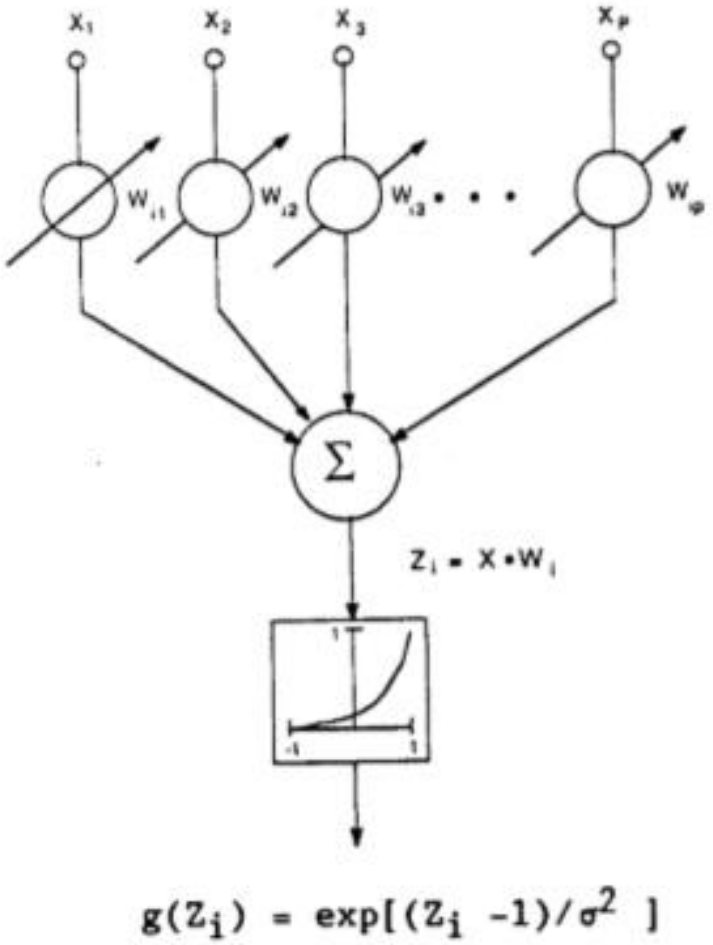


Figure 12. The pattern unit (Specht, 1988).

In the Specht (1988) work, the commonly used Sigmoid Activation Function is replaced by the following nonlinear operation:

$$\exp\left[\frac{Z_i - 1}{\sigma^2}\right]$$

The probabilistic neural network proposed in the work is shown to be used for mapping classification, associative memory, or the direct estimation of a posteriori probabilities.

Proactive Cybersecurity

Although not stressed heavily in this paper, proactive cybersecurity is an inherent element for the cybersecurity and textual data mining algorithm discussed in this paper. An article by Craig, Shackelford, and Hiller (2015) that brings proactive cyber security to the forefront, stressing that standard network security practices cannot stop breaches such as Sony had in 2015. The article outlines the evolution of cybersecurity in a global legal environment (Craig et al., 2015). The article talks about the elements that commercial firms are using to advance proactive cybersecurity:

1. The general trend toward private security and growing awareness of cyber insecurity;
2. The unique nature of cybersecurity (with infrastructure that is often privately owned and for which private sector expertise dominates);
3. The move toward bottom-up regulatory frameworks—in the vein of the 2014 National Institute for Standards and Technology Cybersecurity Framework (Craig et al., 2015).

The article discusses the activities associated with proactive cybersecurity displaying in Figure 13.

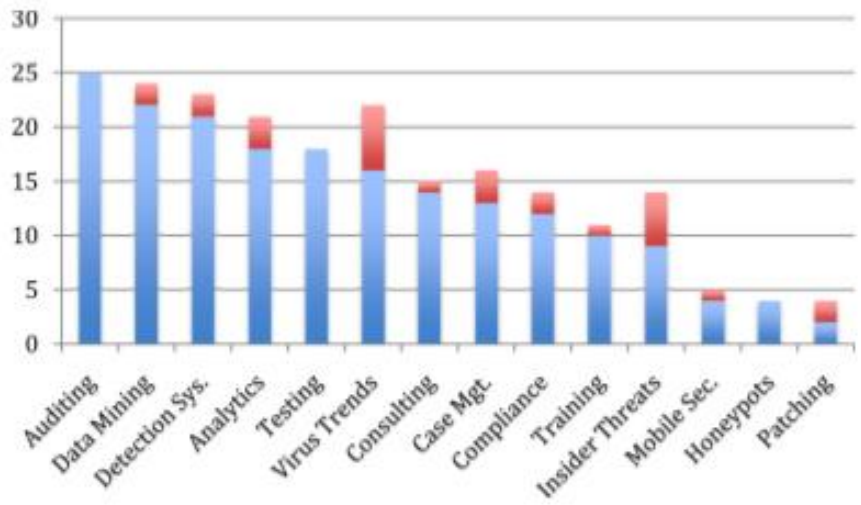


Figure 13. Current proactive cybersecurity practices (Craig et al., 2015).

The article emphasizes that the National Institute for Standards and Technology Cybersecurity Framework can assist firms in the establishment of proactive cyber security practices to slow or even stop determined attackers targeting networks (Craig et al., 2015). This article does not tackle the quantitative elements of textual mining and PNN, but it does contribute to the overall intent of the paper by allowing for better cybersecurity practices and quicker decision analysis.

An article about a study focusing on the prediction of cyber risks through the NVD applied data mining techniques on the NVD data with the purpose of predicting the time to the next vulnerability for a given software application (Zhang et al., 2015).The Zhang et al. (2015) experimented with various features constructed using the information available in the NVD and applied various machine learning algorithms to evaluate the predictive power of the data. The article used TTNV (time to next vulnerability), version difference, software name, and CVSS as the predicted features (Zhang et al., 2015). The

prediction model used the month, day, Versiondiff, TTPV (Time to Previous Vulnerability), CVSS metrics (indicate the properties of the predicted vulnerabilities) as inputs for the model (Zhang et al., 2015). The output of the model is TTNV which implies the risk level of zero-day vulnerabilities. These features were also used as a part of the model used in this paper for examples of features that may drive a decision on a textual document. The article used regression functions such as linear, least median square, multilayer perceptron, radial basis function (RBF) network, sequential minimal optimization (SMO) regression, and Gaussian processes (Zhang et al., 2015). None of machine learning techniques in this article involved the use of PNN, but the feature extraction was useful for textual classification.

In a 2009 report by Nikolić and Ružić-Dimitrijević, the authors found that risk to computers came in terms of losses or damage The article reported that protection of computers apply to confidentiality, integrity, and availability (Nikolić & Ružić-Dimitrijević, 2009). The article's main purpose was to present methodologies of risk management in the information technology area (Nikolić & Ružić-Dimitrijević, 2009). The angle taken by the article presented the methodology from the occupational health area in the IT industry (Nikolić & Ružić-Dimitrijević, 2009). Nikolić and Ružić-Dimitrijević presented from a holistic point of view of risk and did not address the foundation decision making that leads to organization risks. The frequency/probability occurrence model shown in Figure 14 from the article was very subjective, but did lead to more to how risk should be classified.

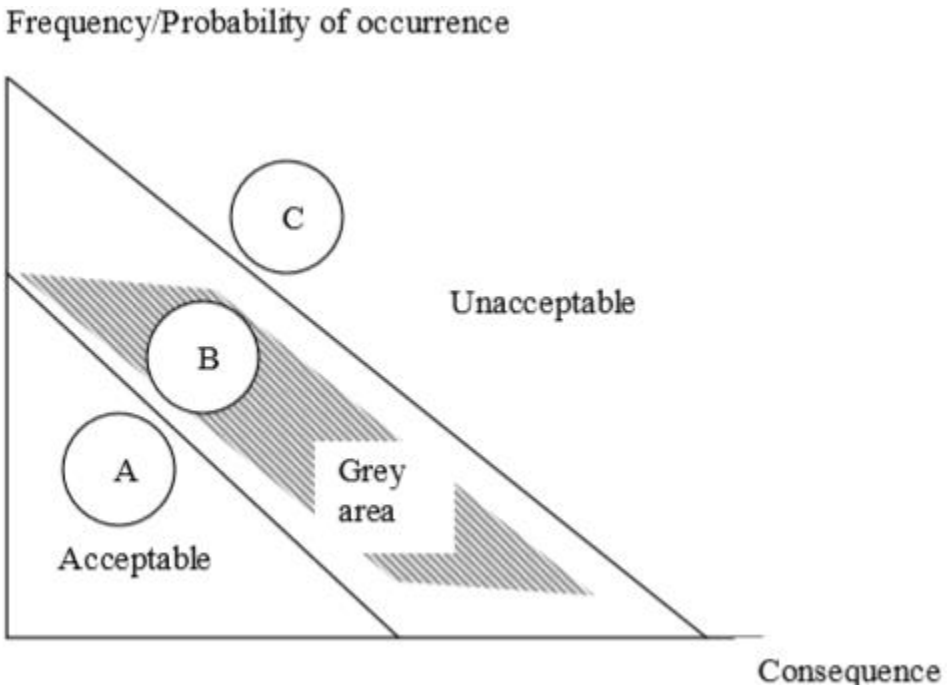


Figure 14. Evaluation of risk (Nikolić & Ružić-Dimitrijević, 2009).

Event A has both low values, and risk is acceptable as far as it is under the limits. Event C is above the limits with high frequency and huge consequence (Nikolić & Ružić-Dimitrijević, 2009). It is unacceptable, and it needs some measurements to reduce consequence and/or probability (Nikolić & Ružić-Dimitrijević, 2009). For event B, which is in grey zone between the limits, it is hard to make decision (Nikolić & Ružić-Dimitrijević, 2009). This work was not fully quantitative, but it did lend to the basic classification technique needed for a quick decision by a system administrator. The frequency of exposure and frequency outlined in the methodology in this article indirectly correlated with the use of frequency as a measurement in this paper's methodology for automated assistance in accepting, denying, or quarantining of a textual document.

Information security risk was again addressed in a work by Shameli-Sendi, which presented taxonomy of security risk assessment drawn from 125 papers published from 1995 to May 2014 (Shameli-Sendi, Aghababaei-Barzegar, & Cheriet, 2016).The article discusses the key features of risk assessment that should be included in an information security management system (Shameli-Sendi et al., 2016). The article discusses that information security is a continuous process which gives businesses an understanding of potential risks to organizational IT assets and tools necessary to evaluate these risks (Shameli-Sendi et al., 2016). The article broke risk analysis into the diagram shown in Figure 15.

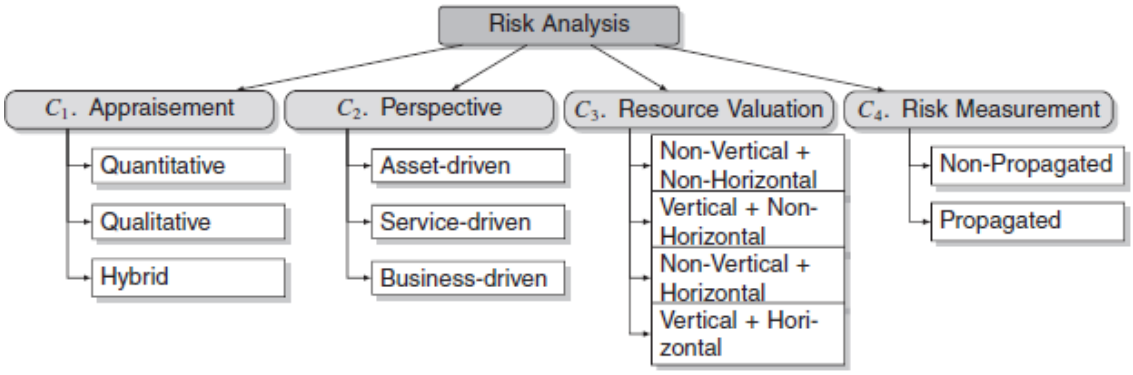


Figure 15. A taxonomy of information security risk assessment approaches (Shameli-Sendi et al., 2016).

Although the authors of the article focus on an overall risk approach, but the risk measurement evaluation is done in a nonpropagated way of multiplying business process value, vulnerability effect, and threat effect (Shameli-Sendi et al., 2016). Additionally, there was a propagated method of backward and forward impact of vulnerability (Shameli-Sendi et al., 2016).The propagated method assist in the prediction of potential

damage (Shameli-Sendi et al., 2016). This article does provide value in regards to showing the different aspects of risk assessment, but it does not show a quantitative process using PNN or any neural networks to facilitate a decision for textual document acceptance.

A review of the Global Technology Audit Guide (GTAG) yielded how internal auditors view risk, control, and governance issues surrounding technology (Mar, Johannessen, Coates, Wegrzynowicz, & Andreesen, 2012). This document gave overall high level risk strategy and how important IT controls are in either accepting, eliminating, sharing, or controlling/mitigating risk (Mar et al., 2012). This work did not contribute heavily to the paper, but laid out an overall understanding of how risk should be addressed within an organization.

Gupta and Lehal (2009) survey text mining techniques and applications in their paper detailing how to link extracted information together is another related work to this paper. The article points out a key difference between text mining and data mining, citing that data mining tools are designed to handle structured data from databases, but text mining can work with unstructured or semistructured data sets to include full-text documents (Gupta & Lehal, 2009). Another key relation to this paper the article brings into focus is the keyword extraction layout shown in Figure 16.

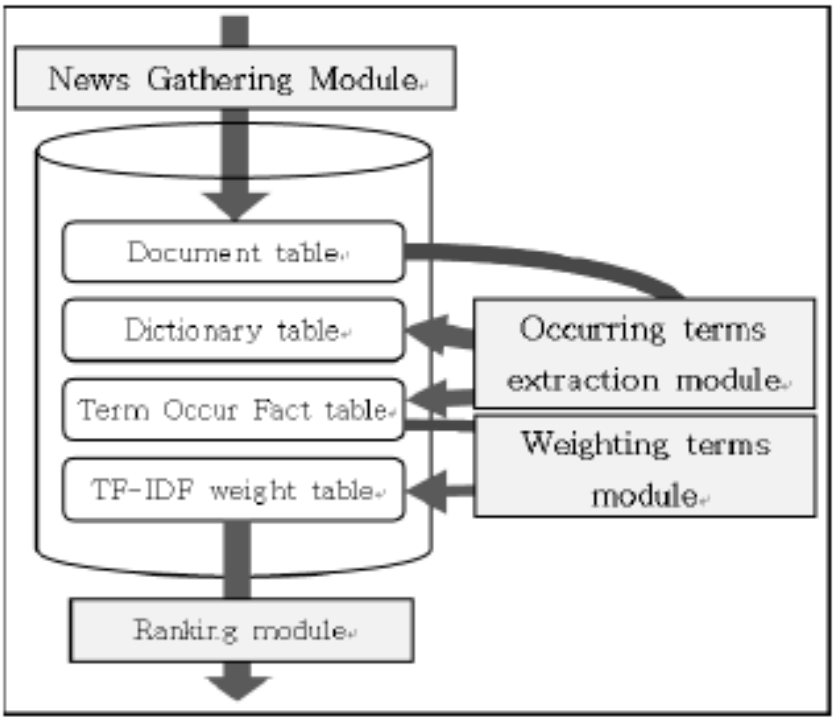


Figure 16. Keyword extraction(Gupta & Lehal, 2009).

