VILNIUS UNIVERSITY
FACULTY OF MATHEMATICS AND INFORMATICS
MODELLING AND DATA ANALYSIS
MASTER'S STUDY PROGRAMME

Vulnerability Testing and Analysis of Educational Institution Websites within Lithuania

Master's thesis

Author: Goda Klaudija Kovalenkinaitė
VU email address: goda.kovalenkinaitė@mif.stud.vu.lt
Supervisor: Professor Julius Venskus

Vilnius

2023

Contents

1 Abstract

The security of web application is an integral part of ensuring the safety of humans and their data in today's increasingly connected world. Web-based systems are often a target to cyber attacks, thus vulnerability testing is a critical process in computer systems and networks security. Automated scanners, two of which been chosen for the thesis, ZAP and N-Stalker, have been an effective tool for identifying vulnerabilities in cost-less and timely manner. This thesis presents a study of the use of automated scanners on 100 websites within educational institutions. It can be challenging to determine which tool is best suited for a particular testing scenario, therefore the comparison between the scanners was done using clustering methods of DBSCAN, K-Means and Gaussian Mixture. Therewithal scaling and dimensionality reduction of t-SNE and MDS were applied for better implications and visualizations of the data. An observation occurred that DBSCAN provided highest Silhouette Scores and Calinksi-Harabasz index, thus performing better than K-Means and Gaussian Mixture algorithms. It was also concluded that from 56% to 92% of websites fell into same clusters, across the three clustering algorithms, illustrating the similarities between the scanner vulnerability detection.

Keywords: Vulnerability Testing, Automated Testing, ZAP, N-Stalker, DBSCAN, K-Means, Gaussian Mixture, t-SNE, MDS.

Glossary

Authentication - Validating of ones identity or the genuinity in documents. Page:12

DBMS - Database Management System credentials hold username and password for authentication. Page:14

GUI-based environment stands for Graphical User Interface based environment that uses graphical features such as windows, buttons, icons, dialog boxes etc. Page:12

Heuristics approach is an approach to a problem that is quick, efficient and effective to reach short-term goals. Page:13

Metagoofil is a tool that extracts and gathers metadata information from the documents that are stored in websites. Page:14

Metasploit is a penetration testing platform that is Ruby-based. Page:11

OAST testing - Out-of-band application security testing is a method of finding vulnerabilities using external servers. Page:15

PCI Data Security - Payment Card Industry Data Security focuses on reducing payment card fraud. Page:11

port is an endpoint of service. More than one server can coexist on a computer with different port numbers.. Page:14

SDLC - Secure Development Life Cycle is a framework that maps development process, made of phases: planning and requirements, design, coding, testing and results and release and maintenance.. Page:11

Smartcard and client digital certificates - Client digital certificates are stored in Smart cards that are hardware devices. Digital certificates are used to prove authenticity. Smart card authenticates using PIN, with two factor security. Page:13

TLS - Transport layer security is a security protocol that provides privacy and data security over the computer networks. Page:12

Web socket - A protocol based on TCP that supports other protocols, which of those can run on top of the Web Socket. Page:13

2 Introduction

Websites need security systems in order to protect themselves from various threats that would compromise the integrity and confidentiality of the websites and their underlying systems. Hackers, malware and unauthorized users can be a source of those threats. Websites, without adequate security measures, are at risk of being exploited by attackers who would potentially gain access to sensitive data, disrupt the functionality of the websites and use the websites as the stepping stone to access other systems on the network. Sensitive data, such as, identification and personal details or financial information and more, must be protected to comply with privacy regulations to also maintain the trust of users. Implementing security measures, such as authentication and authorization controls, input validation, and secure coding practices, can help to ensure the confidentiality and integrity of data and prevent unauthorized access to the website.

Vulnerability testing of web applications is pivotal for securing web-based systems and minimizing the likelihood of cyber threats. It involves identifying, analyzing and reporting vulnerabilities present in systems or applications. Manual and automated testing are two of the methods of evaluating the web application security. Manual testing consists of examining the source code for potential vulnerabilities while automated testing has a specialized tool doing that instead. Vulnerability testing occurs to be of great importance in a robust security program.

In this research, automated testing was performed using automated scanners ZAP and N-Stalker. They are a software tool that are designed to automatically scan a computer system, networks and websites for vulnerabilities. A variation of techniques are used to identify these vulnerabilities, such as, examining source code, testing input fields for common attack vectors and analyzing network traffic for indications of security weaknesses. It is a great alternative to manual testing where more skills and time would be essential. It is important to note that automated scanners have limitations which include the potential for false positives and false negatives.

The 100 websites within Lithuanian education institutions were scanned using two scanners, Zap and N-Stalker and the data was collected. The min-max scaling was applied and each data set was run thought a dimensional reduction with t-SNE and MDS methods. Clustering methods: DBSCAN, K-Means and Gaussian Mixture were performed due to their different technologies, assumingly providing us with a range of results. The clusters created within individual data sets were compared between the two scanners, providing a percentage of difference. For visualization, scatter plots were created to evaluate the clusters produced for different combination of parameters of different clustering algorithms. The said parameters were also evaluated using Silhouette scores and Calinski-Harabasz index.

The expectation of this research is to have both scanners detect same type of vulnerabilities for most if not all websites. Therefore, websites would be places into same clusters for both ZAP and N-Stalker scanner datasets and result in small percentage difference between them.

Our research is based on a combination of related work, security software tools, the progress of collecting data, the methodology used to analyse the datasets, results and evaluation, conclusions and finally propositions of future work.

2.1 Research Object

Lithuanian educational institution vulnerability detection and the analysis. Investigate and compare different scanner provided results.

2.2 Research Aim and Objective

1. Perform literature review and further analysis to choose software tools.
2. Choose the web sites that would be analysed for this project. Using the chosen software tools, scan the chosen websites.
3. Perform the analysis of scanners, by analysing the data collected and finally compare using data science techniques.

3 Related Work

Testing the security of the assets and devices is prevalent in this day of age and is at its utmost importance. As the focus is on the websites, that also tend to have security holes endlessly, many research papers have already covered the methods of testing the security and checking for its vulnerabilities.

