ISSN 2278-3091

Volume 10, No.3, May - June 2021

International Journal of Advanced Trends in Computer Science and Engineering

Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse1571032021.pdf

https://doi.org/10.30534/ijatcse/2021/1581032021

Categorization of CVE Based on Vulnerability Software By Using Machine Learning Techniques

SqnLdr (R) Aneela Kiran, Dr.Samina Rajper, Dr. Riaz Ahmed Shaikh, Imdad Ali Shah, Shahid Hussain Danwar

Department of Computer Science Shah Abdul University, Khairpur. Sindh, Pakistan

Aneelakiranansari73@gmail.com,Samina.rajper@salu.edu.pk,,riaz.shaikh@salu.edu.pk, imdad.shah@salu.edu.pk,shahid.danwar@salu.edu.pk

ABSTRACT

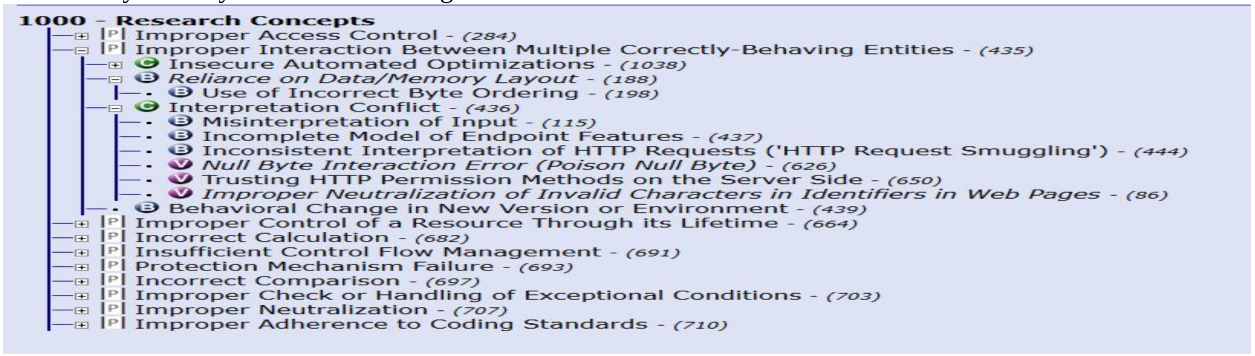
Public Platform is designed as an online website for researchers to collect reliable data for the study. NVD plays a significant role in analyzing The result of analysis in association influence metrics CVSS, type of CWE and applicability reports weakness CPE. The vulnerability testing is not performed by NVD while third-party security researchers and vulnerability controllers give information that has been assigned these attributes. ML plays a significant part in our daily life for the classification of huge data and is giving fruitful results. Because of that result, major steps have been made against criminal activities or unauthorized use of electronic data and protect the data from attackers. The major goal of this research is to categorize CVE Based Vulnerability Software throughout the last two years, 2019-2020.The findings of this study were used to ML for the categorization of CVE and compared and will open door for the fresh researchers and professionals.

Key words: ML.CVE, CPE, CWE, CVSS, and Vulnerability Categorization.

1. INTRODUCTION

Software vulnerability can be explained as software faults, which is to be exploited as a result of security attacks. Security studies used data from a vulnerability database to study for the discovery of new weaknesses for fitting the detection times, when new weaknesses were discovered for predicting. This study provides a complete overview of the enormous range of applications that have counted their heart an ML problem and brings some results of interest to the zoo of problems. After that, we would deliberate some basic tools from statistics and probability theory. Our research is about finding the vulnerability in light of description from text classifying vulnerability severity will measure using the CVSS Weakness Enumeration is to be found a hierarchical list of weakness types that exposures can classify. The scoring weakness and vulnerability types' information is accessible on the NVD. There is much existing research about weakness text; the only classification is limited because of a narrow area [1],[2],[3],[4].

The major goal of this research is to categorize CVE Based Vulnerability Software throughout the last two years, 2019-2020.The findings of this study were used to ML technique for the categorization of CVE and compared two years. The result of this study will open door for fresh researchers and professionals Figure-1.



```
1000 - Research Concepts
 |—⊞ P Improper Access Control - (284)
 |—⊟ P Improper Interaction Between Multiple Correctly-Behaving Entities - (435)
 |   |—⊞ C Insecure Automated Optimizations - (1038)
 |   |—⊟ B Reliance on Data/Memory Layout - (188)
 |   |   |—· B Use of Incorrect Byte Ordering - (198)
 |   |—⊟ C Interpretation Conflict - (436)
 |   |   |—· B Misinterpretation of Input - (115)
 |   |   |—· B Incomplete Model of Endpoint Features - (437)
 |   |   |—· B Inconsistent Interpretation of HTTP Requests ('HTTP Request Smuggling') - (444)
 |   |   |—· V Null Byte Interaction Error (Poison Null Byte) - (626)
 |   |   |—· V Trusting HTTP Permission Methods on the Server Side - (650)
 |   |   |—· V Improper Neutralization of Invalid Characters in Identifiers in Web Pages - (86)
 |   |—· B Behavioral Change in New Version or Environment - (439)
 |—⊞ P Improper Control of a Resource Through its Lifetime - (664)
 |—⊞ P Incorrect Calculation - (682)
 |—⊞ P Insufficient Control Flow Management - (691)
 |—⊞ P Protection Mechanism Failure - (693)
 |—⊞ P Incorrect Comparison - (697)
 |—⊞ P Improper Check or Handling of Exceptional Conditions - (703)
 |—⊞ P Improper Neutralization - (707)
 |—⊞ P Improper Adherence to Coding Standards - (710)
```