This method relates to the previous work as it includes the weighing of term frequency inverse document frequency (TF-IDF; Gupta & Lehal, 2009). The method outlined in this work follows a similar methodology to this paper to in order to assist the decision maker to accept, deny, or quarantine the document:

- Feature extraction
- Text-based navigation
- Search and retrieval
- Categorization (supervised classification)
- Clustering (unsupervised classification)
- Summarization (Gupta & Lehal, 2009).

The main difference between the aforementioned text mining operations compared to this paper is that the classification is done on a probabilistic solution.

A related work from A. K. Singh (2011) was a presentation about PNN. The presentation detailed the origins of the PNN from Specht in the early 1990s (A. K. Singh, 2011). The work presented a detailed explanation of the PNN architecture, the advantages of PNN, applications, and a basic example using the algorithm (A. K. Singh, 2011) The work addressed the fast training process that is an advantage to using a PNN (A. K. Singh, 2011). Conversely, the large memory requirements were discussed as a disadvantage (A. K. Singh, 2011). The presentation contributed heavily to the architecture laid out in this paper.

Another presentation regarding PNN was authored by Cheung and Cannons (2002) which was a more mathematical and detailed explanation (Cheung & Cannons, 2002). The presentation described estimating the probability distribution function (pdf) for the PNN using the samples of the training set or populations (Cheung & Cannons, 2002). The estimate for a single sample:

$$\frac{1}{\sigma} W\left(\frac{x-x_k}{\sigma}\right),$$

where x = unknown (input), xk = "kh" sample, W = weighting function, σ = smoothing parameter (Cheung & Cannons, 2002). According to Cheung and Cannons, for a single population, the estimate for the pdf is

$$\frac{1}{n\sigma} \sum_{k=1}^{n} \frac{1}{\sigma} W\left(\frac{x-x_k}{\sigma}\right)$$

(average of the pdf's for the "n" samples in the population)

The aforementioned equations serve as the basis for the PNN and are part of the coding used in this paper.

Abikoye, Omokanye, and Aro (2018) survey overall data mining approaches to text classification in their paper. The article discusses the issue of unstructured data on the World Wide Web which contributes to the difficulty of textual data mining in today's environment (Abikoye et al., 2018). The relevance of text mining described in the article is the primary reason for this paper. In Abikoye's article, several definitions of text mining are discussed but the one directly relevant to the this paper is the ability "to extract meaningful and useful information from the increasing available data so as to under facts underlying such data and thus make a good decision for the betterment of society" (Abikoye et al., 2018, p. 1). The article outlines a classification process from Kotsiantis:

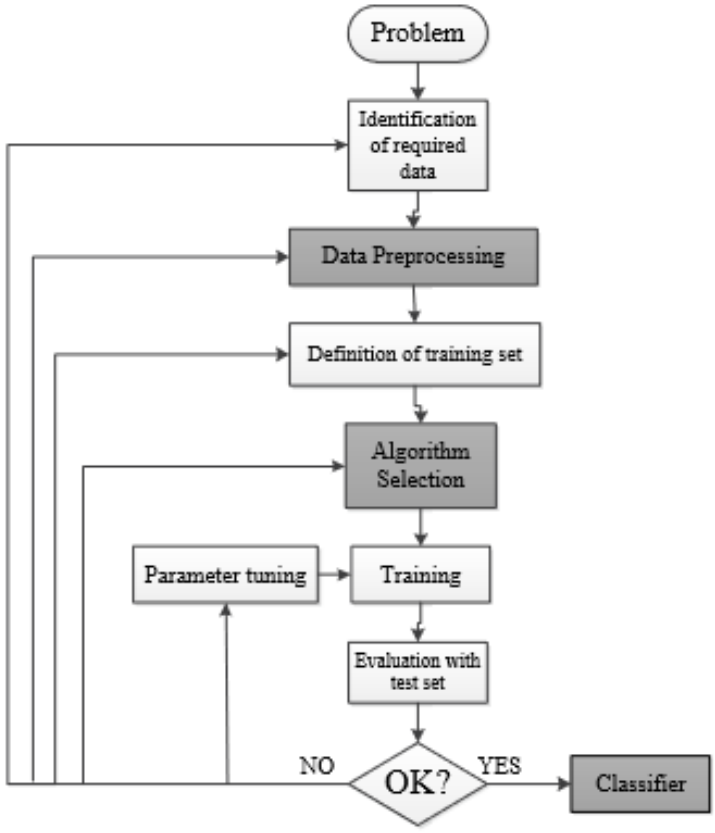


Figure 17. Text classification process (Abikoye et al., 2018).

The article follows up with a survey of machine learning algorithms for text classification such as Naïve Bayes, KNN, SVMs, Artificial Neural Networks (ANN), and decision trees (Abikoye et al., 2018). The article also discusses evaluating an algorithm's performance when determining its usefulness in classification (Abikoye et al., 2018). The most common measures for effectiveness of a classifier are precision, recall, F-measure, accuracy, and error, these are shown in the following equation. This article did not detail the PNN, but provided an evaluation for the effectiveness of classification algorithms overall.

$$Precision = \frac{\mathrm{TP}}{TP + FP} \times 100$$

$$Recall = \frac{\mathrm{TP}}{TP + FN} \times 100$$

$$F - measure = \frac{2 \times \mathrm{Recall} \times \mathrm{Precision}}{Recall + Precision}$$

$$Accuracy = TP + TN$$

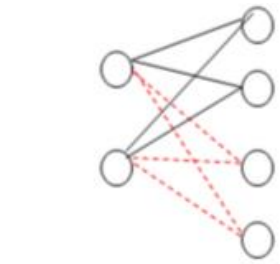
$$Error = \frac{FP + FN}{TP + TN + FP + FN}$$

Figure 18. Classification effectiveness measures (Abikoye et al., 2018). TP = Documents correctly assigned to a category. FP = Documents incorrectly assigned to a category. TN = Documents correctly rejected from a category. FN = Documents incorrectly rejected from a category.

Another related article using the PNN classifier is by Padma and Giridharan (2016), who used the classifier to find and select the texture features of a tumor region. Although this article used PNN, it did not touch on CVSS or actual text documents. This article focused on textual analysis which is a quantitative method that can be used to quantify and detect the structural abnormalities in different tissues (Padma & Giridharan, 2016). This article was not relevant to this paper.

Hewahi (2018) authored a paper on PNN and Hidden Markov Models (HMM) which proposed a theoretical framework based on Probabilistic Neural Network (PNN) concept to represent all HMMs in a given system in one structure. The authored also discussed Hierarchical Hidden Markov Models (HHMM) and the integration of using these Markov models with PNN. Hewahi gave an overview of PNN specifically pointing

out the advantages that PNN don't need training, no local minimum issues, no further training is needed if new instances to be added to the dataset and finally the more examples are provided, the more opportunity to get the optimum solution. The article goes through an example of HMM and PNN with having two visible layers, safe and unsafe, and four invisible layers, safe, unsafe, calm, and chaos (Hewahi, 2018). The article does give a layout of PNN and HMM, but does not discuss textual mining and combining it with PNN or CVSS scores for decision purposes.



Input Layer: Safe, Unsafe

Pattern and output Layer: Safe, Unsafe, Calm, Chaos

Weights between the input layer and pattern layer are: 0.8, 0.2, 0.9, 0.1, 0.4,0.6,0.3,0.7 for the links Safe Safe, Safe Unsafe, Safe Calm, Safe Chaos, UnSafe Safe, Unsafe Safe, Unsafe Calm, Unsafe Chaos respectively.

Figure 19. PNN example with HMM (Hewahi, 2018).

A survey of systems security metrics was conducted by Pendleton, Garcia-Lebron, Cho, and Xu (2016) which was related to this paper based on understanding of attack–defense interactions. These interactions are affected by factors such as degree of system vulnerabilities, power of system defense mechanisms, threat severity, and situations a system at risk faces (Pendleton et al., 2016). The article proposes an ontology relating the metrics detailed earlier to each other (Pendleton et al., 2016). The ontology is

detailed in Figure 20 which emphasizes that vulnerability metrics contribute to situation metrics (see Table 2, Pendleton et al., 2016).

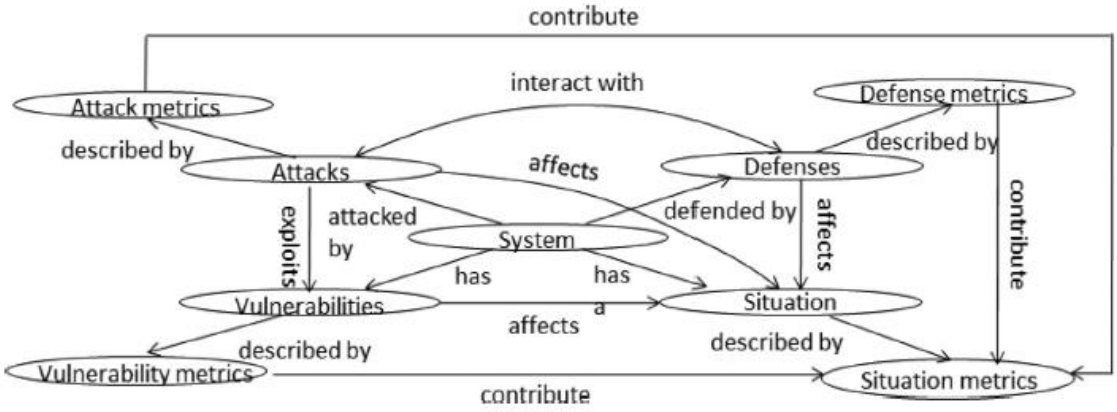


Figure 20. A high-level ontology of systems security metrics consisting of four metrics (Pendleton et al., 2016).

One of the main contributions of this text was the metrics for measuring temporal attributes of vulnerabilities and how the CVSS score is used as a metric probabilistically (Pendleton et al., 2016). A metric that directly relates to this paper is the measurement of vulnerability lifetime pertaining to the client-end vulnerabilities that generally take a long time to patch all infected devices or infeasible to patch all (Pendleton et al., 2016). Additionally, the article discussed probabilistic severity metrics using CVSS scores (Pendleton et al., 2016). The likelihood of exploitation metric used in the Temporal CVSS score was part of what the article determined to be a metric of if an exploit will be executed by an attacker over time (Pendleton et al., 2016).

Table 2

Metrics and Measurement of Vulnerabilities, Defenses, Attacks, and Situations

(Pendleton et al., 2016)

	Vulnerability metrics	Defense metrics	Attack metrics	Situation metrics
Measurement	vulnerabilities of an enterprise system $V(t)$, or a computer system $_{vi}(t)$	strength of defense mechanisms $D(t)$	strength of attacks $A(t)$	*Situation*(t), including system's security $S(t)$
Target	an enterprise system $C(t)$ or computer system $c_i(t)$	defense mechanisms $D(t)$ employed at $C(t)$ or $c_i(t)$	attacks $A(t)$ against $C(t)$ or $c_i(t)$	Evolution of situation and environment
Types	users' vulnerabilities; interface-induced vulnerabilities; software vulnerabilities	preventive, reactive (e.g., detection), proactive (e.g., Moving-Target Defense (MTD)), and overall defense strength	zero-day attacks, targeted attack, botnets, malware spreading (e.g., infection rate), and evasion techniques	security state, security incidents, security investment
Reference Figures	Figures 1(a) and (b), Figure 2	Figure 1(c) and Figure 2	Figure 1(d) and Figure 2	Figure 1(e) and Figure 2

The newest source for CVSS is v3.0 which has a statement related to the Temporal metric. The v3.0 states that "the influence of Temporal metrics has been reduced in v3.0, relative to v2.0. In CVSS v3.0, the term Exploitability has been renamed to Exploit Code Maturity to better represent what the metric is measuring" (FIRST.Org, 2018, p. 6). The v3.0 mainly focused on changes to the Base score (FIRST.Org, 2018).

Rees, Deane, Rakes, and Baker (2011) discuss decision support for cybersecurity risk planning and the tradeoffs necessary for proper risk determination. The article acknowledges that attackers are using a variety of methods to try to infiltrate information

systems and a tremendous about of effort has been devoted to technological tools to blocking software, mitigate software, or secure communication protocols from damaging incursions (Rees et al., 2011). Rees et al. details current approaches to IT security and risk such as checklist which decision makers uses to develop a coverage strategy. There has been a lot of qualitative approaches to assess risk such as OCTAVE and Value Focused Thinking (VFT; Rees et al., 2011). Additionally, the article discusses a quantitative risk analysis approach using the following formula:

$$R = \sum_i E_i L_i(C),$$

where R = risk in dollars per year, i = index representing the different threats facing the firm, E_i = the expected number of security events of type I per year, $L_i(C)$ = expected dollar loss caused by security event I given the current set of countermeasures C (Rees et al., 2011). The aforementioned calculation of the risk uses the risk as the product of threat occurrences and their resultant losses in dollars/event (Rees et al., 2011). The article also details calculating (fuzzy) system risk with the process in Figure 21 (Rees et al., 2011). System (fuzzy) risk is using a genetic algorithm to evaluate the risk determines the threat to the system(Rees et al., 2011). This is a similar overall approach to this paper, but the risk is determined by a PNN for textual documentation.