3.1 Vulnerability detection

Rina Elizabeth Lopez de Jimenez in an article [17] , discusses pentesting using ethical hacking. The different types of pentesting that could be used, varying from black box to white box to grey box penetrative testing,

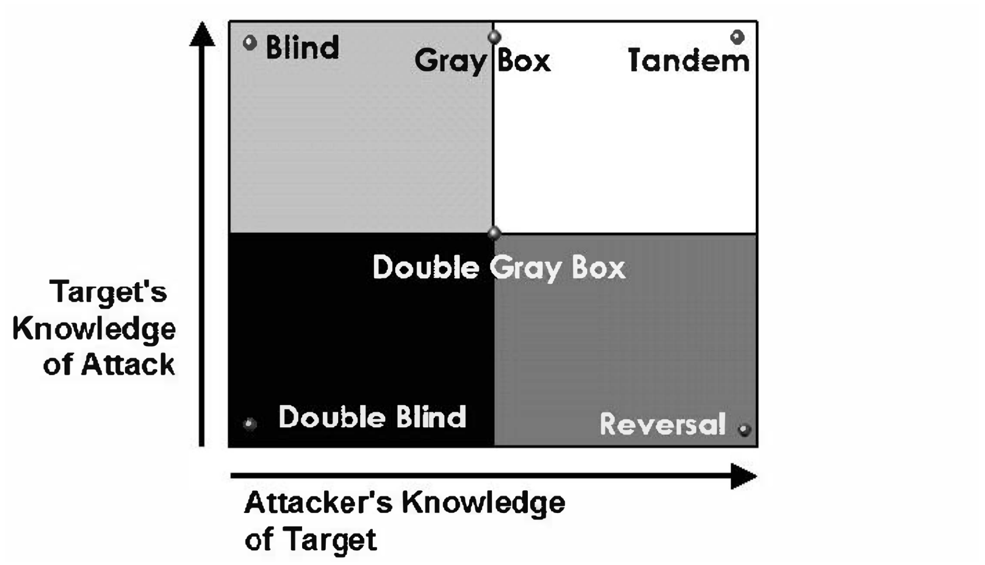


Figure 1: CNN model architectures (source: [17])

that discover a range of vulnerabilities withing the sites, such as Cross-Site Scrapping on web applications and SQL Injections through database attacks. XSS, shortened for Cross-Site Scrapping, [9], is one of the most common vulnerabilities found where an attacker is capable of injecting a code into a victim's web application and compromising data and more malicious intents. SQL Injection on the other hand has the attackers focus on gaining access into the databases of the website to therefore corrupt it or steal sensitive information [10]. The author also analyses the software tools and informs on their differences and their applicability in finding certain vulnerabilities and performing different kinds of penetrative tests.

The research paper 'State of the Art: Automated Black-Box Web Application Vulnerability Testing' [4], based their research on black box scanners specifically. The authors used eight scanners from 'the most-established security companies in the industry' on to three web applications Drupal, phpBB, and Wordpress. The vulnerabilities were classified and the scanner performance was compared, showing how different scanners had strengths in certain vulnerabilities and no scanner was the best in every single of them. Those black box scanners with leading statistics in certain individual vulnerability classification detection and low numbers in false positives, lacked strength in other categories. For instance, XSS and SQL Injection are detected best by the same scanners that are at the bottom in detecting Session Management vulnerabilities. Though, in general scanners have great performance in detection of “reflected” cross-site scripting of rate over 60%. This implies to us that for our research multiple scanners would have to be used for the best evaluate of the websites security.

Prof. Sangeeta Nagpure and Sonal Kurkure in their reseach paper [22], have described vulnerability assessment and penetration testing as two separate items. Vulnerability assessment is done by a black-box scanner first, in this article Acunetix, OWASP and PortSwigger vendors were used, and the use of penetrative testing afterwards. Penetration testing is explored by an ethical hacker utilizing their knowledge to exploit the loopholes of the websites that were found by black box scanners first. The test cases are then formed. Vulnerability assessment is evidently an automated process, conversely penetration testing is not. Later a practical was done on two websites, an e-commerce and cloud application to further indicate the necessity in manual penetration testing. The vulnerabilities that were detected were from cross-site scripting(XSS), SQL Injection, Clickjacking to Cross site request forgery, of which were described also in the paper[22]. The manual testing provided 100% of vulnerability detection across these websites, however automated testing provided 0% in Cross site request forgery and Authentication Bypass[22]. The results revealed the importance in penetrative testing, whoch will be taken into consideration for this research paper.

3.2 Vulnerability detection within different regions

The vulnerabilities were measures across couple of international websites and incorporated withing few research papers as ’Measuring Vulnerabilities of Bangladeshi Websites’ [21] and ’Security Evaluation of Saudi Arabia’s Websites Using Open Source Tools’ [2]. The former paper used black box testing by using penetration methods and then white box testing by source code analysis mechanisms, which is a manual process of analysing the whole code. The author didn’t specify the website sample size however 34 web applications were covered in detail. The black box testing was done on Kali Linux, an operative system with already included many penetrative testing tools such as Nmap, Merasploit and more of black box scanners. Conclusions were made on 64% of tested websites being subject to vulnerabilities, specifically to SQL Injection, XXS and Transport Layer Security (TLS). More so, source code analysis showed ’back-end technologies used in some government websites are more than ten years old in syntax and versions’.

Nonetheless, the latter paper, focused on 100 government websites and 50 commercial ones by applying open source tools, of which W3af and Skipfish were chosen, both also black box scanners. 61% of the Saudi Arabian websites were found to be vulnerable to Clickjacking. 17.5% vulnerable to SQL injection, compared to 56% of Bangladeshi web applications. Though, the ratio between commercial and government sites was unexplained, and could of lead to a misleading conclusions within comparison between them. Both papers have show common denominator on government websites tending to more vulnerabilities and higher security risks. Securing the websites would require server details/equipment to be up to date with security patches. These papers also signified the magnitude of security issues and similarities amongst different countries.

A report for Sudanese government websites has also been formed [23]. Since Sudanese government websites lack feasible frameworks 'for national standards or guidelines for the development of national sites' and do not operate based on international security standards, this research thesis proposed methods for vulnerability discovery. The frameworks that were used were 'whois' 'matesploit' and 'owasp' for gathering information, testing websites and penetration report generation respectively. The three websites that were used for the practical, the results were conclusive on the website security weaknesses containing 17 vulnerabilities within the first website that was tested; 11 vulnerabilities within the second website and 9 vulnerabilities within the third. The vulnerability classifications ranging from SQL Injection, Local File inclusion, BSQLi [16], multiple XSS/CSRF, Frontend XSS–PHP_SELF not properly filtered vulnerability. This could motivate into analysis of security of Lithuanian websites, since research has not yet been done on it.