Figure 1: Shows the CWE research concept and hierarchical list[5]

2. LITERATURE REVIEW

An important problem and attracted much attention to automatically detecting software vulnerabilities, further, there were still not vulnerability competency or vulnerability detectors [6]. Many engineering tasks that greatly trust the classification/regression handcrafted software features, just like detection, software requirements, vulnerability discovery, malware detection, and code review[7]. Introduced a method for predicting an increasing number of software vulnerabilities and it was more exact as compared to VDM in most cases [8]. There are open-source software systems available on the internet today [9]. The information system mostly depends on the vulnerability's supervision procedure in the present day. [10]. Automated web application penetration emerged as a developed [11]. Present CASEI, complete a system that removes information about cyber security events from text and populates a semantic model of interest to integrate into a knowledge graph of cyber security data [12]. To work on the computer's security investigation and its results were benefited by information regarding characteristics of human's attacker behind security events [13]. There is a present info society depending on unfailing functionality of information arrangement, other side cyber-attacks increased from the years and damages have caused [14]. The growing frequency and severity of the cyber-attacks, in the previous years, is witnessed regarding the stealing the personal information of millions of people and ransomware attacks critical on the infrastructure of several countries [15]. An important problem has to tackle "vulnerability for short" as presented by much vulnerability reported daily [16] Figure-2 and -3.