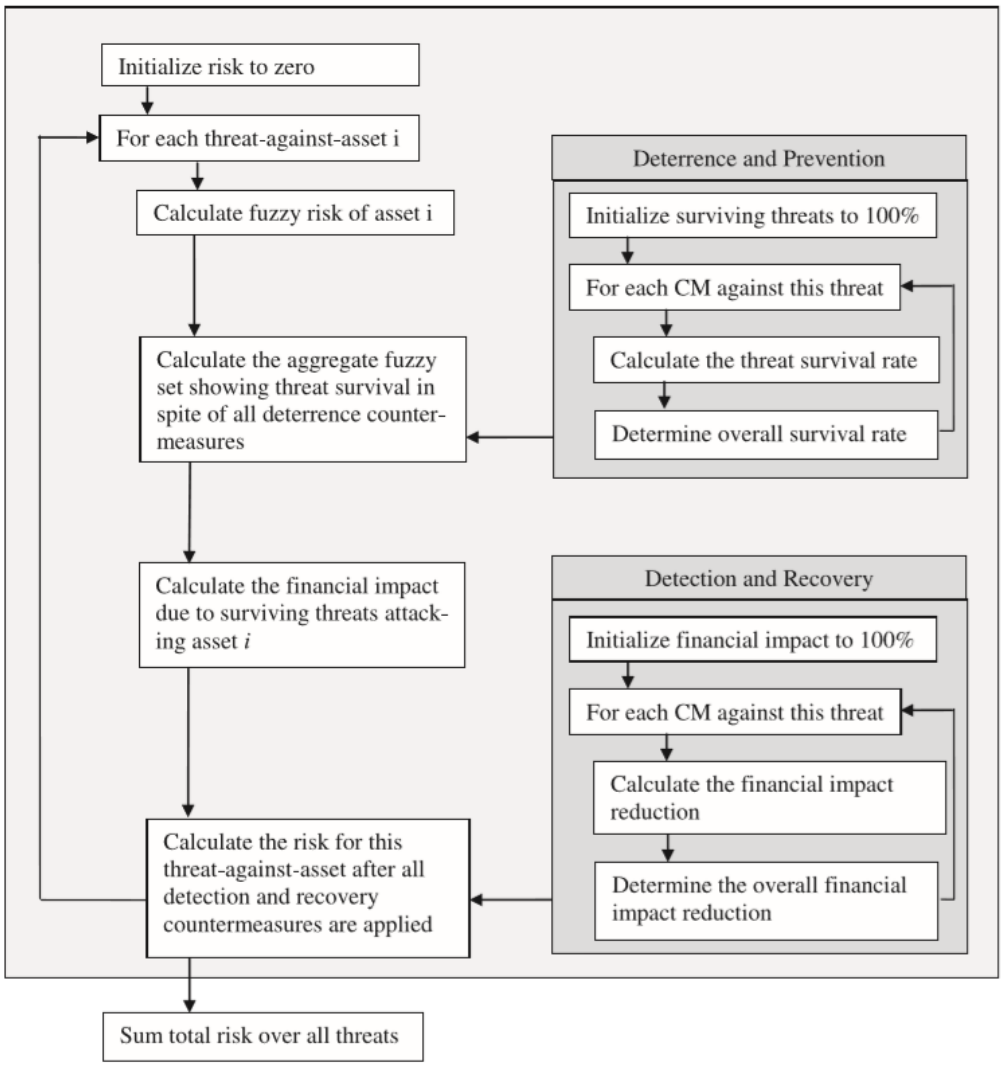


Figure 21. Steps in calculating (fuzzy) system risk (Rees et al., 2011).

Tran, Nguyen, Tsai, and Kong (2011) published an article on the Network Security Detection problems in which predictive models are constructed to detect network security such as spamming. Tran et al. proposes a Booted Subspace Probabilistic Neural Network (BPNN) to assist with detecting the evolving cyber threats that are part of the current cyber landscape. The articles identifies network security detection problems which compromise the integrity, confidentiality or availability of the network,

such as unauthorized access, boosting detection and lessening false alarms, and spamming (Tran et al., 2011). The proposed algorithm is a modification of a Radial Basis Function Neural Network (RBFNN) with an adaptive boosting technique (Tran et al., 2011). The RBFNN is stems from the Vector Quantized-Generalized Regression Neural Network (VQ-GRNN). Compared with the original GRNN which incorporates every training data point into its structure, VQ-GRNN only applies on a smaller number of clusters of input data (Tran et al., 2011). To make the VQ-GRNN suitable for NSD problems, i.e. enhancing its accuracy, Tran propose the BSPNN which combines VQ-GRNN and AdaBoost technique (Tran et al., 2011). AdaBoost is originally designed for two-class classification and therefore not directly applicable to multiclass problems, but the article adopts the Stagewise Additive, Modeling using Multiclass Exponential Loss Function (SAMME) algorithm to make the AdaBoost technique work with the RBFNN and the BSPNN. Figure 22 is the modified design for the BSPNN.

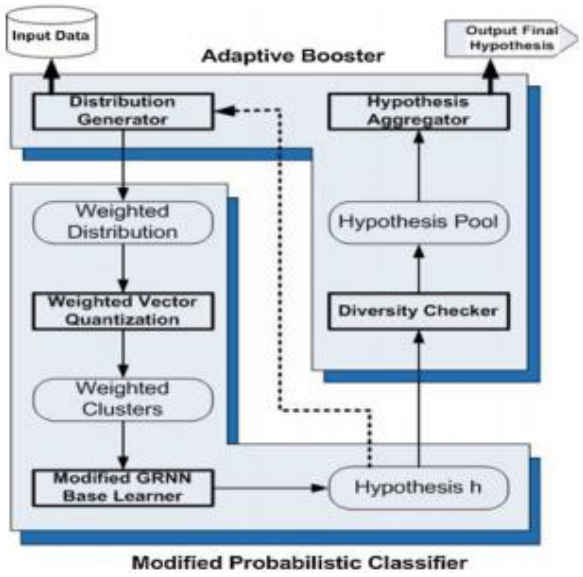


Figure 22. BSPNN high-level design view (Tran et al., 2011).

The article uses the following algorithm as the base hypotheses for its training set:

ALGORITHM 1 Adaptive Booster

Input: $\mathbf{S} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$ and associated distribution $\mathbf{W}$

Initialize $\mathbf{W_i^{(1)}} = \frac{1}{N}$ for all $i = 1 \ldots N$, $\boldsymbol{\alpha}^{(1)} = \mathbf{1}$

Do for $t = 1 \ldots T$

Generate base classifiers (*)

Train a classifier on the weighed sample $\{S, W^{(t)}\}$ using the Modified Probabilistic Classifier and obtain hypothesis $h^{(t)}: x \rightarrow [0, 1]^K$

Compute Kohavi–Wolpert variance ($\boldsymbol{\alpha}^{(t)}$) of current ensemble

$$\boldsymbol{\alpha}^{(t+1)} = \frac{1}{N.L^2} \sum_{j=1}^{N} l\left(x_j\right)\left(L - l\left(x_j\right)\right)$$

Where L and $l\left(x_j\right)$ are the number of base classifiers generated so far in the ensemble and the number of classifiers that correctly classifies x_j. We have $L = t$.

Compute class probability estimates

$$\mathbf{C_k^{(t)}}(\mathbf{x}) = (K-1).\left[\log p_k^{(t)}(x) - \frac{1}{K}\sum_{k'=1}^{K} \log p_{k'}^{(t)}(x)\right], k = 1, \ldots, K$$

Where $p_k^{(t)}(x) = Prob_w\left(h^{(t)}(x) = k|x\right)$ is the weighted class probability of class k.

Update weights

$$\mathbf{W}_i^{(t+1)} = W_i^{(t)}.\exp\left[-\frac{K-1}{K}.\log p^{(t)}(x_i).h^{(t)}(x_i)\right], i = 1, \ldots, n$$

Where $p(x_i) = Prob(x_i)$

Renormalize W

$$\mathbf{W}_i = \frac{W_i}{\sum_{j=1}^{N} W_j}, i = 1 \ldots N$$

End for

Output

$$C_{final}(x) = \underset{k}{\operatorname{argmax}} \sum_{t=1}^{T} \boldsymbol{\alpha}^{(t)}.C_k^{(t)}(x)$$

Figure 23. Algorithm for adaptive booster training set (Tran et al., 2011).

As stated earlier, the article uses the BSPNN for spam recognition which includes an experimental setup using 2,412 authentic messages and 481 spam messages with attachments (Tran et al., 2011). The article concludes that the effectiveness of BSPNN was confirmed by its application to Network Security Detection problems (Tran et al., 2011). Experiments on the Ling-Spam e-mail spam dataset showed that the BSPNN approach achieved better performance compared with other well-known detection methods, with low learning bias and improved generalization at an affordable computational cost (Tran et al., 2011). This approach from Tran did use a probabilistic

approach for textual mining, but did not include the CVSS scoring metrics as part of the algorithm.

A related article to this paper is by Aitnouri and Ouali (2010) based on the performance evaluation of two clustering algorithms, elimination of false clusters (EFC) and Akaike's information criteria (AIC; Aitnouri & Ouali, 2010). The EFC algorithm uses k-means to approximate each node, the next step is suppressing the false clusters them may result from the k-means algorithm. Before proceeding with the elimination, a smoothing operation is performed on the histogram using a PNN (Aitnouri & Ouali, 2010). While this operation is not essential in all cases, it greatly increases the robustness of our model to noise (Aitnouri & Ouali, 2010). The authors represented the EFC algorithm in Figure 24.

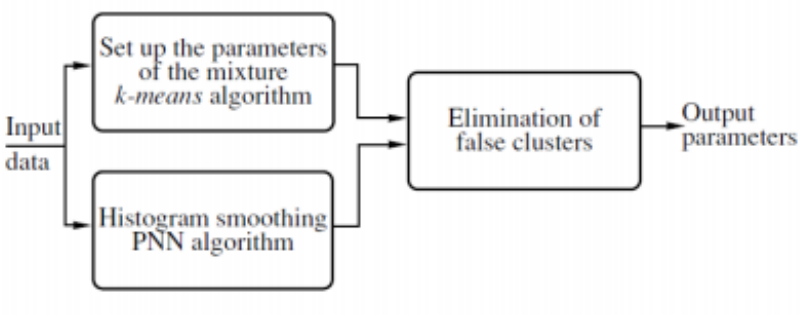


Figure 24. A block diagram of the EFC algorithm (Aitnouri & Ouali, 2010).

The authors used artificial data to show the property of components separation as they are generated according to components overlapping rate (Aitnouri & Ouali, 2010). This article was focused primarily on image data, although it gave an extended use of PNN, the article did not cover the elements of applying PNN to textual data mining and CVSS metrics.

An article by Ruohonen (2017) discussed the time delays between the publication of CVEs and the later publication of CVSS information. The article breaks down CVSS and CVE publication process using the National Vulnerabilities Database (NVD).

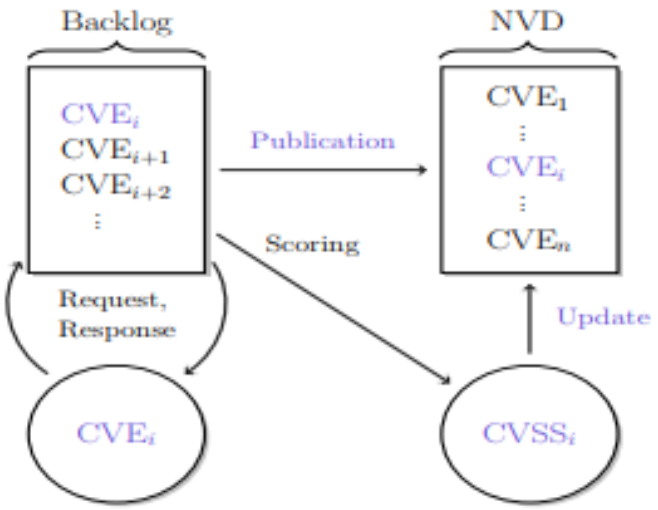


Figure 25. A simplified model for CVSS processing (Ruohonen, 2017).

The article details the process of when security researchers, vendors, and other related personnel request CVEs for vulnerabilities they have discovered or made aware of (Ruohonen, 2017). The MITRE corporation generally maintains the backlog of the CVEs assigned, but has the authority to reject the inclusion of a CVE to the NVD (Ruohonen, 2017). The author states that the structure of the backlog is not officially known, but a simple FIFO (first-in, first-out) might be considered based on the author's research (Ruohonen, 2017). The setup for analysis used by the article is

$$\Delta_i = \tau_{CVSSi} - \tau_{CVEa\ i},$$

given $\tau_{CVSSi} \geq \tau_{CVEa\ i}$ for all i = 1, . . . , n, where τ_{CVSSi} is the timestamp at which the CVSS entry was generated for the i:th CVE that was published at $\tau_{CVEa\ I}$ (Ruohonen,

2017). This time delay is one of the primary reasons that the system administrator or user should have a predictive method for text documents allowed within the network. This paper gives the SA/user a method to establish an evaluation method while waiting for a CVE to publish that may be in the NVD backlog or other vulnerability databases.

Another related article by Ruohonen, Hyrynsalmi, and Leppänen (2017) discussed modeling the delivery of security advisories and CVEs. The article shows a lifecycle for vulnerabilities:

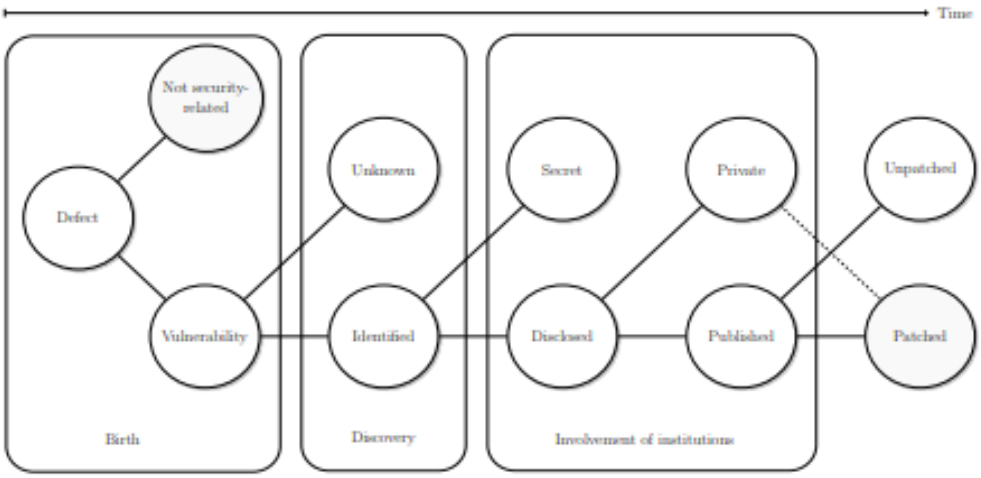


Figure 26. Vulnerability life cycle states (Ruohonen et al., 2017).