3.3 Vulnerability detection using dynamic analysis

Dynamic websites differ from static websites, by having the ability to be manipulated by the viewer, whereas static websites don't. Examples of dynamic websites are e-commerce sites, calendars, blogs, Facebook etc., and static websites such as Wikipedia, and other read-only sites. The research paper 'Detection of Security Threats to Modern Websites' [13], has completed a dynamic analysis of the website security. Errors are looked while executing data in real-time and thus evaluating the program, compared to looking at code offline. Tools that implemented dynamic analysis of the asynchronous forms, appeared to have an inability in detecting the vulnerabilities. For the dynamic analysis to have been effective, the program must be executed with enough input vectors that would cover all possible outputs, however the research paper proposed the assumption that compiling a set of test inputs on the websites that use HMLHttpRequest mechanism created that inability to detect the vulnerabilities. D.V.Ivanova, D.A.Moskvina,and G.S.Kubrina have propositioned a method which interprets JavaScript scrips, 'imitating the work of modern web browsers'. Multiple black box scanners were used, of which Accunetix, WAScan, WVS and SaaS ervice Positive Technologies were not capable of detecting the vulnerable form with dynamic formation, yet w3af and Saas service Detectify was to a certain extent. Finally YAVWS (QtWebEngineLoader), a dynamic analysis tool, successfully detected the vulnerabilities in the websites before and after JavaScript scrips obfuscation, something the previous two scanners had an issue with.

3.4 Vulnerability detection using design flaws

Authors of the research paper 'Analyzing Websites for User-Visible Security Design Flaws' [7] have explored the security of 214 U.S financial websites rather in their user-visible designs. Design flaws are a cause of mistakes that were made during the design phase of the websites, which lead to security issues for the users. The paper focused on break in the chain of trust, insecure pages containing secure login options, contact information on the websites, inadequate policies for user ids and passwords and E-Mailing security sensitive information insecurely, where high percentages of these vulnerabilities were presented in the paper, 74% of sites had at least one flaw, 10% had all five flaws and 24% of these sites were free of the design flaws. The paper explored automatic detection of these flaws using automated tools, after downloading the websites, using wget, for the ability and comfortability of static view. The automated tools did not eliminate the false positives so the author did using manual examination of the data. Each website analysed the HTML pages for certain patterns and design flaws were identified. Automated tools showed, 17% of websites have not let the user know of the potential security issue in use of third-party sites within the websites.

3.5 Vulnerability detection using machine learning

Another way of detecting vulnerabilities within websites is with the use of machine learning. The research paper [12], has placed their focus in the medical field and the security of patients' portals. Web crawlers are one of the biggest sources of cyber-attacks that attack users by cracking passwords, gathering patients sensitive information and more. Navigational behaviour of crawlers and their attributes were analysed to separate known, malicious, unknown ethical and unethical crawlers. New model was developed for detecting these malicious web crawlers using machine learning and appropriate features for crawler detection that were gathered during the investigation. The investigation included the analysis of log files of compromised websites. The conclusions were made based on the evaluation of the accuracy of the models, showing how with the proper extraction of necessary features, the performance of the models, such as Support Vector Machine, Bayesian networks and Decision tree has increased.

Another research paper also using machine learning to aid with vulnerability detection [5] have designed a tool, Mitch, for black box detection of Cross-Site Request Forgery (CSRF) vulnerabilities based on the methology described in details in the paper [5]. The authors have tested the effectivness of the tool on top 10k Alexa, 20 websites with single sign-on access to make the security testing process with only 2 created accounts possible. 191 sensitive requests and 47 CSRF potential vulnerabilities were found of which 35 with 7 false positives were exploited to deduce vital security issues.

4 Security Software Tools

15 scanners to be discussed, are on the market at a current date, some are older, some are newer, some are free and others are not. Each scanner can differ in speed, costs, ability to handle different amount of data, production dates, reliability and amounts of false positives. Three scanners after the analysis have been chosen to proceed with the practical part of this research. The scanners from one to three will also be compared and analysed using statistical methods to confirm or disprove the effective of each.

Open source tools, were included in the current subsection and such are freely available software tools that have no necessity in commercial license and are accessible to the public. Open source tools are also free.

OWASP is an Open Web Application Security Project which is a non-profit organisation that works on software security, that also counts for reliable source of information.

The OWASP Top 10 associates to the top security vulnerabilities that the organisation has outlined to be critical. These include Broken Access Control, Cryptographic Failures, Injection, Insecure Design, Security Misconfiguration, Vulnerable and Outdated Components, Identification and Authentication Failures, Software and Data Integrity Failures, Security Logging and Monitoring Failures, Server-Side Request Forgery.

4.1 Acunetix

Acunetix is a web vulnerability scanner that analyses the website by thoroughly scanning it and extracting all the necessary information on the web application. The 'Acunetix Acu' sensor if enabled can withdraw the directories and files of the said website. Further then vulnerabilities are presented and a report is created.

Acunetix offers a 14 day trial and free penetration testing manual tools, but are not part of Acunetix product and need to be downloaded separately.

Acunetix Manual Tools contain the following modules: HTTP Editor, Subdomain Scanner, Target Finder, Blind SQL Injector, HTTP Fuzzer, Authentication Tester, Web Services Editor, and HTTP Sniffer; however are not part of an open-source project. The scanner works on Microsoft Windows operating system and can be used alongside other tools like Metasploit* exploitation framework, ZAP, w3af audit framework, etc. to broaden the information on the vulnerabilities detected.

The 14 day trial scan offers a few options from XSS, SQL Injection, Weak Passwords, high risk vulnerabilities or crawling only.

4.2 N-Stalker

N-Stalker offers an assessment of the 3rd party packages alongside web vulnerabilities. The said vulnerabilities include but not limited to "OWASP Top 10" and "PCI Data Security"* in addition to ensuring the SDLC*- Secure Development Life Cycle.

N-Stalker Web Application Security Scanner X Free Edition has more than 20,000 signature filled database. A signature being a footprint of a malicious attack on a system or computer network. A restricted amount of security checks are available in the free edition; HTTP Fingerprinting and File Backup Extensions analysers are few of the vulnerability checks of the scanner, as well as Cross-sire Scripting. Important to note than the free edition only scans up to 500 pages per each web application.