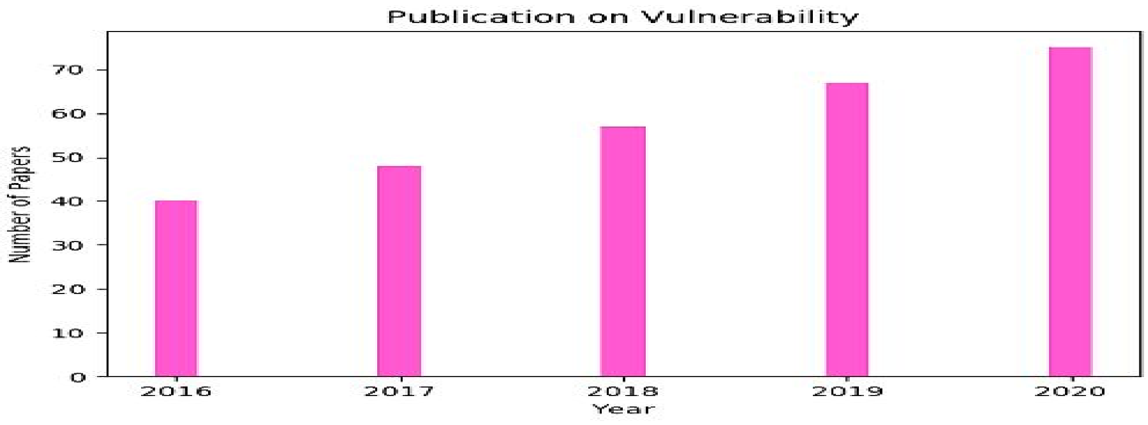


Figure-2 Show the NVD Data Researcher Used in Different year

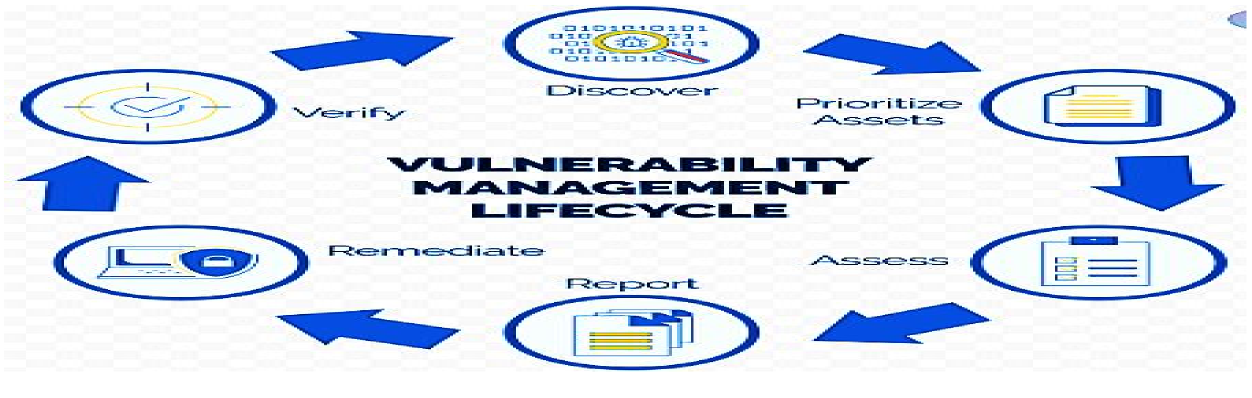


Figure-3 show Vulnerability Management Life Cycle

3. AUTOMATIC VULNERABILITY CLASSIFICATION

There are thousands of security vulnerabilities to be discovered every year in software production which is publicly appraised to common exposures database & vulnerabilities / discovered internally in the propriety code. The figure of software vulnerabilities was increasing each year, these discovered within proprietary code. Hence those weaknesses were the risk of an exploit, which may not compromise the result's system and info can leak. The employed method used for finding vulnerabilities practice software programs. The fuzzing techniques are to be used for finding bugs by executing in the study of Computers Security, vulnerability is a weakness that can be exploited from an attacker. While weakness may be any type of defect in a computer system that can be the chief reason for the compromise on information security. There is unknown vulnerability to the parties; they have the primary responsibility to correct them.That is to be called zero-day vulnerability. Defect reports are to be regularly written to a system where responsible authorities or parties could study and take required measures on a top priority basis in light of information about potential vulnerability which would be publicly visible and they are responsible to continuously monitor the status of defect corrections. Further, these reports consist of a few sentences long descriptions about software defects, and some defects reporting are publicly available. Take a +longer time to gather and distribute among non-security related software patches than those that are identified as vulnerabilities when they are reported. Resolved that after completely investigating the bug database for the MySQL database software a major number of previously unknown vulnerabilities were identified [17]. The National Vulnerability Database (NVD) comprises information about vulnerability explanations, SRF, security checklists, misconfigurations, and influence metrics. Database NVD is to be maintained by the U.S. government and the data is freely available at their data feeds. The data feeds are to be updated daily basis and vulnerability descriptions are relatively short sentences about vulnerabilities. These sentences can be used to identify potential vulnerabilities in any other text, including detecting reports in defect tracking systems. The security specialist's cruelty vulnerability severities, root cause, and required information are also available among the NVD data [18], [19].

a. CVSS BASE METRIC

The specification of CVSS version 3.1 was published in June 2019. The major difference to the previous version is that there are separate metrics to classify to calculate the actual score. The CVSS version 3.1 base metrics and classifications are cited in Table 1. CVSS3.1

Table-1 CVSS3.1 Metrics and Classifications

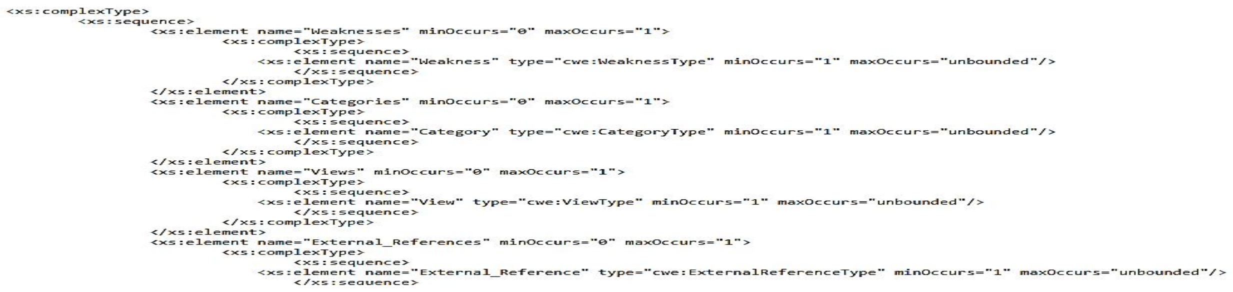
Metric	Description	Classifications	Numeric
Attack Vector AV	Reproduces framework which vulnerability exploitation is potential	Network N Adjacent A Local L Physical P	0,85 0,62 0,55 0,2
Attack Complexity AC	Measurement of attack complexity which required exploiting the vulnerability.	High H Low L Non N	0,44 0,77
Privileges Required PR	There are explanation level of privileges, the attacker would be possess before successfully exploiting vulnerability	High H Low L Non N	0,27 / 0,5 0,62 / 0,68 0,85
User Interaction UI	User's requirement has to captures other than attackers, to contribute in positive compromise of vulnerability.	Required R None N	0,62 0,85
Scope S	Possibility provides to the collection of privileges. These privileges are to be assigned based on some process of identification and authorization.	Changed C Unchanged U	Modifies privileges Required if the scope is changed
Confidentiality Impact C	Assessment effect the privacy of productively exploited vulnerability.	High H Low L Non N	0,56 0,22 0
Integrity Impact I	Assessment effect on the integrity of productively exploited vulnerability	High H Low L Non N	0,56 0,22 0
Availability Impact A	Assessment effect to available of positively exploited vulnerability"	High H Low L Non N	0,56 0,22 0

b. COMMON WEAKNESS ENUMERATION

The concept of CWE, software weakness types is a hierarchical list. The CWE is maintained by a nonprofit MITRE organization. Version 3.1 was published in June 2019. These lists are to be divided based on the concept view describes weakness & dependencies with each other to identify theoretical gaps within CWEs. The development concept view organizes weaknesses according to common architectural security tactics, on an object to identify potential mistakes that can recover in a software development process. In our research concept view selected as a basis to resolve CWE items Figure 4 &5.