The article details the releases of advisories in Figure 27.

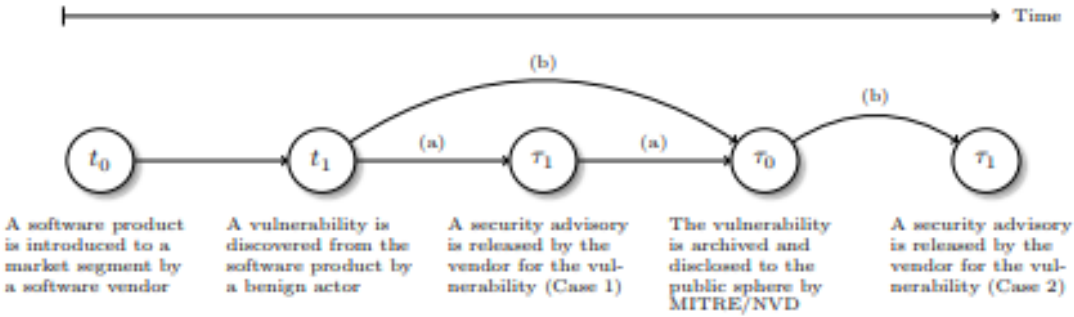


Figure 27. Timelines for advisory releases (Ruohonen et al., 2017).

The article also introduces an equation for approximating the age of a given operating system product at the time of the corresponding security advisory release shown as follows:

$$A_i = \tau_1 - t_0, \qquad A_i \geq 0,$$

As shown in this article age of vulnerability should be taken into effect for calculation. In this paper the temporal CVSS is used to illustrate that aging effect and give the SA/user a more informative score than the base score. Additionally, the method described in this paper allows the user to reach an informed decision.

Akilandeswari (2012) article surveyed the various methods to identify the legitimate/illegitimate traffic on different networks. A new method is proposed in the article builds a reliable identification model for Flash Crowd and Distributed Denial of Service (DDoS) attacks (Akilandeswari, 2012). The authors propose a PNN based traffic pattern classification method that is used for effective classification of attack traffic from legitimate traffic (Akilandeswari, 2012). The authors discuss that Flash Crowd Event

(FCE) means network or host receives lot of traffic (Akilandeswari, 2012). Moreover, this type of attack creates sudden change in traffic levels (Akilandeswari, 2012). The FCE is created by the large surge of legitimate requests and it is focused on some specific sites on Internet, over particular period of time. DDoS attacks also have the similar possessions such as sudden changes in traffic levels; focus on particular web server, and website unavailability due to huge number of requests (Akilandeswari, 2012). The two forms of DDoS attacks are flooding attacks and vulnerability attacks (Akilandeswari, 2012). The authors used the metrics detailed in Table 3 to put in their PNN. In table 3, the TPR and FPR represent the True Positive and False Positive rates, respectively.

Table 3

FCE and DDoS Attack Detection Techniques (Akilandeswari, 2012)

Techniques	Metric	FPR (%)	TPR (%)	Classification accuracy
Maximum Entropy	Traffic speed or Change of traffic	2	80	Medium
Flash Events and DDoS Distinguisher (FDD)	Cluster distribution and their request	3	85	High
Hybrid Probability Metric	Access intents, Distribution of source IP address, Level of traffic changes	1	87	High
Information Distance Measurement	Distribution of Source IP address	2	65	Low
Persistent Increment Feature	DDoS attack attribute	4	79	Low

The process that Akilandeswari et al. (2012) outline is shown in Figure 28.

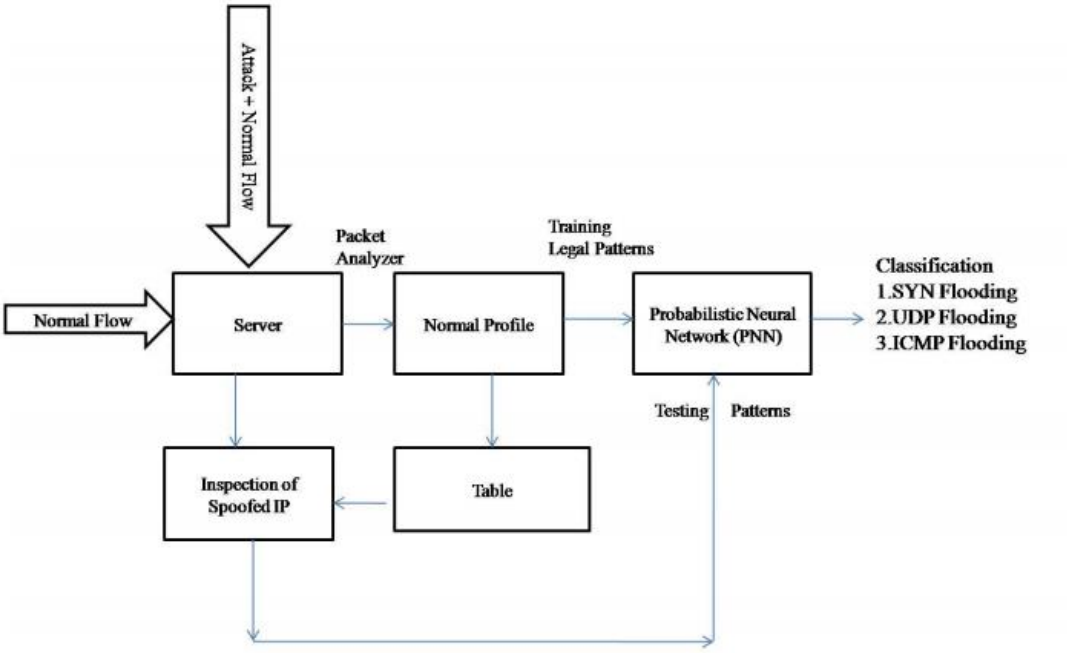


Figure 28. PNN based attack classification (Akilandeswari, 2012).

The proposed Probabilistic Neural Network (PNN) based pattern classification provides the classification and separation of TCP SYN, UDP, ICMP attack patterns (Akilandeswari, 2012). These patterns will help to reduce the real time traffic challenges which lie in the highly distributed attacks with both low and high packet rates (Akilandeswari, 2012). Although this article uses PNN to assist in the classification of cyber threats it did not take textual data into account.

Ancona, Colla, Rovetta, and Zunino (1998) focused on the implementation of PNNs in hardware. The article is an overview of the PNN and a modification of the PNN described by Specht in the original paper (Ancona et al., 1998). The authors elaborated that the standard PNN has a very short training time when implemented in hardware (on a

commercially available digital neural chip the Nestor/Intel Ni1000), but this new modified PNN would allow for less classification and an unlimited number of units (Ancona et al., 1998). The proposed PNN provided an elimination criterion to avoid the storage of unnecessary patterns (Ancona et al., 1998). According to the authors the proposed algorithm makes it possible to realize the PNN in hardware and compensates for the inadequacies from the standard PNN which does not perform well with small training sets (Ancona et al., 1998). The article also details Parzen's estimate and the Bayesian decision criterion (Ancona et al., 1998). The Parzen windows method is a nonparametric identification procedure that creates an estimate of a probability density function by superposition of a number of windows, replicas of a function g() called the kernel (Ancona et al., 1998):

$$f(x) \cong f_n(x) = \frac{1}{n\lambda} \sum_{i=1}^{n} g\left(\frac{x - x(l)}{\lambda}\right)$$

Where *x* is the dummy argument for a point in the sample space, the patterns *x*(*l*) form the training set, and λ is a function of such that

$$\lim_{n \to \infty} \lambda = 0 \quad \text{and} \quad \lim_{n \to \infty} n\lambda = \infty$$

In the proposed algorithm, two modifications to the standard PNN were proposed, the first was the design of a criterion for selecting some patterns to be stored and reflecting others that were considered unnecessary (Ancona et al., 1998). The second modification implement consists of assigning new roles to the parameters of the model (Ancona et al., 1998). We allow the value of the window size λ to vary from unit to unit. The modified

algorithm is structured into two blocks: the first is the creation of the windows; the second is the optimization of the window parameters. The training algorithm consisted of the pseudo code in Figure 29 for training the PNN.

```
ALGORITHM: PNNtraining
begin procedure
mark all patterns as unused
while training set contains unused
    patterns and cost(p) < C_th
     repeat
          set l = randominteger(1, N) and
    set k = class(x(l))
     until x(l) used
     mark x(l) as used
     compute the class probability
    values
     choose class c with max
    probability
     choose class c' with the second
    max probability
     if f^(c)(x(l))/f^(c')(x(l)) < r and n < n_max
    then
          add a new pattern unit
    centred on x(l)
          while number of iterations
    ≤ I_max
               select new p = optimize(p)
          end while
     end if
end while
end procedure
```

Figure 29. Modified PNN training algorithm (Ancona et al., 1998).

According to the authors of the article, Experimental verifications have demonstrated that, if compared with the standard version, the modified algorithm requires a much smaller number of units to obtain a comparable performance level. This algorithm

was primarily for hardware implementation and was not judged as feasible for textual document classification.

In another work related to PNNs, Modaresi and Araghinejad (2014) uses the neural network for classification of water quality. Although not strongly related to this paper, the article compared SVM, PNN, and KNN to determine which classification would produce the lowest error rate and error value using the equations in Figure 30.

1- Error Rate (ER): It is calculated by the following equation:

$$Error\ Rate(\%) = \frac{Number\ of\ misclassified\ data}{Total\ number\ of\ data} \times 100$$

2- Error Value (EV): which is determined by the Eq. (14):

$$Error\ Value = (Observed\ class - Simulated\ class)^2$$

Figure 30. Error rate and error value measures (Modaresi & Araghinejad, 2014).

One of the primary reasons the PNN was chosen for this paper was the result of the Modaresi and Araghinejad (2014) study. In the article, the results show that the SVM algorithm was the best performance of all although the PNN algorithm also has the low number and magnitude of errors (Modaresi & Araghinejad, 2014). The KNN algorithm, had the most total number and value of errors, which showed it was the weakest at classification of data (Awan & Rana, 2016). The article also noted that the training process of the SVM algorithm is more difficult than the PNN and KNN (Modaresi & Araghinejad, 2014).

Confusion Matrix

In Lantz's (2013) book regarding machine learning using R, the author discusses a method to evaluate a machine model. A confusion matrix is a table that categorizes predictions according to whether they match the actual value in the data (Lantz, 2013). One of the table's dimension indicates the possible categories of predicted values while the other dimension indicates the same for actual values (Lantz, 2013). When the predicted value is the same as the actual value, this is a correct classification (Lantz, 2013). Correct predictions (O) fall on the diagonal in the confusion matrix (Lantz, 2013). The off-diagonal matrix cells (O) indicate the cases where the predicted value differs from the actual value, these are considered incorrect predictions (Lantz, 2013). Figure 31 shows a confusion matrix.

T R U T H	O	X	X
	X	O	X
	X	X	O
P R E D I C T E D			

Figure 31. Confusion matrix (Lantz, 2013).

The confusion matrix is used to summarize the performance of a machine learning classification algorithm (Phifer, 2011). The matrix gives the implementer what the classification model is getting right and what it is getting wrong. In other words; The

number of correct and incorrect predictions are summarized with count values and these are broken down by each class. This is the key to the confusion matrix, and it is especially useful when the number of classes in the classification goes beyond 2 (the so called binary classification; Phifer, 2011).

Typically, the correct and incorrect numbers/counts are organized into a table or matrix, where each row of the matrix or table corresponds to a predicted class, while each column correspond to an actual class (Phifer, 2011). The points where these two intersect is on the diagonal of the matrix. This is where predicted class and actual class match up. The off-diagonal values/counts show how many of a given class were incorrectly predicted and into which class they were placed.

Some what can be had from a confusion matrix:

1. True positive (TP)
2. False positive (FP)
3. True negative (TN)
4. False negative (FN)
5. Accuracy
6. Error rate
7. Sensitivity
8. Specificity (Phifer, 2011).

Chapter 3: Methodology and Process

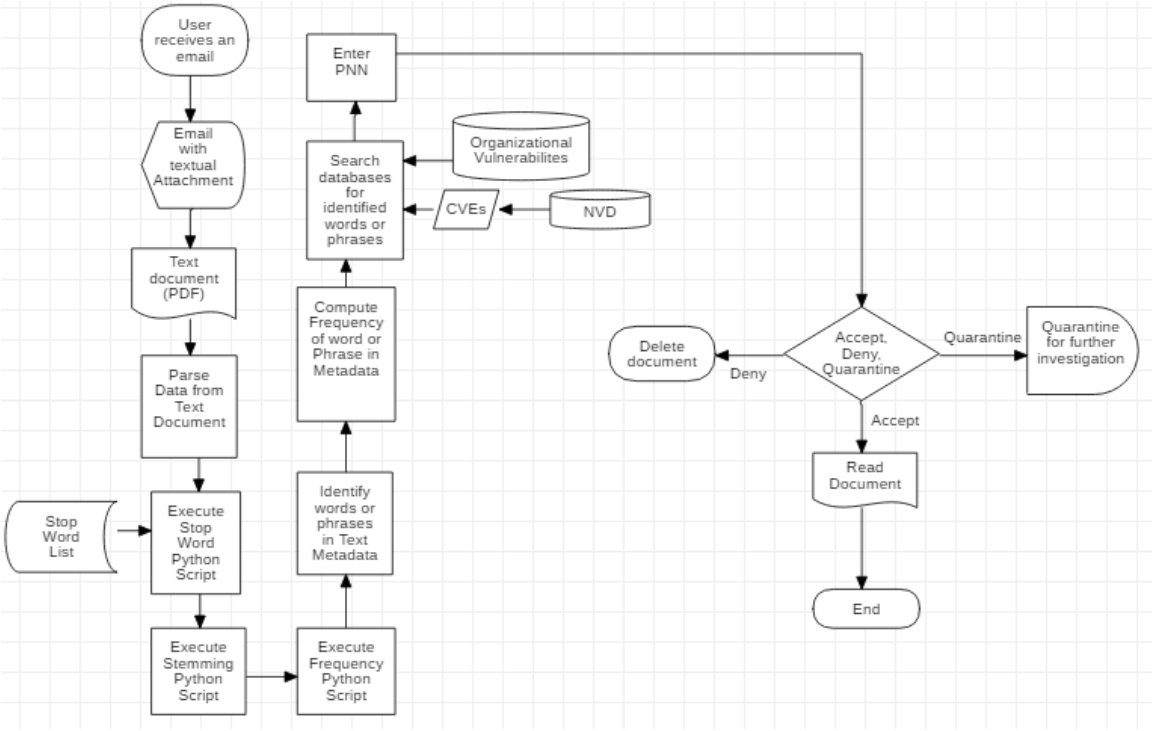


Figure 32. Methodology for automated calculation of risk for a textual document using PNN.