4.3 Grabber

Grabber is a web application scanner capable of detecting the following vulnerabilities: Cross-Site Scripting, SQL Injection (Blind SQL Injection module), File Inclusion, backup files check, AJAX check, Hybrid analysis/Crystal ball testing for PHP application using PHP-SAT, JavaScript source code analyzer, generation of a file for a statistical analysis.

It is a simple and free tool developed on python, however it is not fast and could not handle large web applications and only last updated in 2017. Python, BeautifulSoup and PyXML packages would need to be used in order to complete the security scans. The developer of the Grabber has implemented the scanner to be specifically good for the Blind SQL Injection, SQL Injection and File Inclusion and backup files tests.

4.4 Vega

Vega is a free open-source web vulnerability scanner. It is a program written on Java that performs security tests that also includes a GUI-based environment*. Vega is available for use on OS X, Windows and Linux. JavaScript was used for writing the detection modules, thus Vega an API written in JavaScript could be used as well.

It can be used to find SQL injection, blind SQL injections, header injection, directory listing, shell injection, cross-site scripting, remote file inclusion, String Format attacks and others. Other than that, Vega demonstrates ways to improve the security of TLS* Servers.

Nonetheless, Vega does portray some false positive, however, taking into account its a completely free tool and covers many vulnerability classifications, it is not far behind most prominent web scanners in the industry.

4.5 Zed Attack Proxy

Zed Attack Proxy(ZAP) is a penetration testing scanner developed by OWASP(Open Web Application Security Project). It is a tool that can be used on Linux, Windows and Mac OS.

The vulnerabilities that this scanner detects includes SQL injection, broken authentication or access control, sensitive data exposure, XSS, security misconfiguration, missing security headers, insecure deserialization. ZAP allows to either scan a whole website by inputting a URL or specific pages can be tested. Luckily it is a tool that is easy to use not only to scan the websites to detect possible threats but also offers Authentication* support, Web socket* support, Smartcard and client digital certificates* support and more.

4.6 Wapiti

Wapiti checks for website or web application security, and performs black-box testing which crawls the websites looking at scripts where data could be injected, but doesn't study the source code. The scanner performs a technique called fuzzing, where invalid, random data is provided into a computer program and monitors for crashes, memory leaks to see if there is vulnerability within scripts.

Wapiti provides many features, from SQL Injections, reflected and permanent XSS, file disclosure detection, command Execution detection, XXE (Xml eXternal Entity) injection,CRLF Injection, search for potentially dangerous files on the server and many more.

Wapiti is a command line application, thus more on a difficult side for beginners but could appear to be great help for professionals.

4.7 W3af

W3af is a python developed framework that secures web applications and also open-source. Its main features are that its a flexible and customizable tool for developers and hackers; It exploits main vulnerabilities like SQL injection and XSS and over 200 more using plugins, that are python codes that send HTTP requests to forms. W3af GUI also allows to send a custom version of these HTTP requests, craft them, cluster the HTTP requests and more.

4.8 WebScarab

WebScarab is a java based framework that focuses on HTTP based web applications. HTTP protocol would be of use to know to write necessary codes, thus it is more suitable for experts. The tools functionality can be extended by the plugins that it provides. One of the plugins helps take control of request and response when there is an observation of the traffic between a server and a browser by a proxy(intercepting proxy). It also contains a spider, it can fine new URLs connected to the target website automatically. Moreover, WebScarab can extract HTML and scripts of the page.

Regular vulnerabilities like SQL Injection, XSS, CRLF and many other can be detected using fuzzer tool.

4.9 Skipfish

Skipfish is a penetration tool in Kali Linux written on C. Very good for HTTP handling and using minimum CPU, where 2,000 requests are easily handles without addition of load on to the CPU. A Heuristics approach* is used to crawl the websites and test them, where each page can be checked for various vulnerabilities. Skipfish offers few false positives and a final report.

Skipfish is free, open source and available on GitHub. Includes 15 modules, such as Metagoofil*. Also tracks enumeration.

4.10 SQLMap

SQLMap is another open-source security testing tool. It focuses on SQL injection vulnerabilities 6 different techniques: time-based blind, Boolean-based blind, error-based, UNION query, stacked queries and out-of-band.

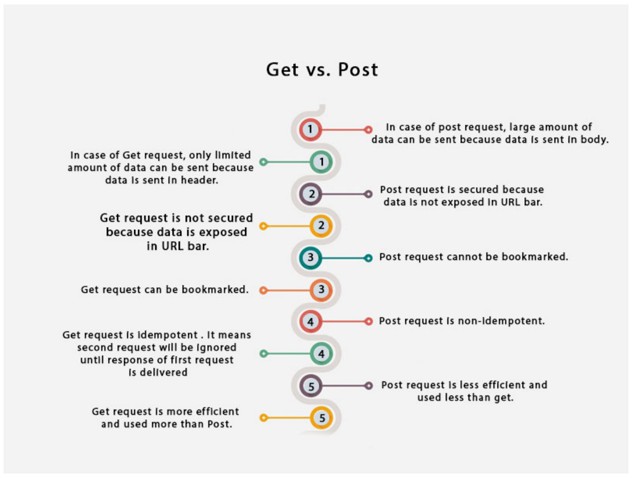
MySQL, Oracle, Apache Derby, PostgreSQL, Microsoft SQL Server, Microsoft Access, Amazon Redshift, Vertica, IBM DB2, SQLite, MimerSQL, Firebird, Sybase and SAP MaxDB is only a small sample of all database servers that are supported by SQLMap. It can connect to these databases directly with necessary information like IP address, database name, port* and DBMS* credentials. Many more features are provided by SQLMap.

4.11 Wfuzz

Wfuzz is a bruteforcer that performs security assessments onto web applications. It is free, open source and available on GitHub. Web fuzzer does not have GUI interface so command line would have to be used.

Brute force in cyber attacks is a method of guessing passwords, login credentials, much more of sensitive information, gaining access to unauthorised data through trial and error. Brute forcing has no strategy behind it and the attack is primarily executed via GET and POST requests for measuring security of various vulnerabilities like SQL, XSS, LDAP etc. Many more other features are supported by Wfuzz.

Figure below is a light description of what GET and POST requests are and their differences.