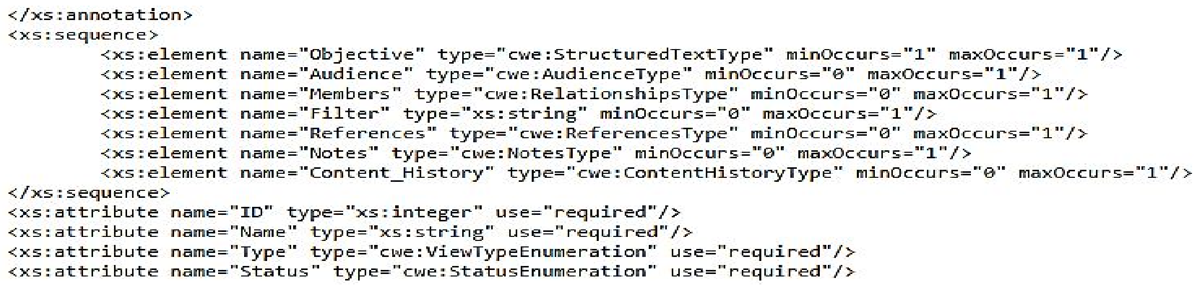
c. COMMON PLATFORM ENUMERATION

The complete CPE dictionary is freely available at the NVD website which is to be updated daily basis. The CPEs are mapped to vulnerabilities listed among the NVD data feeds. As formatted string binding within the NVD data used in this research [22].



```
<xs:complexType>
        <xs:sequence>
                <xs:element name="Weaknesses" minOccurs="0" maxOccurs="1">
                        <xs:complexType>
                                <xs:sequence>
                            <xs:element name="Weakness" type="cwe:WeaknessType" minOccurs="1" maxOccurs="unbounded"/>
                                </xs:sequence>
                        </xs:complexType>
                </xs:element>
                <xs:element name="Categories" minOccurs="0" maxOccurs="1">
                        <xs:complexType>
                                <xs:sequence>
                            <xs:element name="Category" type="cwe:CategoryType" minOccurs="1" maxOccurs="unbounded"/>
                                </xs:sequence>
                        </xs:complexType>
                </xs:element>
                <xs:element name="Views" minOccurs="0" maxOccurs="1">
                        <xs:complexType>
                                <xs:sequence>
                            <xs:element name="View" type="cwe:ViewType" minOccurs="1" maxOccurs="unbounded"/>
                                </xs:sequence>
                        </xs:complexType>
                </xs:element>
                <xs:element name="External_References" minOccurs="0" maxOccurs="1">
                        <xs:complexType>
                                <xs:sequence>
                            <xs:element name="External_Reference" type="cwe:ExternalReferenceType" minOccurs="1" maxOccurs="unbounded"/>
                                </xs:sequence>
```

Figure-4 show NVD Data



```
</xs:annotation>
<xs:sequence>
        <xs:element name="Objective" type="cwe:StructuredTextType" minOccurs="1" maxOccurs="1"/>
        <xs:element name="Audience" type="cwe:AudienceType" minOccurs="0" maxOccurs="1"/>
        <xs:element name="Members" type="cwe:RelationshipsType" minOccurs="0" maxOccurs="1"/>
        <xs:element name="Filter" type="xs:string" minOccurs="0" maxOccurs="1"/>
        <xs:element name="References" type="cwe:ReferencesType" minOccurs="0" maxOccurs="1"/>
        <xs:element name="Notes" type="cwe:NotesType" minOccurs="0" maxOccurs="1"/>
        <xs:element name="Content_History" type="cwe:ContentHistoryType" minOccurs="0" maxOccurs="1"/>
</xs:sequence>
<xs:attribute name="ID" type="xs:integer" use="required"/>
<xs:attribute name="Name" type="xs:string" use="required"/>
<xs:attribute name="Type" type="cwe:ViewTypeEnumeration" use="required"/>
<xs:attribute name="Status" type="cwe:StatusEnumeration" use="required"/>
```

Figure-5 shows the attribute

4. METHOD AND DATA

The ML Techniques, python tool and K-NN technique has been used for the categorization Figure 6,7, 8, 9, 10, 11& 12.