The methodology developed by the author of this paper was based on text mining processes and classification of textual documents using a PNN classifier. The method in this paper is also flexible as it can be open to modification by an organization if a CVE is identified within the organization before it can be placed in the NVD for public information. The organization can adjust the parameters of the method outlined in this paper based on the organization risk tolerance or what is the level of risk the organization is willing to accept.

The data was gathered from the UCI database of text documents that needed to parsed and formatted in a file that could be readable by a python script. This paper uses Portable Document Format (PDF) as the example text document because of the security issues associated with the PDF files. According to Castiglione, De Santis, and Soriente (2010) PDF documents are open to privacy related issues such as it is possible to retrieve any text or object previously deleted or modified and extract user information. Additionally, PDF documents are susceptible to push button malware attacks such as LuckySploit, CrimePack, and Fragus for relatively cheap amounts ranging from $100 to $1000 (Phifer, 2011). PDF documents are also known as the industry standard for portable file exchange formats implemented by many free and commercially available programs (Phifer, 2011). This universal exchange format opens a large attack surface for PDF users with the use of JavaScript objects within the Adobe Reader software package that is necessary to read the PDF documents (Phifer, 2011). Although Adobe has an auto-update feature to its software packages, Phifer (2011) says that "few businesses can afford to simply block PDF attachments and downloads—legitimate PDFs are just far too prevalent and ingrained in our business practices."

The first step in the process is to take a given PDF file and read the metadata in by using an author modified python package PyPDF2 for extracting information from PDF documents (Shinyama, 2014). This tool focuses on analyzing text data and has a PDF parser that is an integral part of this process (Shinyama, 2014).

The steps for extracting text from a PDF document using a python script:

1. Import the PyPDF2 module
2. Import the collections module (if multiple files)
3. define the PDF file that is to be turned to text
4. Identify the number of pages within the PDF from step 3 and read those pages
5. Perform the extraction
6. Display or print text content.

Once the PDF document is changed to a text document, an author modified python module is executed to eliminate the stop words. The most frequently occurring words according to Fox (1989) are *the, of, and*, and *to*. A stop word list was used from the Natural Language Toolkit (NLTK) stop word list that is part of the python toolkit stored in 16 different languages (Upadhyay, n.d.).

Following the elimination of the stop words, the next phase of the process is to conduct the stemming operation to identify the stem of the words. Stemming is a process in which the variant word forms are shortened to their base forms (J. Singh & Gupta, 2016). There are three types of stemmers that currently exist, rule based, statistical, and hybrid (J. Singh & Gupta, 2016). The stemmer used in this paper was a rule based stemmer that was primarily affix removal which removes the suffix and/or the prefix from the variant word forms (J. Singh & Gupta, 2016).

Once the data has been thinned to a usable level from the previously described processes a frequency formula was developed by the author of this paper which is applied to assign how often a word appears in the document.

$$F = \frac{number\ of\ words\ after\ thinning\ in\ text\ document}{number\ of\ times\ CVE\ associated\ word\ occurs}$$

This word which is associated with a CVE could be in the metadata or the text associated with the body of the document.

The frequency of the words can now be turned in to vectors and ready for class comparison using another python script. This script will read in the word list of the most frequent words turning them into vectors and assign class numbers. The vectors are then sorted into the K sets, where each set contains one class of feature vectors (Frequency, Time CVE is in the NVD [age], Time of discovery of vulnerability). Then we define the PNN to feed the vectors into it and classify them.

The probability of a vulnerability being exploited hits 90% between 40–60 days after discovery (Seals, 2015). This means that the remediation gap, or time that vulnerability is most likely to be exploited before it is closed, is nearly 60 days (Seals, 2015).

The architecture developed for this paper uses three classes (K=3). The input layer contains the 3 features (nodes) used for the PNN. Therefore, the features for the determination of acceptance, quarantine, denial would align to

1. The *frequency (F)* of the word in the document
2. The time (*T*) CVE is in the NVD—age of CVE
3. The time (*t*) of discovery of the vulnerability.

Step 1: Obtain frequency vector (*F*)

a. All of this is standard text mining steps using python which were defined previously.
b. An author modified Python function to read the metadata and contents from the PDF files.
c. An author modified Python Function to identify word or phrases that occur in the metadata.
d. Compute the frequency of the word or phrase

Step 2: Obtain Age vector (*T*)

a. Search the CVE database for the identified the word or phrase.
b. A author developed Python function to determine if the word or phrase is identified/described as threat (e.g. CVE-1999-1576 Candidate Buffer overflow in Adobe Acrobat ActiveX control (pdf.ocx, PDF.PdfCtrl.1) 1.3.188 for Acrobat Reader 4.0 allows remote attackers to execute arbitrary code via the pdf.setview method. BUGTRAQ:19990924 Several ActiveX Buffer Overruns | URL:http://www.securityfocus.com/archive/1/28719 | CERT-VN:VU#25919 | URL:http://www.kb.cert.org/vuls/id/25919 | BID:666 | URL:http://www.securityfocus.com/bid/666 | XF:adobe-acrobat-pdf-bo(3318) | URL:https://exchange.xforce.ibmcloud.com/vulnerabilities/3318 Assigned (20050421) None (candidate not yet proposed)

c. Determine the time stamp for the most recent CVE entry for this word or phrase

d. Determine the time stamp for the oldest CVE entry for this word or phase

e. A CVSS score may or can attached to the CVE entry

f. The Operator can enter the known threat word or phase that does not have a CVE record in the CVS database (ex. MYRRICO06)

Step 3: Time stamp of document or organizational discovery of vulnerability (*t*)

a. The operator can enter time of discovery or time stamp of document can be read in from metadata of the textual document.

In order to simulate the features in the system, a reliable dataset containing at least 3 features and 3 classes or categories of classifications was necessary. As stated earlier, the organization can set *F*, *T*, and *t* to the correct organizational parameters. As discussed in Chapter 2, risk changes over time, therefore the basis for the classification of the PNN, an organization may tighten or loosen its risk parameters based on company growth or additional cyber security systems/procedures in place.

The IRIS and Balance Scale datasets were selected. Each dataset contains 4 features and 3 classifications. This meant that a feature selection method had to be used to select the best 3 of the four features. By using an author developed KNN and a Sequential Forward Selection, the best 3 features were selected for k=3 on both datasets.

Sequential Forward Selection (k=3):

(1, 2, 3) CV Score: 0.972756410256

(accuracy to which the KNN has selected the best 3 features within the IRIS dataset)

In this paper the IRIS database is used to demonstrate the use of the textual mining and PNN approach. This dataset is perhaps the best known database to be found in the pattern recognition literature (Dua, 2017). The data set contains 3 classes of 50 instances each, where each class refers to a type of Iris plant (Dua, 2017). One class is linearly separable from the other 2; the latter are NOT linearly separable from each other (Dua, 2017).

Table 4

IRIS Dataset (Dua, 2017)

Data Set Characteristics:	Multivariate
Number of Instances:	150
Attribute Characteristics:	Real
Associated Tasks:	Classification
Number of Attributes:	4

Another dataset used in this paper is the Balance Scale dataset which was generated to model psychological experimental results (Dua, 2017). Each example is classified as having the balance scale tip to the right, tip to the left, or be balanced (Dua, 2017). The attributes are the left weight, the left distance, the right weight, and the right distance (Dua, 2017). The correct way to find the class is the greater of (left-distance * left-weight) and (right-distance * right-weight; Dua, 2017). If they are equal, it is balanced.

Table 5

Balance Scale Dataset (Dua, 2017)

Data Set Characteristics:	Multivariate
Number of Instances:	625
Attribute Characteristics:	Categorical
Associated Tasks:	Classification
Number of Attributes:	4

Chapter 4: The Results

The objective of this paper is to show that textual documents can be evaluated by a user whether the document is safe to accept it, deny it, or quarantine it for further investigation. The use of the NVD, CVSS, and available CVEs are part of the evaluation used by the user or organization. The training set used by this paper is as follows:

Training data (75 data points) for IRIS dataset [*F, T, t*]:

[3.5 1.4 0.2]

[3. 1.4 0.2]

[3.2 1.3 0.2]

[3.1 1.5 0.2]

[3.6 1.4 0.2]

[3.9 1.7 0.4]

[3.4 1.4 0.3]

[3.4 1.5 0.2]

[2.9 1.4 0.2]

[3.1 1.5 0.1]

[3.7 1.5 0.2]

[3.4 1.6 0.2]

[3. 1.4 0.1]

[3. 1.1 0.1]

[4. 1.2 0.2]

[4.4 1.5 0.4]

[3.9 1.3 0.4]

[3.5 1.4 0.3]

[3.8 1.7 0.3]

[3.8 1.5 0.3]

[3.4 1.7 0.2]

[3.7 1.5 0.4]

[3.6 1. 0.2]

[3.3 1.7 0.5]

[3.4 1.9 0.2]

[3.2 4.7 1.4]

[3.2 4.5 1.5]

[3.1 4.9 1.5]

[2.3 4. 1.3]

[2.8 4.6 1.5]

[2.8 4.5 1.3]

[3.3 4.7 1.6]

[2.4 3.3 1.]

[2.9 4.6 1.3]

[2.7 3.9 1.4]

[2. 3.5 1.]

[3. 4.2 1.5]

[2.2 4. 1.]

[2.9 4.7 1.4]

[2.9 3.6 1.3]

[3.1 4.4 1.4]

[3. 4.5 1.5]

[2.7 4.1 1.]

[2.2 4.5 1.5]

[2.5 3.9 1.1]

[3.2 4.8 1.8]

[2.8 4. 1.3]

[2.5 4.9 1.5]

[2.8 4.7 1.2]

[2.9 4.3 1.3]

[3.3 6. 2.5]

[2.7 5.1 1.9]

[3. 5.9 2.1]

[2.9 5.6 1.8]

[3. 5.8 2.2]

[3. 6.6 2.1]

[2.5 4.5 1.7]

[2.9 6.3 1.8]

[2.5 5.8 1.8]

[3.6 6.1 2.5]

[3.2 5.1 2.]

[2.7 5.3 1.9]

[3. 5.5 2.1]

[2.5 5. 2.]

[2.8 5.1 2.4]

[3.2 5.3 2.3]

[3. 5.5 1.8]

[3.8 6.7 2.2]

[2.6 6.9 2.3]

[2.2 5. 1.5]

[3.2 5.7 2.3]

[2.8 4.9 2.]

[2.8 6.7 2.]

[2.7 4.9 1.8]

[3.3 5.7 2.1]

Training Classes for the IRIS dataset are

[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1

1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

2]

The 0s, 1s, and 2s correspond to the accept, quarantine, and deny, respectively.

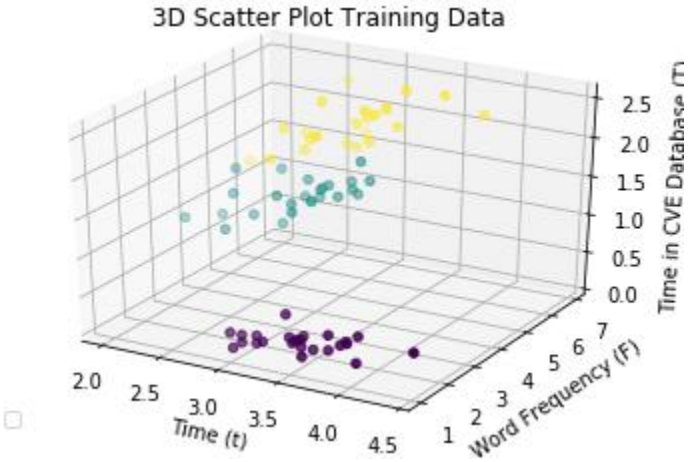


Figure 33. Plot of IRIS training data. Green: deny (high frequency short times), yellow: quarantine (high frequency longer times), violet: accept (low frequency).

After training the PNN with IRIS data, the test data of 75 points is as follows:

Test data for IRIS dataset [*F, T, t*]:

[3. 1.6 0.2]

[3.4 1.6 0.4]

[3.5 1.5 0.2]

[3.4 1.4 0.2]

[3.2 1.6 0.2]

[3.1 1.6 0.2]

[3.4 1.5 0.4]

[4.1 1.5 0.1]

[4.2 1.4 0.2]

[3.1 1.5 0.1]

[3.2 1.2 0.2]

[3.5 1.3 0.2]

[3.1 1.5 0.1]

[3. 1.3 0.2]

[3.4 1.5 0.2]

[3.5 1.3 0.3]

[2.3 1.3 0.3]

[3.2 1.3 0.2]

[3.5 1.6 0.6]

[3.8 1.9 0.4]

[3. 1.4 0.3]

[3.8 1.6 0.2]

[3.2 1.4 0.2]

[3.7 1.5 0.2]

[3.3 1.4 0.2]

[3. 4.4 1.4]

[2.8 4.8 1.4]

[3. 5. 1.7]

[2.9 4.5 1.5]

[2.6 3.5 1.]

[2.4 3.8 1.1]

[2.4 3.7 1.]

[2.7 3.9 1.2]

[2.7 5.1 1.6]

[3. 4.5 1.5]

[3.4 4.5 1.6]

[3.1 4.7 1.5]

[2.3 4.4 1.3]

[3. 4.1 1.3]

[2.5 4. 1.3]

[2.6 4.4 1.2]

[3. 4.6 1.4]

[2.6 4. 1.2]

[2.3 3.3 1.]

[2.7 4.2 1.3]

[3. 4.2 1.2]

[2.9 4.2 1.3]

[2.9 4.3 1.3]

[2.5 3. 1.1]

[2.8 4.1 1.3]

[3.2 6. 1.8]

[2.8 4.8 1.8]

[3. 4.9 1.8]

[2.8 5.6 2.1]

[3. 5.8 1.6]

[2.8 6.1 1.9]

[3.8 6.4 2.]