Get vs. Post

1 In case of post request, large amount of data can be sent because data is sent in body.

1 In case of Get request, only limited amount of data can be sent because data is sent in header.

2 Post request is secured because data is not exposed in URL bar.

2 Get request is not secured because data is exposed in URL bar.

3 Post request cannot be bookmarked.

3 Get request can be bookmarked.

4 Post request is non-idempotent.

4 Get request is idempotent . It means second request will be ignored until response of first request is delivered

5 Post request is less efficient and used less than get.

5 Get request is more efficient and used more than Post.

Figure 2: GET vs POST requests (source:)

4.12 Nessus

Nessus is a web scanner developed by Tenable that offers paid and free versions, however the paid version is from 2 to 8 thousands dollars worth and the free version only allows scans up to 16 IP addresses per scanner.

This scanner can run on Windows, Mac, Amazon Linux, Ubuntu, Fedora, RHEL, FreeBSD Unix, Debian and SUSE. Nessus states to provides the lowest false positive rate in the industry and broadest vulnerability coverage and but is reviewed to be pretty slow in comparison to the competitors.

The paid version web scanner continuously scans for known vulnerabilities, as well as for signs of unusual behaviour. It offers customizable reporting, risk-based vulnerability prioritization, machine learning automation, complete visibility of the network infrastructure. Nessus is an intuitive application, made customers easy to learn UI.

4.13 Burp suit

Burp Suite offers community version, professional and enterprise levels for security testing. Community version is free compared to the other two however, it is extremely limited to the features it provides including the plugins it provides. For example, only enterprise version offers automated dynamic scanning and only manual control is proposed for web vulnerability scanning.

Community version provides essential tools such as, Repeater, Decoder, Sequencer, and Comparer and HTTP(s) / WebSockets proxy and history. Professional version will be able to provide an addition of Auto and manual OAST testing*, Automatically crawl and discover content to test, ability to save files. Pricing starting from under $500 also has strong user community which would aid customers if needed.

4.14 Nmap

Nmap is a free and open source tool that is over 25 years old, started its journey as a Linux utility and has thus stayed most popular between Linux users to this day. Mac OS x, HP-Ux, Amiga Microsoft Windows and more are now supported by Nmap.

It is a network discovery and security auditing tool, also used for things as monitoring host or service uptime, scanning networks for ports, devices, service. Nmap uses raw IP packets to establish the available hosts on the network, the services, operating systems, firewalls are of the hosts and more. Huge networks could be scanned at a high speed but also works with single hosts. Most importantly Nmap offers a vulnerability scanning characteristic, even though it isn't the primary use of this scanner.

Although Nmap is has won several awards and has been a feature in magazines, books or movies, it is reviewed to lack quality in UI and user support system.

4.15 Final chosen Scanners

Few characteristics had an impact on which scanners were chosen. One of them being the pricing, ranging from free to the mid range of 4 figures and thus the free and/or open source scanners were chosen.

It was at most importance to make sure that the chosen scanners used a GET method to find the vulnerabilities, instead of the POST method. As previously shown the difference between the two, the former chooses to only gather information on the source code, where as the latter injects the codes into the body of the website. The POST method can be dangerous, as it injects malicious codes that could disrupt the targeted website. Thought such method is highly useful for detecting vulnerabilities such as SQL injection, which focuses on databases, it requires the permission of the websites owner to proceed with such method in order for it to be ethical and legal. Conclusions were made to focus on GET method scans only.

Another factor that has impacted the choice of the scanner was the vulnerabilities that the free editions of the expensive scanners or the open source tool have checked. The scanners that were chosen are Zap, N-Stalker and VEGA. Zap scanner being an open source tool while N-Stalker, the Free Edition X was chosen.

Zap scanner had many options for automated scanning, one of which being 'spider' option. It scanned many vulnerabilities that were classified into OWASP Top 10 groups in the GUI. The 'spider' scans also only used GET method requests into the websites.

N-Stalker offers a few policies to scan, 'OWASP Policy' being one of them, among manual tests. As it was noted before, only 500 pages per URL can be scanned, but it is enough to make conclusions with false positives being taken care of by the scanner. The OWASP feature also portrays its vulnerabilities with classifying into OWASP groups. The Zap scanner and N-Stalker both having in common the OWASP feature, the comparison between them could be made.

5 Collecting Data

This research paper has chosen 100 Lithuanian websites to test their security issues. The websites are based on educational institutions due to them being vital in modern education system. They provide a platform for students, teachers, and administrators to access and share information, resources, and tools. It is crucial to conduct vulnerability testing on educational sector websites as addressing the security weaknesses will ensure the users sensitive information, any online assessments and exams stay protected. Most importantly, Lithuanian websites, especially of educational institutions have not yet been researched in vulnerability testing.

The 100 websites were to be scanned with 2 scanners, thus Virtual Machine has been incorporated into the practical work along side of couple of devices for this study. The virtual machine that was chosen was VirtualBox, and up to three virtual machines were created with CPUs of 2 each on a Ubundu operative system. The devices with Microsoft and Mac OS operative systems were also used.

5.1 ZAP

ZAP was run on up to three virtual machines on top of 2 devices, for a quicker automatic scan completion. Once the URL of the website was inputted into automatic scan section, the spider scan option was checked and thus scanned. The following vulnerabilities were found through out all of the 100 scans with respective vulnerability classifications of OWASP Top 10:

A01:2021 Broken Access Control

- Absence of Anti-CSRF Tokens
- Cookie without SameSite Attribute

A01:2021 Broken Access Control
A03:2017 Sensitive Data Exposure

- Information Disclosure - Debug Error Messages
- Private IP Disclosure
- Server Leaks Information via ”X-Powered-By” HTTP Response Header Field(s)
- Timestamp Disclosure - Unix

A02:2021 Cryptographic Failures
A05:2021 Security Misconfiguration

- HTTP to HTTPS Insecure Transition in Form Post

A03:2021 Injection

- SQL Injection - SQLite
- User Controllable HTML Element Attribute (Potential XSS)

A04:2021 Insecure Design
A03:2017 Sensitive Data Exposure

- Hash Disclosure - Mac OSX salted SHA-1
- PII Disclosure
- Big Redirect Detected (Potential Sensitive Information Leak)

A05:2021 Security Misconfiguration

- CSP: Wildcard Directive
- CSP: script-src unsafe-inline
- CSP: style-src unsafe-inline
- Content Security Policy (CSP) Header Not Set
- Hidden File Found
- Missing Anti-clickjacking Header
- Cookie No HttpOnly Flag
- Cookie Without Secure Flag
- Secure Pages Include Mixed Content
- Server Leaks Information via ”Server” HTTP Response Header Field(s)
- Strict-Transport-Security Header Not Set

- Strict-Transport-Security Multiple Header Entries (Non-compliant with Spec)
- X-Content-Type-Options Header Missing

A06:2021 Vulnerable and Outdated Components
A09:2017 Using Components with Known Vulnerabilities

- Vulnerable JS Library

A08:2021 Software and Data Integrity Failures

- Cross-Domain JavaScript Source File Inclusion

The years represent which year the OWASP Top 10 has been formed. The ZAP application has also provided characteristics for each scan as, the total number of URLs found, the quantity of nodes used, as well as the quantity of high, medium and low risks are across the whole web application. Most importantly list of vulnerabilities. Each vulnerability has portrayed its own level of risk, its quantities within the web application and specific URLs that were vulnerable to it, respectively to the method that was used for the request individually. A short descriptions and solutions have also been formed.