$$d_i = \sqrt{\sum_{i=1}^{p} (x_{2i} - x_{1i})^2}$$

Information:

x_1 = Sample Data

x_2 = Data Test / Testing

i = Variable Data

d = Distance formed

p = Dimension Data

Figure-6 Show the formula of K-NN algorithm

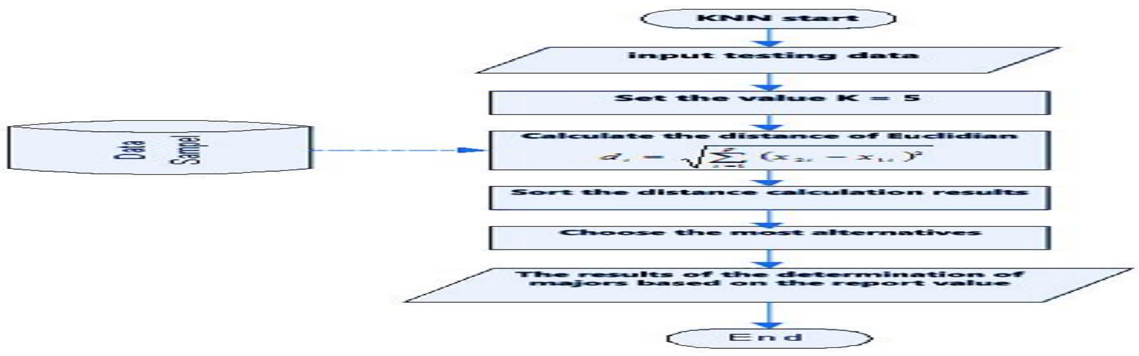

KNN start
input testing data
Set the value K = 5
Data Sampel
Calculate the distance of Euclidian
$d_i = \sqrt{\sum_{i=1}^{p} (x_{2i} - x_{1i})^2}$
Sort the distance calculation results
Choose the most alternatives
The results of the determination of majors based on the report value
End