[2.8 5.6 2.2]

[2.8 5.1 1.5]

[2.6 5.6 1.4]

[3. 6.1 2.3]

[3.4 5.6 2.4]

[3.1 5.5 1.8]

[3. 4.8 1.8]

[3.1 5.4 2.1]

[3.1 5.6 2.4]

[3.1 5.1 2.3]

[2.7 5.1 1.9]

[3.2 5.9 2.3]

[3.3 5.7 2.5]

[3. 5.2 2.3]

[2.5 5. 1.9]

[3. 5.2 2.]

[3.4 5.4 2.3]

[3. 5.1 1.8]

Test Classes for IRIS dataset are

[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1

1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

 2]

Predicted Classes are

[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 2 1 1 1

1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 1 2 2 2 2 2 1 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2

2]

The 0s, 1s, and 2s correspond to the accept, quarantine, and deny, respectively.

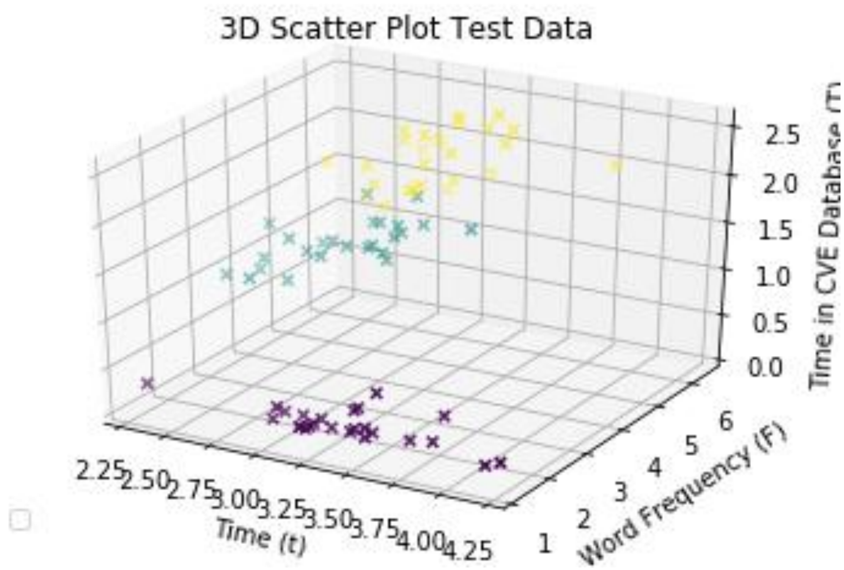


Figure 34. Plot of IRIS test data. Green: deny (high frequency short times), yellow: quarantine (high frequency longer times), violet: accept (low frequency).

The combined dataset is plotted in Figure 35.

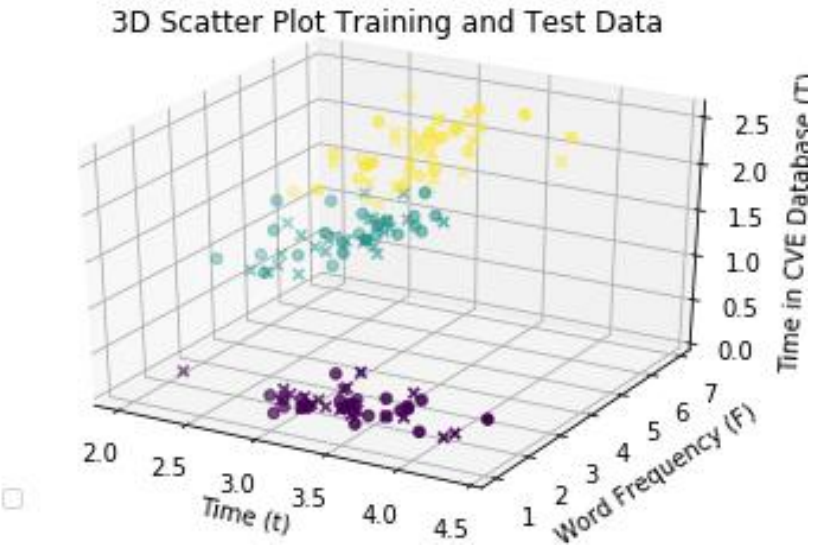


Figure 35. Plot of IRIS combined dataset. Green: deny (high frequency short times), yellow: quarantine (high frequency longer times), violet: accept (low frequency).

Accuracy for IRIS Dataset

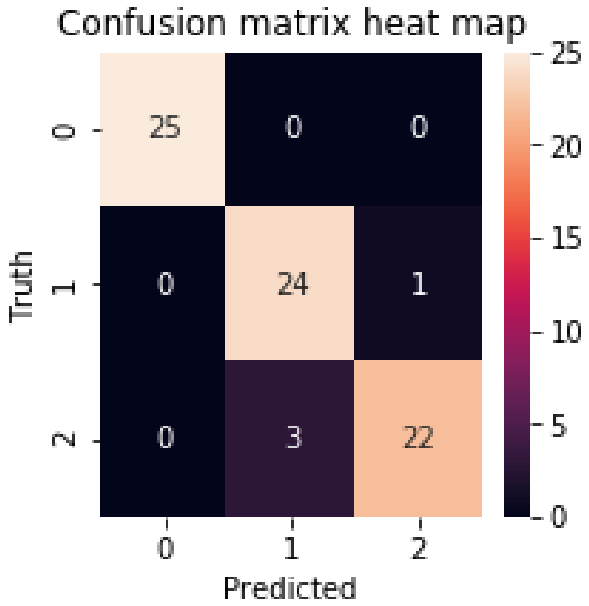


Figure 36. Confusion matrix for IRIS database.

The results from the confusion matrix for the IRIS show:

For class 0 (zero), ACCEPT, all 25 were correctly predicted to be in that class.

For class 1 (one), QUARANTINE, 24 were correctly predicted to be in this class, while 1 was falsely or incorrectly predicted to be acceptable (in class 0).

For class 2 (two), DENY, 22 were correctly predicted to be in this class, while 3 were falsely predicted to be eligible for quarantine. The heat map in Figure 36 indicates that the lighter colors relate to higher numbers while the darker colors correspond to lower numbers.

The Balance Scale dataset training data is located in Appendix A. The Balance Scale Training data [*F, T, t*]:

[1 1 1]

[1 1 2]

[1 1 3]

[1 1 4]

[1 1 5]

[1 1 1]

[1 1 2]

[1 1 3]

[1 1 4]

[1 1 5]

[1 1 1]

[1 1 2]

[1 1 3]

[1 1 4]

Training Classes are

[8 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 7 8 9 9 9 8 9 9 9 9 9 9

9 9 9 9 9 9 9 9 9 9 9 9 9 7 7 8 9 9 7 9 9 9 9 8 9 9 9 9 9 9 9 9 9 9 9 9 9

9 7 7 7 8 9 7 8 9 9 9 7 9 9 9 9 8 9 9 9 9 9 9 9 9 9 7 7 7 7 8 7 7 9 9 9 7

9 9 9 9 7 9 9 9 9 8 9 9 9 9 7 8 9 9 9 9 8 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9

9 9 7 7 7 8 9 7 8 9 9 9 7 9 9 9 9 8 9 9 9 9 9 9 9 9 9 7 7 7 7 7 7 7 8 9 9

7 8 9 9 9 7 9 9 7 7 7 7 7 7 7 7 7 8 7 7 7 8 7 7 7 7 7 7 7 7 7 7 8 7 7 7 7

7 7 8 7 7 8 7 8 7 7 7 7 7 7 7 7 8 7 8 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7

7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7

7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7]

The 7s, 8s, and 9s correspond to the accept, quarantine, and deny, respectively.

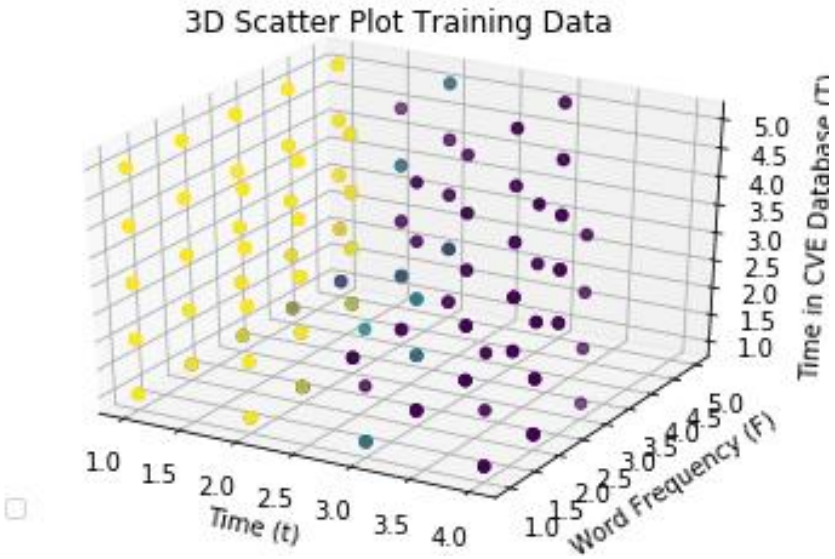


Figure 37. Plot for balance scale training data.

A portion of the Balance Scale Test data is located below, but the remainder of the test data Balance Scale is located in Appendix B [*F*, *T*, *t*]:

[2 3 4]

[2 3 5]

[2 3 2]

[2 3 3]

[2 3 4]

[2 3 5]

[2 4 5]

[2 4 3]

[2 4 4]

[2 4 5]

[2 4 3]

[2 4 4]

[2 4 5]

[2 4 2]

[2 4 3]

Test Classes are

[9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9

9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 8 9 9 9 9 9 9 9 9 9 8 9 8 9 9

9 9 9 8 9 9 8 9 9 8 9 8 9 9 9 9 9 9 9 8 9 9 9 9 9 9 9 9 9 8 9 9 9 9 8 9 9

9 9 9 9 9 7 7 7 7 7 7 7 7 7 8 9 7 7 8 9 9 7 7 9 9 9 7 7 7 7 7 7 7 7 7 7 7

7 7 7 7 7 7 7 8 9 7 7 7 9 9 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 8 7 7 7

8 9 7 7 7 7 8 7 7 9 9 9 7 9 9 9 9 7 9 9 9 9 8 9 9 9 9 7 7 7 7 7 7 7 7 7 8

7 7 7 9 9 7 7 9 9 9 7 8 9 9 9 7 7 7 7 7 7 7 7 7 7 7 7 7 7 8 7 7 7 9 9 7 7

8 9 9 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 8 7 7 7 8 9 7 7 7 7 7 7 7 7 7

7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 8]

Predicted Classes are

[9 9 8 8 9 9 7 7 8 7 7 8 7 8 7 8 7 7 8 7 7 8 7 7 8 8 9 7 8 8 9 7 8 8 9 8 7

8 8 9 8 7 8 8 9 7 7 8 7 7 8 8 7 7 8 8 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7

7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7

7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7

7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7

7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7

7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7

7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7

7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7]

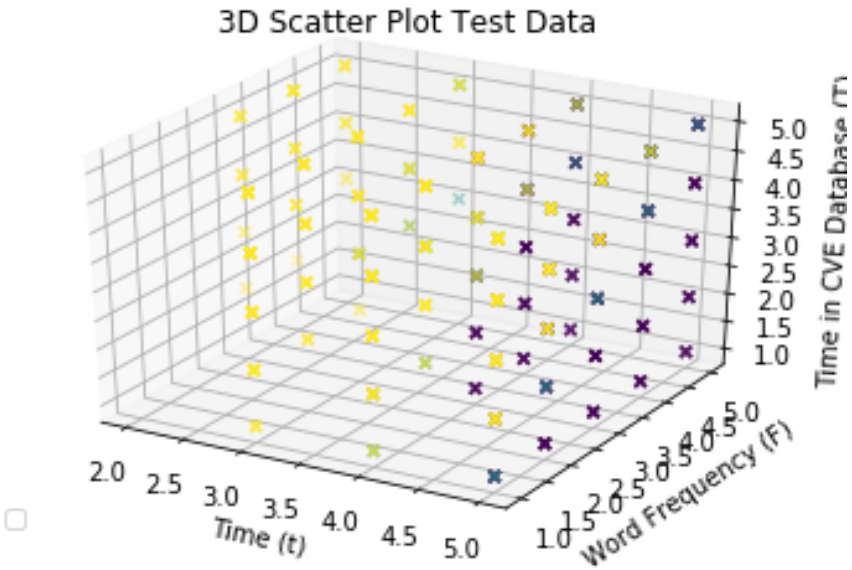


Figure 38. Plot for balance scale test data.

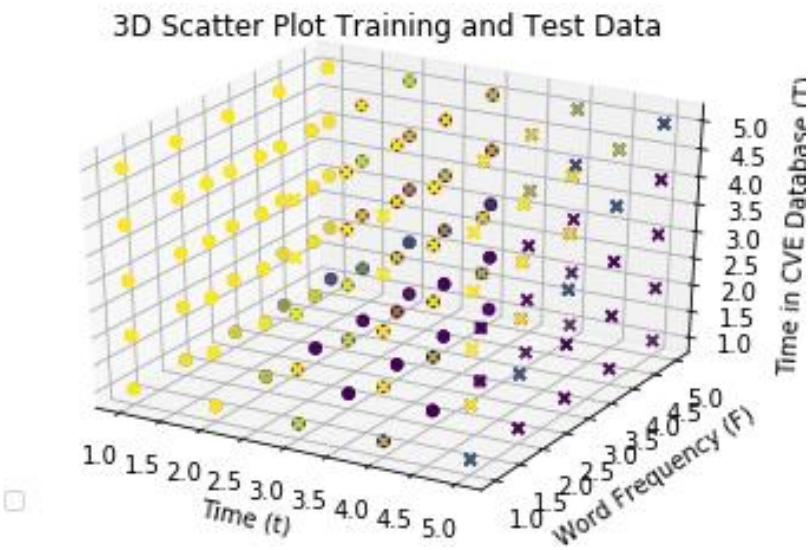


Figure 39. Plot of training and test data for balance scale dataset.