An example of a scan is shown below:

The Request code that was automatically inputted for a specific website path:

```
GET http://www.***/***/***
Host: www.***.**
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:105.0) Gecko/20100101
    Firefox/105.0
Pragma: no-cache
Cache-Control: no-cache
Referer: http://www.aviliukas.com
Cookie: is_mobile=0; language=en
```

As each website has reported a list of information mentioned previously, it has been collected into a concise excel data set.

5.2 N-Stalker

Upon downloading N-Stalkers, the front page had a user input a base URL and choose a policy to scan, which for this research, OWASP policy was chosen. The optimization of the scan was also a chosen option that enabled false positive for multiple extensions and enabled them for 404 pages; 30 Urls were used per nodes and only one host to be configured, none additionally added. Since the N-Stalker is a paid tool, however the free version X was used, there were limitations to it with 500 Urls to be Spider scanned per each website base. Nonetheless, it was enough to draw conclusions for this research. Lastly, other 13 modules were run. The Scan Modules that were run:

- N-Stalker Spider Module
- Clickjacking Inspection Module

- Cookie Inspection Module
- Flash/Silverlight Cross-domain inspection Module
- Cross-Site request forgery Assessment
- WebDAV Detection Module
- HTTP Method Finder
- File Extensions Finder
- 3rd-Party Package Scanner
- HTTP Parameter Pollution Assessment
- Information Leakage Assessment
- Cross-Site Scripting Assessment
- WebServer Infrastucture Assessment
- Sensitive File Search Assessment

Following Vulnerability classes were detected and reported:

OWASP (None) — CWE (None)

- Web Server Infrastructure attack
- HTTP TRACE
- PHP Info Found
- Packages names'

OWASP (Top 10 A1) - Broken Access Control — CWE (235)

- HPP

OWASP (Top 10 A3) - Injection— CWE (79)

- Cross-Site Scripting

OWASP (Top 10 A5) - Security Misconfiguration

- Wordpress — CWE (22)
- Web Server Infrastructure attack — CWE (310)
- httponly cookie — CWE (614)
- non-secure cookie — CWE (614)
- ClickJack — CWE (693)

OWASP (Top 10 A6) - Vulnerable and Outdated Components

- Information Leakage — CWE (209)
- Source Code Exposure — CWE (540)

OWASP (Top 10 A8) - Software and Data Integrity Failures— CWE (352)

- CSRF

The GUI of N-Stalker was somewhat similar to ZAP. It produced information on which vulnerability belongs to which OWASP group, the vulnerability class and limited information on each of the vulnerabilities due to its edition. The severity level, high, medium or low was also portrayed for each of the cases. The data was collected into an excel spreadsheet for the further analysis of this research.

5.3 Collected Data Conclusions

With the same 100 websites, N-Stalker and ZAP scanned a total of 1,140,521 Urls. According to the table below, N-Stalker detected the highest count of medium level vulnerabilities and ZAP, the low level.

	Number of Websites	Total Url Count	Level of Vulnerability		
			High	Medium	Low
N-Stalker	100	38038	1049	1899	276
ZAP	100	1102483	120	424	623
Total	100	1140521	1169	2323	899

6 Methodology

For this research paper, the comparison between the scanners was chosen to be done to evaluate how well or similarly do scanners detect websites vulnerabilities. As well as there is no real data reported on Lithuanian websites vulnerabilities, nor the data on scanners performance on real websites, to compare to. The collected data has no ground truth, therefore unsupervised learning methods should be considered. Common trends and features, if found, can inform further analysis or research.

One way to compare the data sets collected is by using clustering. Clustering algorithm is a method that aims to group together the data points that are similar to one another and those that are dissimilar, to be separated. With the help of clustering, the clusters of the data could be visualised and also compared, analysing the data points within each cluster.

Before the subsection of clustering, the data sets should be preprocessed, transformed through dimensionality reduction for better visualization. After all of which, different clustering algorithms would be applied to later also be analysed and evaluated. Finally, the cluster intersection between the two data sets will be examined to find whether the data points, websites, fall into the same clusters from different data sets.

6.1 Scaling

Scaling is a significant part when working with data sets, to make sure that all features are of the same scale, thus brings forth an analysis of similar importance. When features are of difference range, scaling the data makes an approach to comparison much easier, and furthermore to perform other types of analysis like clustering. Clustering algorithms are sensitive to scale of the data, such as k-means [28], if the features are not on the same scale, then some features may dominate the distance measure, making it harder to compare observations based on the other features.

Scaling data causes the features to be equally weighted in the distance measure, making the comparison between the datas based on all the features and easier procedure, consequently improving the result interpretability.

6.1.1 Min-Max Normalization

Min-max normalization helps to address an issue in data sets with different range and scale by scaling the features in a certain range of [0,1] [28].

This method of scaling performs a linear transformation on the original data.[3].

$$X_{new} = \frac{X - min(X)}{max(X) - min(X)}$$

X_{new}=The new value after normalization has been applied
X=Old value
$Max(X)$=Maximum value in the data set within the feature, in this research it being a vulnerability.
$Min(X)$=Minimum value in the data set [11]

Research papers [11], [25], [3], though with the use in different algorithms, have supported the idea of min-max normalization being the best fit for scaling data for the best results. This research too will focus on normalizing the two data sets with min-max normalization.

6.2 Dimensionality Reduction

The collected data set from the performed scans are high dimensional datasets, however to visualize the data would be very difficult, yet possible by reducing it to lower dimensional space with the help of dimensionality reduction [14]. Visualizing data in two or three dimensions, would be much easier to see patterns, interpret the data and identify clusters in the data. Dimensionality reduction produces latent space representation preserving properties and information of the original data [14].