Fig-7 Show the flowchart of K-NN classifier procedure

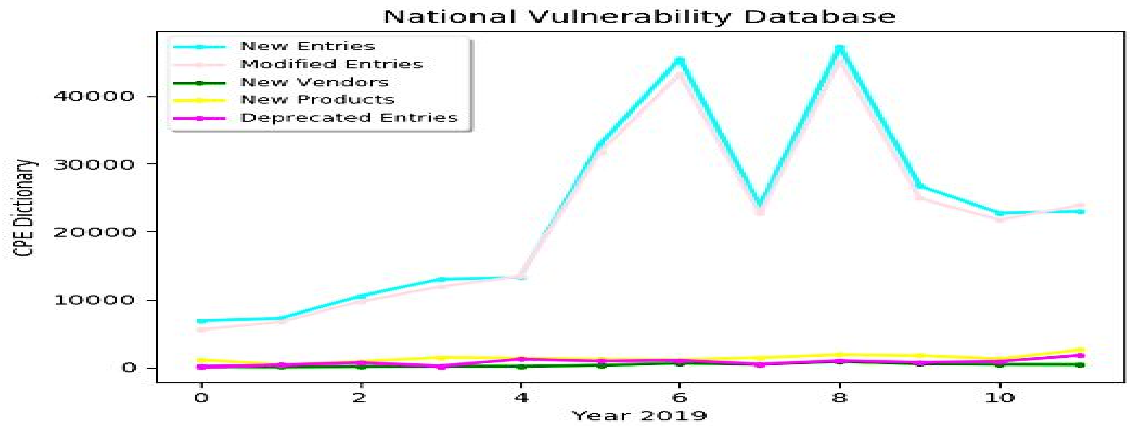


Figure-8shows the NVD data base

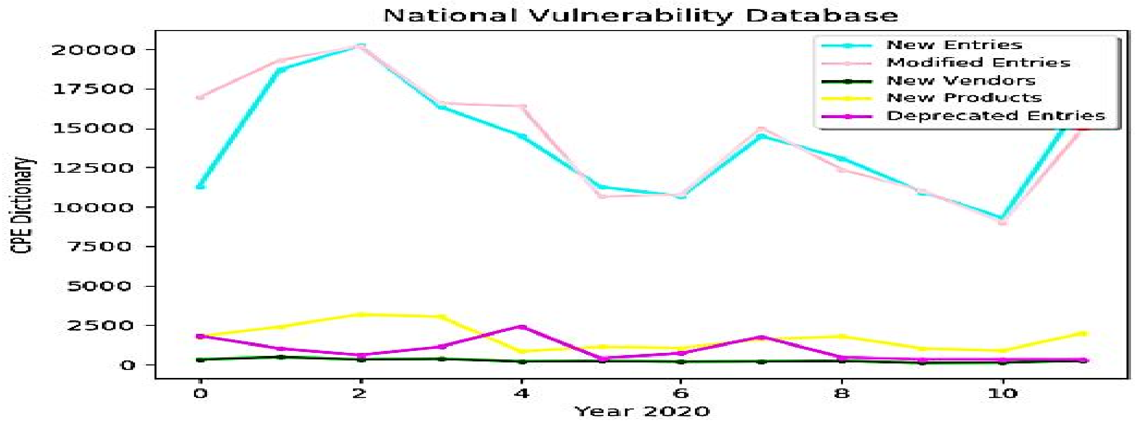


Figure-9 shows the NVD data base

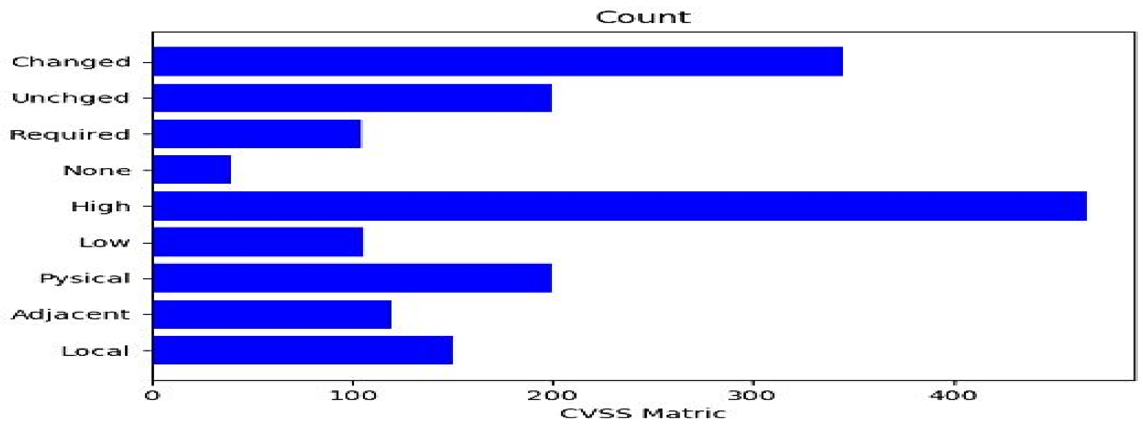


Figure-10 CVSS Distribution by Vectors

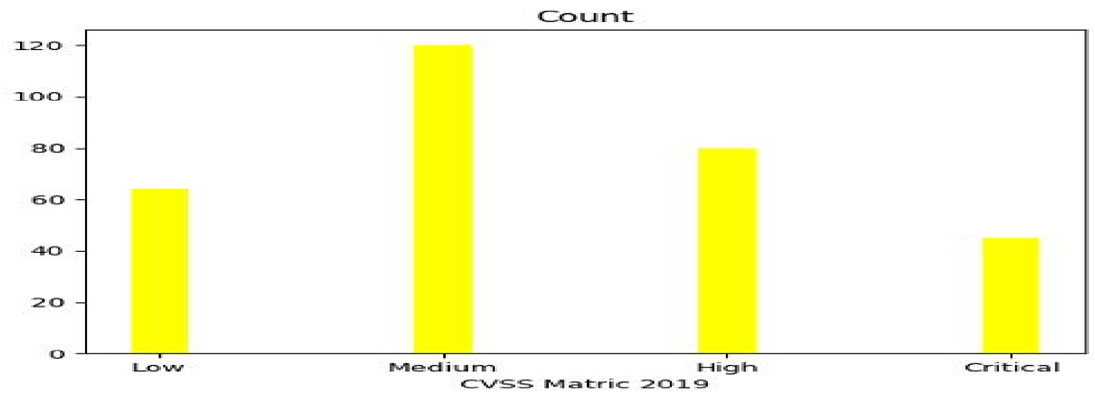


Figure-11Distribution of CVSS Based on Vulnerability Severity

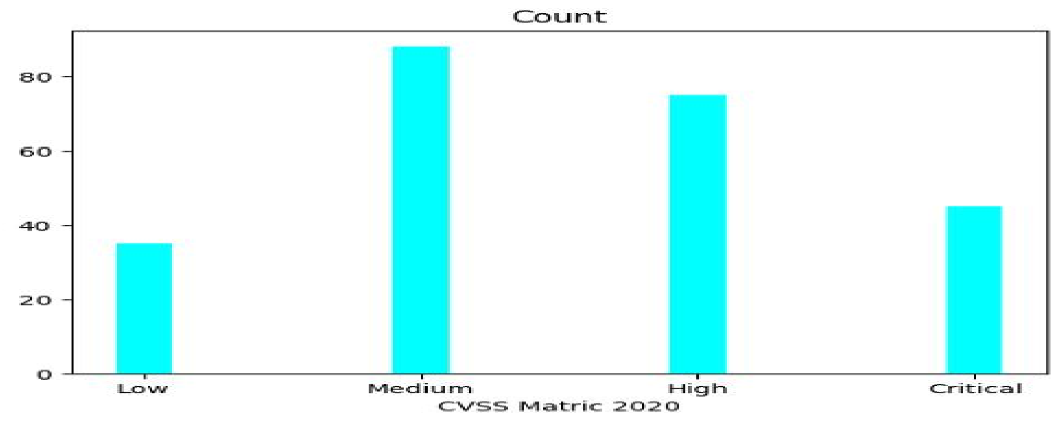


Figure-12Distribution of CVSS Based on Vulnerability Severity

5. CONCLUSION

Public Platform is designed as an online website for researchers to collect reliable data for the study. NVD plays a significant role in analyzing CVEs that are published in the CVE Dictionary. The result of analysis in association influence metrics CVSS, type of CWE and applicability reports weakness CPE. The vulnerability testing is not performed by NVD while third-party security researchers and vulnerability controllers give information that has been assigned these attributes. ML