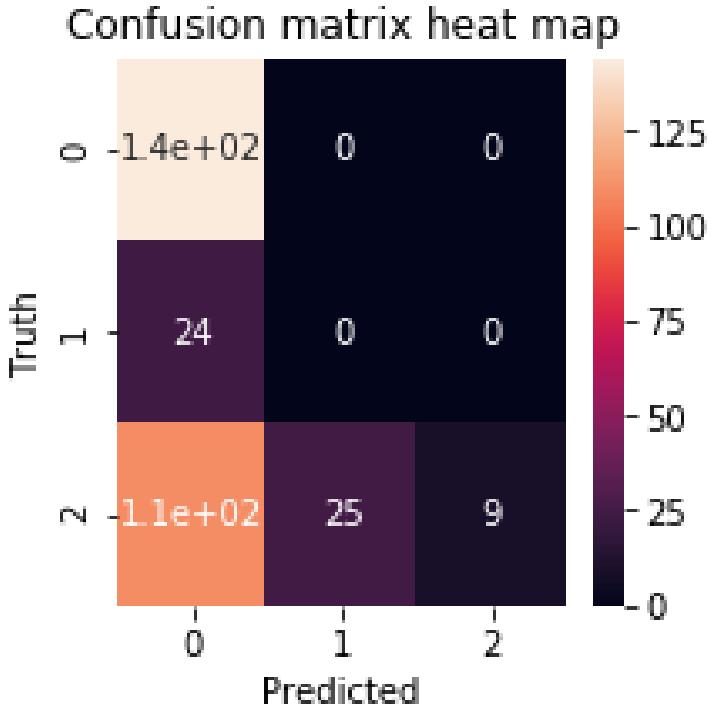


Figure 40. Confusion matrix for balance scale dataset.

The results from the confusion matrix for the Balance Scale show:

For class 0 (zero), ACCEPT, all 140 were correctly predicted to be in that class.

For class 1 (one), QUARANTINE, 0 were correctly predicted to be in this class, while 24 was falsely or incorrectly predicted to be acceptable (in class 0 [zero]).

For class 2 (two), DENY, 9 were correctly predicted to be in this class, while 25 were falsely predicted to be eligible for quarantine and 110 were falsely predicted to be in accept. The heat map in Figure 40 indicates that the lighter colors relate to higher numbers while the darker colors correspond to lower numbers.

$$\text{Accuracy} = \frac{(\#\text{ correct predictions})}{(\#\text{ total predictions})}$$

(140+0+9)/(9+25+110+24+140) = 149/308 =0.4838 (Accuracy of the Balance Scale dataset).

Chapter 5: Conclusion and Discussion

The accuracy of the automated calculation of a risk decision for a textual document using PNN was shown with the two datasets IRIS and Balance Scale. The IRIS dataset performed extremely well within the algorithm yielding an accuracy of almost 95%. On the other hand, the Balance Scale dataset showed a fairly low accuracy rate of below 50%. Contributors to the accuracy difference within the dataset may be attributed to the facts detailed earlier that IRIS is a more well known and used dataset for pattern recognition and may have better classification. The Balance Scale dataset is not as widely known and as well classified. Additionally, looking at the vectors, the IRIS dataset was more precise with real numbers versus the Balance Scale using simply integrators for classification.

In this paper, the focus was to enable the organization or the user with an informed risk decision about a text document or possible set of textual documents for acceptance, denial, or quarantine. The paper showed that using the elements of textual data mining and a vulnerability database, an accurate classification could be determined using a PNN coded in python. The use of the automated calculation of risk for a textual document using PNN methodology provided an organization/system administrator/user the ability to classify vulnerabilities allowing for dynamic risk tolerance levels. Essentially, the feature selection provides for more control and flexibility of whether to accept a textual document. This method also provided for speed of decision making for acceptance of the document.

Future

Research continues in machine learning for textual data mining and cyber risk decision evaluations. Future directions based on the algorithm and methods outlined in this paper are vast with possibilities as risk and vulnerabilities grow within the cyber environment. Extension of the data sources for this automated algorithm using real organizational data. This extension to the organizational data would likely assist in tuning the PNN to what the organization is willing to accept based on the accuracy depicted in the confusion matrices.

Another path of future work would entail adding more features to the PNN for an even more informed decision on the whether to accept, deny, or quarantine the text document.

Additional features could be based on what the organization is tracking for risk. This could be a internal score that is fed to the PNN or other cyber scoring systems that an organization or user may view as more accurate with features that are more relevant. Moreover, future work could encompass a organization forming a more accurate database of vulnerabilities and have that as an additional feed with national vulnerabilities databases other than simply the NVD.

PDFs are not the only types of documents available that an organization will likely receive, other textual documents can be used along with e-mails and spam messages. This would enable to classify e-mail and spam messages quickly based on a changing cyber landscape. Lastly, if integrated within a cloud environment, this method

could be used as a vulnerability as a service (VaaS) for expedient classification of acceptable messages and documents for user and organizational use.

References

Abdelmoula, A. K. (2015). Bank credit risk analysis with k-nearestneighbor. *Accounting and Management Information Systems, 14*(1), 79–106. Retrieved from http://www.cig.ase.ro/revista_cig/Fisiere/14_1_4.pdf

Abikoye, O., Omokanye, S., & Aro, T. (2018). *Text classification using data mining techniques: A review.* Retrieved from http://ezproxy.umuc.edu/login?url=http://search.ebscohost.com.ezproxy.umuc.edu/login.aspx?direct=true&db=edsgic&AN=edsgcl.541103802&site=eds-live&scope=site

Aitnouri, E., & Ouali, M. (2010). Performance evaluation of clustering techniques for image segmentation. Computer Science Journal of Moldova, vol.18, no.3, 271-302. Retrieved from https://pdfs.semanticscholar.org/d0f5/3a3bc72af3e25c5a1a2270015de743801efc.pdf

Akilandeswari, V. (2012, December). *Probabilistic neural network based attack traffic classification.* Paper presented at 4th International Conference on Advanced Computing. doi:10.1109/ICoAC.2012.6416848

Alguliev, R. M., Aliguliyev, R., & Nazirova, S. (2011). Classification of textual e-mail spam using data mining techniques. *Applied Computational Intelligence and Soft Computing*, (416308), 1–8. doi:10.1155/2011/416308

Alhazm, O. H., Sung-Whan, W., & Malaiya, Y. K. (2006). *Security vulnerability catagories in major software systems.* Retrieved from http://www.cs.colostate.edu/~malaiya/pub/CNIS-547-097.pdf

Ancona, F., Colla, A., Rovetta, S., & Zunino, R. (1998). Implementing probabilistic neural networks. *Neural Computing & Applications, 7,* 37–51. doi:10.1007 /BF01413860

Awan M. S. K., & Rana, P. B. O. (2016). Identifying cyber risk hotspots: A framework for measuring temporal variance in computer network risk. *Computers & Security, 57,* 31–46. doi:10.1016/j.cose.2015.11.003

Bramer, M. (2013). *Principles of data mining.* London, England: Springer.

Castiglione, A., De Santis, A., & Soriente, C. (2010). Security and privacy issues in the portable document format. *Journal of Systems and Software, 4,* 1813–1822. Retrieved from https://www.journals.elsevier.com/journal-of-systems-and -software

Cheung, V., & Cannons, K. (2002). *An introduction to probabilistic neural network.* Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.182 .4592&rep=rep1&type=pdf

Cisar, P., Rajnai, Z., Cisar, S., & Pinter, R. (2016). Scoring system as a method of improving IT vulnerability status. *Annals of Faculty Engineering Hunedora-International Journal of Engineering,* 207–218. Retrieved from http://annals.fih .upt.ro/pdf-full/2016/ANNALS-2016-3-35.pdf

Craig, A., Shackelford, S., & Hiller, J. S. (2015). Proactive cybersecurity: A comparative industry and regulatory analysis. *American Business Law Journal, 52*(4), 721–787. Retrieved from https://alsb.org/alsb-publications/

de Gusmão, A. P. H. (2018). Cybersecurity risk analysis model using fault tree analysis and fuzzy decision theory. *International Journal of Information Management, 43,* 248–260. doi:10.1016/j.ijinfomgt.2018.08.008

DHS/NCCIC. (2018). *National Vulnerabilities Database.* Retrieved from https://nvd.nist.gov/#

Dua, D. A. (2017). *UCI Machine Learning Repository.* Irvine: University of California, School of Information and Computer Science.

FIRST.Org. (2018). *Common Vulnerability Scoring System v3.0.* Retrieved from https://www.first.org/cvss/cvss-v30-specification-v1.8.pdf

Fox, C. (1989). A stop list for general text. *ACM SIGIR Forum, 24*(1), 19–21. Retrieved from https://dl.acm.org/

Gold, J. (2016). *Open-source vulnerabilities database shuts down.* Retrieved from https://www.networkworld.com/article/3053613/open-source-tools/open-source-vulnerabilities-database-shuts-down.html

Gupta, V., & Lehal, G. (2009). A survey of text mining techniques and applications. *Journal of Emerging Technologies in Web Intelligence, 1*(1), 60–76. Retrieved from https://www.researchgate.net/publication/42802926_A_Survey_of_Text_Mining_Techniques_and_Applications

Hewahi, N. M. (2018). Hidden Markov model representation using probabilistic neural network. *BRAIN: Broad Research in Artificial Intelligence & Neuroscience, 9*(3), 50–62. Retrieved from https://www.researchgate.net/journal/1798-0461_Journal_of_Emerging_Technologies_in_Web_Intelligence

Holm, H., & Afridi, K. K. (2015). An expert-based investigation of the Common. *Computers & Security, 53,* 18–30. Retrieved from https://www.journals.elsevier .com/computers-and-security/

Holm, H., Ekstedt, M., & Andersson, D. (2012). Empirical analysis of system-level vulnerability metrics through actual attacks. *IEEE Transactions on Dependable and Secure Computing, 9,* 825–837. Retrieved from https://ieeexplore.ieee.org/

Khambhammettu, H. E. (2013). A framework for risk assessment in access control systems. *Computers and Security, 39,* 86–103. Retrieved from https://www .sciencedirect.com/science/

Khanbabaei, M., & Alborzi, M. (2013). The use of genetic algorithm, clustering and feature selection techniques in construction of decision tree models for credit scoring. *International Journal of Managing Information Technology, 5*(4), 13–32. Retrieved from http://www.airccse.org/journal/ijmit/papers/5413ijmit02.pdf

Khazaei, A., Ghasemzadeh, M., & Derhami, V. (2016). An automatic method for CVSS score prediction using vulnerabilities description. *Journal of Intelligent and Fuzzy Systems,* (30), 89–96. Retrieved from https://www.iospress.nl/journal/journal-of -intelligent-fuzzy-systems/

Lantz, B. (2013). *Machine learning with R.* Birmingham, England: Packt.

Li, X., & Ye, N. (2006). A supervised clustering and classification algorithm for mining data with mixed variables. *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans, 36,* 396–406. Retrieved from https://

www.researchgate.net/publication/3412511_A_supervised_clustering_and_classification_algorithm_for_mining_data_with_mixed_variables

Lund, M. S. (2011). *Model-driven risk analysis.* Heidelberg, Germany: Springer.

Mar, S., Johannessen, R., Coates, S., Wegrzynowicz, K., & Andreesen, T. (2012, March). *Global Technology Audit Guide (GTAG®) 1 information technology risk and controls.* Retrieved from https://na.theiia.org/standards-guidance/recommended-guidance/practice-guides/Pages/GTAG1.aspx

Martinasek, Z., Zeman, V., Malina, L., & Martinasek, J. (2016). k-Nearest neighbors algorithm in profiling power analysis attacks. *Radioengineering, 25,* 365–382. Retrieved from https://www.researchgate.net/publication/304005799_k-Nearest_Neighbors_Algorithm_in_Profiling_Power_Analysis_Attacks

McEwan, E. K. (2018). *Root words, roots and affixes.* Retrieved from http://www.readingrockets.org/article/root-words-roots-and-affixes

Mell, P., Scarfone, K., & Romanosky, S. (2007). *A complete guide to the common vulnerability scoring system.* Gaithersburg, MD: National Institute of Standards.

Modaresi, F., & Araghinejad, S. (2014). A comparative assessment of support vector machines, probabilistic neural networks, and k-Nearest neighbor algorithms for water quality classification. *Water Resources Management, 28,* 4095–4111. Retrieved from https://link.springer.com/article/10.1007%2Fs11269-014-0730-z

National Institute of Standards and Technology. (2012). *Guide for conducting risk assessments* (NIST 800-30). Gaithersburg, MD: Author.

Nikolić, B., & Ružić-Dimitrijević, L. (2009). Risk assessment of information technology systems. *Issues in Informing Science and Information Technology, 6,* 595–615. Retrieved from http://iisit.org/Vol6/IISITv6p595-615Nikolic673.pdf

Nimbhorkar, N. B. (2014). Probabilistic neural network in solving various pattern classification problems . *IJCSNS International Journal of Computer Science and Network Security, 14,* 133–137.

Oparaugo, C. (2015). ISO/IEC 27001 process mapping to COBIT 4.1 to derive a balanced scorecard for IT governance. *COBIT Focus,* pp. 1–14.

Ouali, M. (2010). Performance evaluation of clustering techniques for image segmentation. *Computer Science Journal of Moldova, 18,* 271–302. Retrieved from https://www.researchgate.net/publication/220491929_Performance_evaluation_of_clustering_techniques_for_image_segmentation

Padma, A., & Giridharan, N. (2016). Performance comparison of texture feature analysis methods using PNN classifier for segmentation and classification of brain CT images. *International Journal of Imaging Systems & Technology, 26*(2), 97–105. Retrieved from http://www.guide2research.com/journal/international-journal-of-imaging-systems-and-technology

Pendleton, M., Garcia-Lebron, R., Cho, J., & Xu, S. (2016). *A survey on systems security metrics.* doi:10.1145/3005714

Phifer, L. (2011, May 4). *Top 5 PDF risks and how to avoid them.* Retrieved from https://www.esecurityplanet.com/security-how-to/Top-5-PDF-Risks-and-How-to-Avoid-Them-3932511.htm

Rees, L. P., Deane, J. K., Rakes, T. R., & Baker, W. H. (2011). Decision support for cybersecurity risk planning. *Decision Support Systems, 51,* 493–505. Retrieved from https://www.sciencedirect.com/science/

Ruohonen, J. (in press). A look at the time delays in CVSS vulnerability scoring. *Applied Computing and Informatics*. doi:10.1016/j.aci.2017.12.002

Ruohonen, J., Hyrynsalmi, S., & Leppänen, V. (2017). Modeling the delivery of security advisories and CVEs. *Computer Science and Information Systems, 14,* 532–555. doi:10.2298/CSIS161010010R

Seals, T. (2015, October 1). *Companies take an average of 100–120 days to patch vulnerabilities.* Retrieved from https://www.infosecurity-magazine.com/news /companies-average-120-days-patch/

Shameli-Sendi, A., Aghababaei-Barzegar, R., & Cheriet, M. (2016). Taxonomy of information security risk assessment (ISRA). *Computers & Security, 57,* 14–30. Retrieved from https://www.sciencedirect.com/science/

Shinyama, Y. (2014). *PDFminer. MIT license.* Retrieved from https://pypi.org/project /pdfminer/

Singh, A. K. (2011). *Probabilistic neural network.* Retrieved from http:// avinashfuturevision.weebly.com/uploads/1/3/3/0/13301560/probabilistic_neural _network.pdf

Singh, J., & Gupta, V. (2016). Text stemming: Approaches, applications, and challenges. *ACM Computing Surveys, 49*(3), 1–46. doi:10.1145/2975608

Specht, D. (1988). *Probabilistic neural networks for classification, mapping, or associative memory.* Retrieved from http://www.inf.ufrgs.br/~engel/data/media/file/cmp121/PNN.pdf

Specht, D. (1990). Probabilistic neural networks. *Neural Networks, 3,* 109–118.