6.2.1 t-Distributed Stochastic Neighbor Embedding (t-SNE)

t-SNE is nonlinear dimensionality reduction technique that has an algorithm of the application of SNE (Stochastic Neighbor Embedding) to the data points, that passes a method of conversion between high-dimensional Euclidean distances between data points into conditional probabilities constituting similarities. So when the data points are mapped ontp a lower dimensional space, the relationship between the points is preserved as probabilities. t-distribution with a single degree of freedom is applied to avoid overcrowding. t-SNE also retains structure of the global and local data[18] with larger perplexity emphasising global structure and smaller perplexity defining local structure. t-SNE appears to be sensitive to its parameters, “perplexity” evidently being one of them. Tt is a representation of the smooth version of the number of the nearest neighbours in the t-SNE algorithm. The research paper [14] has stated with references ”the best visualization technique for the past decade has been t-SNE”, thereafter it will be used for the collected data.

For both Zap and N-Stalker datasets, the t-SNE parameters were the same. The $n\_components = 2$, representing 2 dimensions. *Perplexity* equating to 40 because the clusters printed in scatter plots were most defined and separate using this value.

6.2.2 Multidimensional Scaling (MDS)

Multidimensional scaling is used on the dataset due to it being a linear dimensionality reduction technique, potentially producing a different set of results from t-SNE, and accordingly could be then compared.

MDS is a method where relationships between the data points, the distances can be preserved in the low-dimensional space, using dissimilarity matrix[15], [19]. The spatial structure of the data points is found using the dissimilarity information[8]. Due to MDS representing the distances of interpoint, it makes it useful for identifying and visualising clusters.[15].

Two of the data sets were reduced to 2 dimensions using MDS, with $n_components$ value being 2. *eps*=1e-3 because *eps* is a parameter that determines the minimum distance between the points in the low-dimensional space and smaller eps value will result in points that are closer together. $metric = True$ is a metric MDS and $metric = False$ is a non-metric MDS. The later is distanced-based, as distances between the data points is meaningful and proportional. The former, is similarity-based, where data is treated as original data.

6.3 Clustering Methods

Three clustering algorithms were chosen: DBSCAN, K-Means and Gaussian Mixture due to their difference in use of technology.

The clusters were evaluated using Silhouette Score and Calinski-Harabasz index. The silhouette index is an unsupervised method that does not require any training and can be applied on evaluating clustering results [27]. It evaluates the effectiveness of clustering by measuring the difference between two values. One being the average distance within a cluster, and two being the minimum distance between clusters [29]. The silhouette index is in the range of [-1.0, 1.0], where 1.0 is the best, 0.0 implies overlapping and -1.0 is the worst value. These values intepret on how similar or dissimilar is the object from its own clusters. 1.0 representing similarity and -1.0, the opposite [18]. The higher the silhouette score is for the corresponding cluster count, the more optimal the cluster count is [26].

The Calinski-Harabasz index is a measure of the quality of a clustering. An evaluation index based on the degree of dispersion between clusters[29], in other words, based on the variance between clusters, with a high Calinski-Harabasz index suggesting that the clusters in the data are distinct and well-defined, which is a desirable characteristic of good clustering [1].

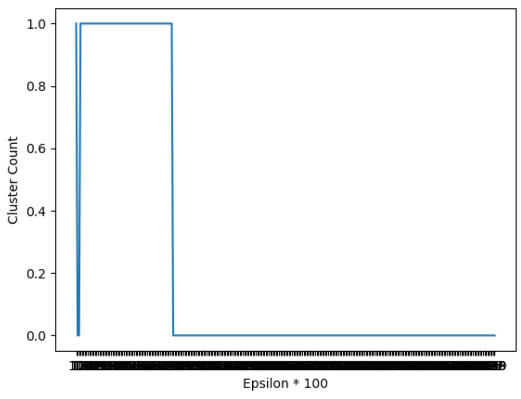
6.3.1 DBSCAN

Density-based spatial clustering of applications with noise, shortened to DBSCAN is an algorithm for clustering data. Min_sample and *eps* being the two parameters, representing the minimum cluster size and the maximum distance within the clusters data points respectively.

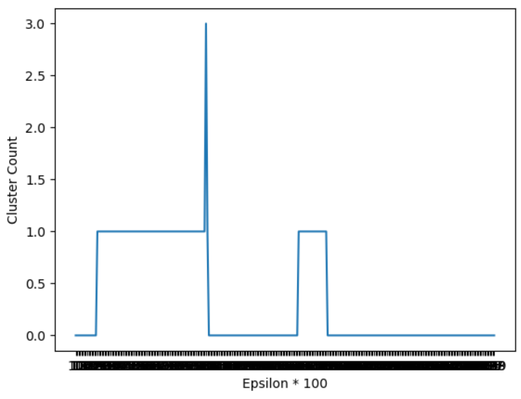
The DBSCAN algorithm works by identifying point in the dataset, that have many other points close to them, in other words neighbouring points within the maximum distance between them. It then expands the clusters from the main points by adding points that are close to the main points and satisfying the minimum sample size requirement and maximum distance. Points that do not belong to any cluster because there are considered "noise" and are not included in the clusters.

Clusters are identified using density of the data points in the feature space. Clusters being density-based causes them to be of different sizes and shapes. DBSCAN experiences smaller effect of outliers due to the separation of the 'noise' cluster if there is a too of an apparent difference between the data points and is not forced into a cluster. [6]

The *eps* was considered through creating a plot which portrayed how many cluster were created per $eps * 100$. The *eps* value that created the most clusters was taken into account. However, for ZAP scanner dataset, according to Figure (a) below, the *eps* value was in the range of multiple x-axis values, where they subsequently were inputted into the DBSCAN function, *eps* parameter, and evaluated by Silhouette scores and Calinski-Harabasz index to make conclusions on which from the range *eps* value is the best. The min_sample value was chosen to be 3, as the choice between the two values of 3 and 4 made no difference to the Silhouette scores and Calinski-Harabasz index. Taking into consideration the *eps* and min_sample values, the DBSCAN clustering was performed and visualized using a scatter plot. The scatter plot used the dimensionaly reduced data with labels of min-max scaled data.