plays a significant part in our daily life for the classification of huge data and is giving fruitful results. Because of that result, major steps have been made against criminal activities or unauthorized use of electronic data and protect the data from attackers. The major goal of this research is to categorize CVE Based Vulnerability Software throughout the last two years, 2019-2020.The findings of this study were used to ML for the categorization of CVE and compared and will open door for the fresh researchers and professionals Figure-13 & -14.

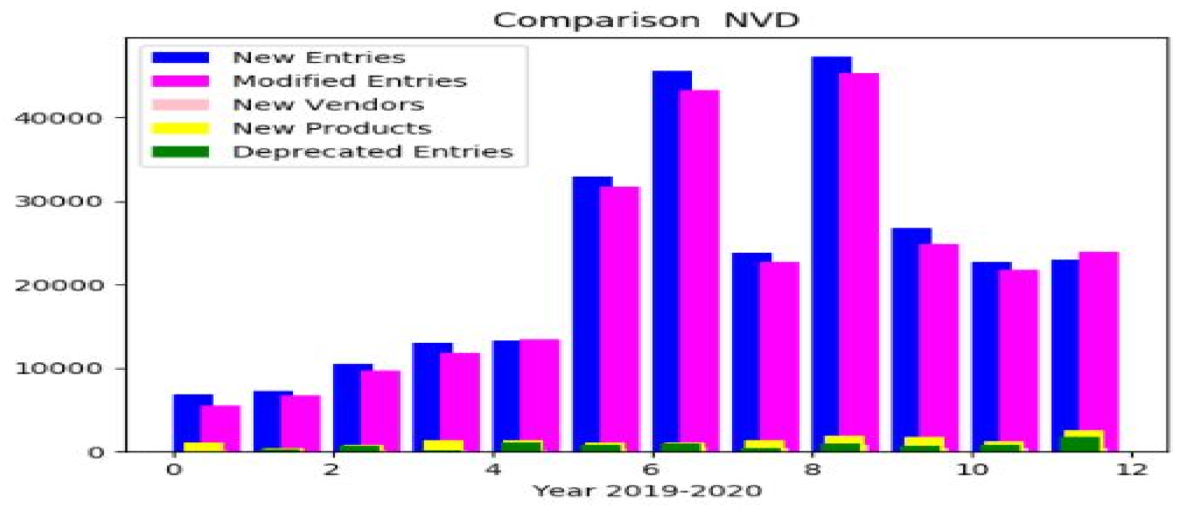


Figure-13 Describe the comparison NVD data

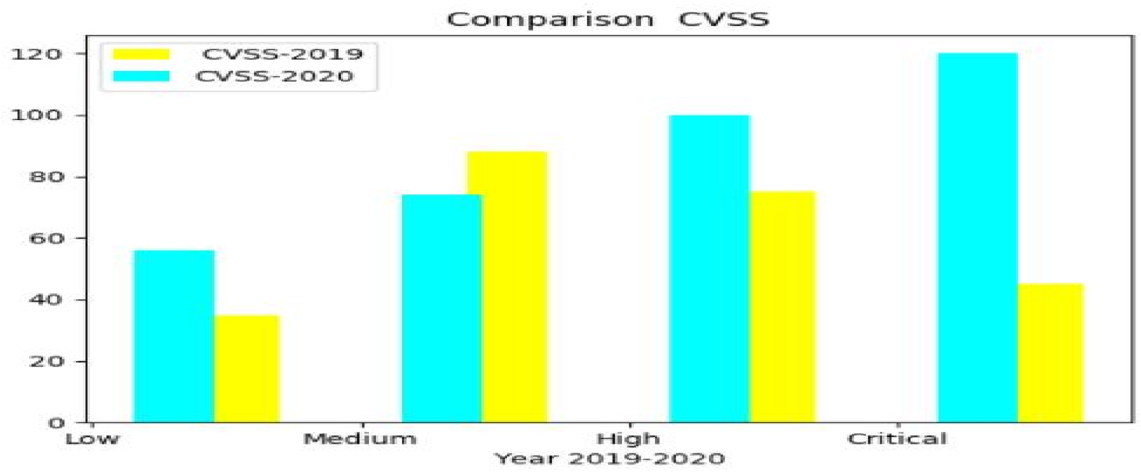


Figure-14 Describe the comparison of CVSS

REFERENCES

[1]. Yıldırım, M., Geçer, E., &Akgül, Ö. (2021). The impacts of vulnerability, perceived risk, and fear on preventive behaviours against COVID-19. *Psychology, health & medicine*, *26*(1), 35-43.https://www.tandfonline.com/doi/full/10.1080/13548506.2020.1776891