Sun, J., Li, H., Chang, P., & Huang, Q. (2015). Dynamic credit scoring using B&B with incremental-SVM-ensemble. *Kybernetes, 44,* 518–535. Retrieved from https://www.emeraldinsight.com/doi/abs/10.1108/K-02-2014-0036?journalCode=k

Talib, M. A. (2012). Guide to ISO 27001: UAE case study. *Issues in Informing Science and Information Technology,* 331–349.

Tran, T. P., Nguyen, T. T. S., Tsai, P., & Kong, X. (2011). BSPNN: Boosted subspace probabilistic neural network for email security. *Artificial Intelligence Review, 35,* 369–382. Retrieved from https://link.springer.com/article/10.1007%2Fs10462-010-9198-2

Upadhyay, P. (n.d.). *Removing stop words with NLTK.* Retrieved from https://www.geeksforgeeks.org/removing-stop-words-nltk-python/

Wiedemann, P. M. (2013). Supporting non-experts in judging the credibility of risk assessments (CORA). *Science of the Total Environment,* 463–464, 624–630. doi:10.1016/j.scitotenv.2013.06.034

Zhang, S., Ou, X., & Caragea, D. (2015). Predicting cyber risks through National Vulnerability Database. *Information Security Journal: A Global Perspective, 24,* 194–206. Retrieved from http://www.cse.usf.edu/~xou/publications/su_infosec15.pdf

Appendix A: Balance Scale Training Data

[1 1 1]
[1 1 2]
[1 1 3]
[1 1 4]
[1 1 5]
[1 1 1]
[1 1 2]
[1 1 3]
[1 1 4]
[1 1 5]
[1 1 1]
[1 1 2]
[1 1 3]
[1 1 4]
[1 1 5]
[1 1 1]
[1 1 2]
[1 1 3]
[1 1 4]
[1 1 5]
[1 1 1]
[1 1 2]
[1 1 3]
[1 1 4]
[1 1 5]
[1 2 1]
[1 2 2]
[1 2 3]
[1 2 4]
[1 2 5]
[1 2 1]
[1 2 2]
[1 2 3]
[1 2 4]
[1 2 5]
[1 2 1]
[1 2 2]
[1 2 3]
[1 2 4]
[1 2 5]
[1 2 1]

[1 2 2]
[1 2 3]
[1 2 4]
[1 2 5]
[1 2 1]
[1 2 2]
[1 2 3]
[1 2 4]
[1 2 5]
[1 3 1]
[1 3 2]
[1 3 3]
[1 3 4]
[1 3 5]
[1 3 1]
[1 3 2]
[1 3 3]
[1 3 4]
[1 3 5]
[1 3 1]
[1 3 2]
[1 3 3]
[1 3 4]
[1 3 5]
[1 3 1]
[1 3 2]
[1 3 3]
[1 3 4]
[1 3 5]
[1 4 1]
[1 4 2]
[1 4 3]
[1 4 4]
[1 4 5]
[1 4 1]
[1 4 2]
[1 4 3]
[1 4 4]

[1 4 5]
[1 4 1]
[1 4 2]
[1 4 3]
[1 4 4]
[1 4 5]
[1 4 1]
[1 4 2]
[1 4 3]
[1 4 4]
[1 4 5]
[1 4 1]
[1 4 2]
[1 4 3]
[1 4 4]
[1 4 5]
[1 5 1]
[1 5 2]
[1 5 3]
[1 5 4]
[1 5 5]
[1 5 1]
[1 5 2]
[1 5 3]
[1 5 4]
[1 5 5]
[1 5 1]
[1 5 2]
[1 5 3]
[1 5 4]
[1 5 5]
[1 5 1]
[1 5 2]
[1 5 3]
[1 5 4]
[1 5 5]
[1 5 1]
[1 5 2]
[1 5 3]
[1 5 4]
[1 5 5]
[2 1 1]
[2 1 2]

[2 1 3]
[2 1 4]
[2 1 5]
[2 1 1]
[2 1 2]
[2 1 3]
[2 1 4]
[2 1 5]
[2 1 1]
[2 1 2]
[2 1 3]
[2 1 4]
[2 1 5]
[2 1 1]
[2 1 2]
[2 1 3]
[2 1 4]
[2 1 5]
[2 1 1]
[2 1 2]
[2 1 3]
[2 1 4]
[2 1 5]
[2 2 1]
[2 2 2]
[2 2 3]
[2 2 4]
[2 2 5]
[2 2 1]
[2 2 2]
[2 2 3]
[2 2 4]
[2 2 5]
[2 2 1]
[2 2 2]
[2 2 3]
[2 2 4]
[2 2 5]
[2 2 1]
[2 2 2]
[2 2 3]
[2 2 4]
[2 2 5]

[2 2 1]
[2 2 2]
[2 2 3]
[2 2 4]
[2 2 5]
[2 3 1]
[2 3 2]
[2 3 3]
[2 3 4]
[2 3 5]
[2 3 1]
[2 3 2]
[2 3 3]
[2 3 4]
[2 3 5]
[2 3 1]
[2 3 2]
[2 3 3]
[2 3 4]
[2 3 5]
[2 3 1]
[2 3 2]
[2 3 3]
[2 3 1]
[2 4 1]
[2 4 2]
[2 4 3]
[2 4 4]
[2 4 5]
[2 4 1]
[2 4 2]
[2 4 3]
[2 4 4]
[2 4 1]
[2 4 2]
[2 4 1]
[2 4 2]
[2 4 1]
[2 5 1]
[2 5 2]
[2 5 3]
[2 5 4]
[2 5 5]

[2 5 1]
[2 5 2]
[2 5 3]
[2 5 4]
[2 5 5]
[2 5 1]
[2 5 2]
[2 5 3]
[2 5 1]
[2 5 2]
[2 5 1]
[2 5 2]
[3 1 1]
[3 1 2]
[3 1 3]
[3 1 1]
[3 1 1]
[3 2 1]
[3 2 2]
[3 2 3]
[3 2 4]
[3 2 5]
[3 2 1]
[3 2 2]
[3 2 3]
[3 2 1]
[3 2 2]
[3 2 1]
[3 2 1]
[3 3 1]
[3 3 2]
[3 3 3]
[3 3 4]
[3 3 5]
[3 3 1]
[3 3 2]
[3 3 3]
[3 3 4]
[3 3 1]
[3 3 2]
[3 3 1]
[3 3 2]
[3 3 1]

[3 4 1]
[3 4 2]
[3 4 3]
[3 4 4]
[3 4 5]
[3 4 1]
[3 4 2]
[3 4 3]
[3 4 4]
[3 4 5]
[3 4 1]
[3 4 2]
[3 4 3]
[3 4 1]
[3 4 2]
[3 4 1]
[3 4 2]
[3 5 1]
[3 5 2]
[3 5 3]
[3 5 4]
[3 5 5]
[3 5 1]
[3 5 2]
[3 5 3]
[3 5 4]
[3 5 5]
[3 5 1]
[3 5 2]
[3 5 3]
[3 5 4]
[3 5 1]
[3 5 2]
[3 5 3]
[3 5 1]
[3 5 2]
[4 1 1]
[4 1 2]
[4 1 3]
[4 1 1]
[4 1 1]
[4 2 1]
[4 2 2]

[4 2 3]
[4 2 4]
[4 2 5]
[4 2 1]
[4 2 2]
[4 2 3]
[4 2 1]
[4 2 2]
[4 2 1]
[4 2 1]
[4 3 1]
[4 3 2]
[4 3 3]
[4 3 4]

Appendix B: Balance Scale Test Data

[2 3 4]
[2 3 5]
[2 3 2]
[2 3 3]
[2 3 4]
[2 3 5]
[2 4 5]
[2 4 3]
[2 4 4]
[2 4 5]
[2 4 3]
[2 4 4]
[2 4 5]
[2 4 2]
[2 4 3]
[2 4 4]
[2 4 5]
[2 5 4]
[2 5 5]
[2 5 3]
[2 5 4]
[2 5 5]
[2 5 3]
[2 5 4]
[2 5 5]
[3 1 4]
[3 1 5]
[3 1 2]
[3 1 3]
[3 1 4]
[3 1 5]
[3 1 2]
[3 1 3]
[3 1 4]
[3 1 5]
[3 1 1]
[3 1 2]
[3 1 3]
[3 1 4]
[3 1 5]
[3 1 1]

[3 1 2]
[3 1 3]
[3 1 4]
[3 1 5]
[3 2 4]
[3 2 5]
[3 2 3]
[3 2 4]
[3 2 5]
[3 2 2]
[3 2 3]
[3 2 4]
[3 2 5]
[3 2 2]
[3 2 3]
[3 2 4]
[3 2 5]
[3 3 5]
[3 3 3]
[3 3 4]
[3 3 5]
[3 3 3]
[3 3 4]
[3 3 5]
[3 3 2]
[3 3 3]
[3 3 4]
[3 3 5]
[3 4 4]
[3 4 5]
[3 4 3]
[3 4 4]
[3 4 5]
[3 4 3]
[3 4 4]
[3 4 5]
[3 5 5]
[3 5 4]
[3 5 5]
[3 5 3]
[3 5 4]
[3 5 5]
[4 1 4]

[4 1 5]
[4 1 2]
[4 1 3]
[4 1 4]
[4 1 5]
[4 1 2]
[4 1 3]
[4 1 4]
[4 1 5]
[4 1 1]
[4 1 2]
[4 1 3]
[4 1 4]
[4 1 5]
[4 1 1]
[4 1 2]
[4 1 3]
[4 1 4]
[4 1 5]
[4 2 4]
[4 2 5]
[4 2 3]
[4 2 4]
[4 2 5]
[4 2 2]
[4 2 3]
[4 2 4]
[4 2 5]
[4 2 2]
[4 2 3]
[4 2 4]
[4 2 5]
[4 3 5]
[4 3 1]
[4 3 2]
[4 3 3]
[4 3 4]
[4 3 5]
[4 3 1]
[4 3 2]
[4 3 3]
[4 3 4]
[4 3 5]

[4 3 1]
[4 3 2]
[4 3 3]
[4 3 4]
[4 3 5]
[4 3 1]
[4 3 2]
[4 3 3]
[4 3 4]
[4 3 5]
[4 4 1]
[4 4 2]
[4 4 3]
[4 4 4]
[4 4 5]
[4 4 1]
[4 4 2]
[4 4 3]
[4 4 4]
[4 4 5]
[4 4 1]
[4 4 2]
[4 4 3]
[4 4 4]
[4 4 5]
[4 4 1]
[4 4 2]
[4 4 3]
[4 4 4]
[4 4 5]
[4 4 1]
[4 4 2]
[4 4 3]
[4 4 4]
[4 4 5]
[4 5 1]
[4 5 2]
[4 5 3]
[4 5 4]
[4 5 5]
[4 5 1]
[4 5 2]
[4 5 3]

[4 5 4]
[4 5 5]
[4 5 1]
[4 5 2]
[4 5 3]
[4 5 4]
[4 5 5]
[4 5 1]
[4 5 2]
[4 5 3]
[4 5 4]
[4 5 5]
[4 5 1]
[4 5 2]
[4 5 3]
[4 5 4]
[4 5 5]
[5 1 1]
[5 1 2]
[5 1 3]
[5 1 4]
[5 1 5]
[5 1 1]
[5 1 2]
[5 1 3]
[5 1 4]
[5 1 5]
[5 1 1]
[5 1 2]
[5 1 3]
[5 1 4]
[5 1 5]
[5 1 1]
[5 1 2]
[5 1 3]
[5 1 4]
[5 1 5]
[5 1 1]
[5 1 2]
[5 1 3]
[5 1 4]
[5 1 5]
[5 2 1]

[5 2 2]
[5 2 3]
[5 2 4]
[5 2 5]
[5 2 1]
[5 2 2]
[5 2 3]
[5 2 4]
[5 2 5]
[5 2 1]
[5 2 2]
[5 2 3]
[5 2 4]
[5 2 5]
[5 2 1]
[5 2 2]
[5 2 3]
[5 2 4]
[5 2 5]
[5 2 1]
[5 2 2]
[5 2 3]
[5 2 4]
[5 2 5]
[5 3 1]
[5 3 2]
[5 3 3]
[5 3 4]
[5 3 5]
[5 3 1]
[5 3 2]
[5 3 3]
[5 3 4]
[5 3 5]
[5 3 1]
[5 3 2]
[5 3 3]
[5 3 4]
[5 3 5]
[5 3 1]
[5 3 2]
[5 3 3]
[5 3 4]

[5 3 5]
[5 3 1]
[5 3 2]
[5 3 3]
[5 3 4]
[5 3 5]
[5 4 1]
[5 4 2]
[5 4 3]
[5 4 4]
[5 4 5]
[5 4 1]
[5 4 2]
[5 4 3]
[5 4 4]
[5 4 5]
[5 4 1]
[5 4 2]
[5 4 3]
[5 4 4]
[5 4 5]
[5 4 1]
[5 4 2]
[5 4 3]
[5 4 4]
[5 4 5]
[5 4 1]
[5 4 2]
[5 4 3]
[5 4 4]
[5 4 5]
[5 5 1]
[5 5 2]
[5 5 3]
[5 5 4]
[5 5 5]
[5 5 1]
[5 5 2]
[5 5 3]
[5 5 4]
[5 5 5]
[5 5 1]
[5 5 2]

[5 5 3]
[5 5 4]
[5 5 5]
[5 5 1]
[5 5 2]
[5 5 3]
[5 5 4]
[5 5 5]
[5 5 1]
[5 5 2]
[5 5 3]
[5 5 4]
[5 5 5]