[2]Harer, J. A., Kim, L. Y., Russell, R. L., Ozdemir, O., Kosta, L. R., Rangamani, A., ... &Antelman, E. (2018). Automated software vulnerability detection with machine learning. *arXiv preprint arXiv:1803.04497*.https://arxiv.org/abs/1803.04497

[3]Lancet, T. (2020). Redefining vulnerability in the era of COVID-19. *Lancet (London, England)*, *395*(10230), 1089. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7270489/

[4] Li, Zhen, et al. "VulDeeLocator: A Deep Learning-based Fine-grained Vulnerability Detector." *arXiv preprint arXiv:2001.02350* (2020).https://arxiv.org/abs/2001.02350

[5] https://nvd.nist.gov/vuln

[6] Li, Zhen &Zou, Deqing&Xu, Shouhuai& Chen, Zhaoxuan& Zhu, Yawei& Jin, Hai. (2020). VulDeeLocator: A Deep Learning-based Fine-grained Vulnerability Detector. https://arxiv.org/abs/2001.02350

[7]Qiu, Y., Liu, Y., Liu, A., Zhu, J., &Xu, J. (2019). Automatic Feature Exploration and an Application in Defect Prediction. *IEEE Access*, *7*, 112097-112112.https://ieeexplore.ieee.org/abstract/document/8794540

[8]Harer, Jacob A., et al. "Automated software vulnerability detection with machine learning." *arXiv preprint arXiv:1803.04497* (2018). https://arxiv.org/abs/1803.04497

[9] Al-Msie'Deen, Ra'Fat. (2018). Automatic Labeling of the Object-oriented Source Code: The Lotus Approach. Science International-Lahore. 30. 45–48. https://arxiv.org/abs/1803.00048

[10] A. Dobrovoljc, D. Trček and B. Likar, "Predicting Exploitations of Information Systems Vulnerabilities Through Attackers' Characteristics," in *IEEE Access*, vol. 5, pp. 26063-26075, 2017. doi: 10.1109/ACCESS.2017.276906https://ieeexplore.ieee.org/abstract/document/8094238

[11]Seng, Lim &Ithnin, Norafida&Shaid, Syed. (2018). Automating Penetration Testing Within Ambiguous Testing Environment. International Journal of Innovative Computing. 8. 10.11113/ijic.v8n3.180. https://ijic.utm.my/index.php/ijic/article/view/180

[12]Taneeya Satyapanich, Tim Finin and Francis Ferraro, CASIE: Extracting Cybersecurity Event Information from Text, 34th AAAI Conference on Artificial Intelligence, New York, Feb. 2020. https://mdsoar.org/handle/11603/17206

[14]Polatidis, N., Pimenidis, E., Pavlidis, M., Papastergiou, S., &Mouratidis, H. (2018). From product recommendation to cyber-attack prediction: Generating attack graphs and predicting future attacks. *Evolving Systems*, 1-12.https://link.springer.com/article/10.1007/s12530-018-9234-z

[15]Goyal, P., Hossain, K. S. M., Deb, A., Tavabi, N., Bartley, N., Abeliuk, A. E., ...&Lerman, K. (2018). Discovering signals from web sources to predict cyber attacks. *arXiv preprint arXiv:1806.03342*.https://arxiv.org/abs/1806.03342

[16] Li, Zhen &Zou, Deqing&Xu, Shouhuai& Jin, Hai& Zhu, Yawei& Chen, Zhaoxuan& Wang, Sujuan& Wang, Jialai. (2018). SySeVR: A Framework for Using Deep Learning to Detect Software Vulnerabilities. https://arxiv.org/abs/1807.06756

[17]Almukaynizi, M., Marin, E., Nunes, E., Shakarian, P., Simari, G. I., Kapoor, D., &Siedlecki, T. (2018, November). Darkmention: A deployed system to predict enterprise-targeted external cyberattacks. In *2018 IEEE International Conference on Intelligence and Security Informatics (ISI)* (pp. 31-36). IEEE.https://ieeexplore.ieee.org/abstract/document/8587334

[18]Ruohonen, J., 2018, December. An empirical analysis
of vulnerabilities in Python packages for web applications. In *2018 9th International Workshop on Empirical Software Engineering in Practice (IWESEP)* (pp. 25-30). IEEE. https://ieeexplore.ieee.org/abstract/document/8661215

[19] Q. Chen, L. Bao, L. Li, X. Xia and L. Cai, "Categorizing and Predicting Invalid Vulnerabilities on Common Vulnerabilities and Exposures," *2018 25th Asia-Pacific Software Engineering Conference (APSEC), Nara, Japan, 2018, pp. 345-354. doi: 10.1109/APSEC.2018.00049* https://ieeexplore.ieee.org/abstract/document/8719428