(a) Cluster count per *eps*, ZAP

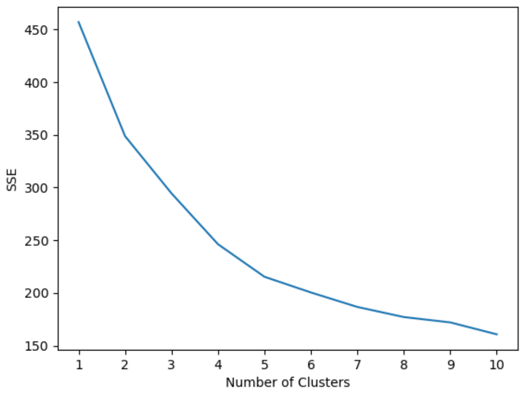


(b) Cluster count per *eps*, N-Stalker

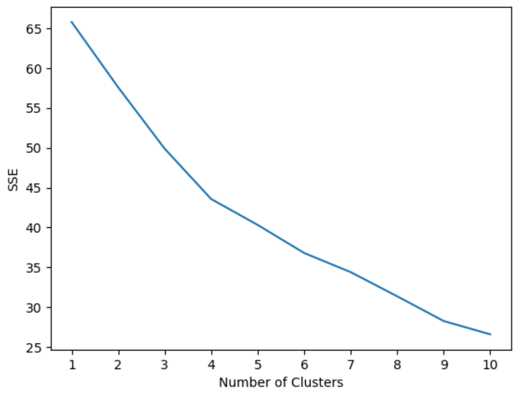
The Figures (a) and (b) show the *eps* values that produces most clusters of the ZAP dataset is [0.1,0.67] and for N-Stalker, 6 clusters were created at $eps = 1.0$.

6.3.2 K-means

K-Means is a centroid-based clustering, with a parameter $n_clusters$, representing a cluster count. K-Means selects n initial locations of a centroid randomly, or the first n objects in a sequence. Upon choosing the number of cluster, K-Means forces its data points into clusters making it sensitive to outliers. This clustering algorithm will produce inaccurate results if some features are of large size, therefore data would have to be preprocessed. The distance between the data point and the centroid is determined and thereafter the data points are assigned to the clusters corresponding to the nearest centroid. The centroids converge at which point the algorithm stops. [6] [20]



(a) K-Means of N-Stalker



(b) K-Means of ZAP

The count of clusters was taken into consideration through performing an elbow method.

The k value was chosen based on where is a 'bend' in the plot. For N-Stalker scanner, the k value was not precise and for ZAP scanner, k=4. The clusters using these k values were also evaluated using silhouette scores and calinksi-harabasz index and compared with results using other k values to validate the initial choice.

6.3.3 Gaussian Mixture

Gaussian Mixture is a probabilistic clustering method generating density based clusters. Clustering data points into clusters is done by fitting separate Gaussian distributions to each cluster. The model computes the probability of each data point of the dataset belonging to each cluster, based on the fitted Gaussian distributions. The data point is then assigned to the cluster with the highest probability. Gaussian Mixture model use mean and covariance as opposed to only means like in K-Means, to provide better results. [24]

Initially AIC and BIC methods were used to find out which cluster value should be used for Gaussian Mixture clustering, unfortunately the results were not interpretable for further use. Therefore, a range of cluster counts were inputted and evaluated using the same cluster evaluation methods.

6.4 Cluster Intersection

Finally, the clusters created within each datasets were due to be compared. It was checked how similarly did same data points from different data sets were found in the same cluster across all three clustering algorithms. For this, the percentage of rows that have different cluster predictions were calculated for all three clustering methods on top of a small range of different parameters.

7 Results

7.1 DBSCAN

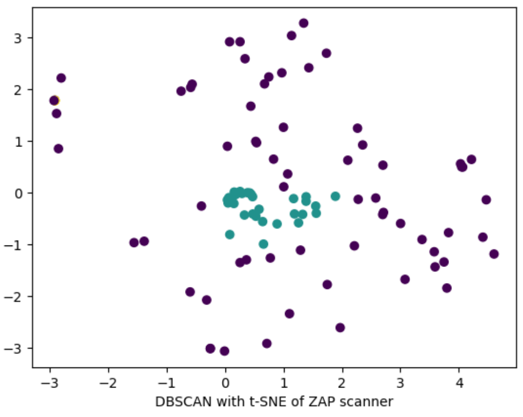
Different values of the parameter 'eps' were used to find the Silhouette Score and Calinski-Harabasz index. Prior to that. the outliers have been removed using the DBSCAN function, with a 'min_samples' value of 3. The score and the index were calculated using the resulted clean data (the data without outliers). The results are in the following table:

DBSCAN

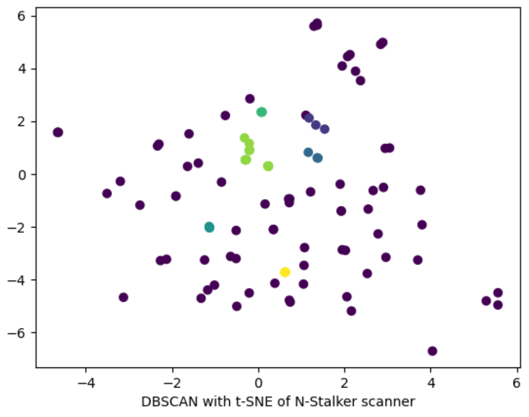
ZAP	eps	0.1	0.13	0.25	0.5	0.6	0.7
	Silhouette score	0.57	0.88	0.80	0.67	0.56	0.52
	Calinski-Harabasz index	26.73	378.31	165.27	70.55	35.31	29.27
N-Stalker	eps	0.13	0.5	0.7	0.9	1	1.6
	Silhouette score	0.97	0.78	0.72	0.52	0.39	0.36
	Calinski-Harabasz index	11086.29	152.81	64.45	64.45	18.16	13.83

The table above shows *eps* = 0.13 providing the highest Silhouette scores and the Calinski-Harabasz index for the dataset using ZAP scanner, and same for N-Stalker scanner data set. Previously it was noted, that range of *eps* that has most clusters was [0.1, 0.76] and 0.13 falls into this range. However, the N-Stalker *eps* value does not. The value for *eps* = 1.0 shows quiet poor results.

Using DBSCAN, the clusters were visualized using scatter plots using both dimensionality reductions, t-SNE and MDS. Plots used *eps* = 0.13 for both ZAP and N-Stalker databases, due to the table produced previously. The parameters for t-SNE were *perplexity* = 40 because it gave the most distinct clusters and *n_components* = 2 as the focus was on 2 dimensions. *Learning_rate* and *n_iter* were 70 and 360 respectively as it was the upper bound of what resulted in better clusters, known through experimenting with different values.

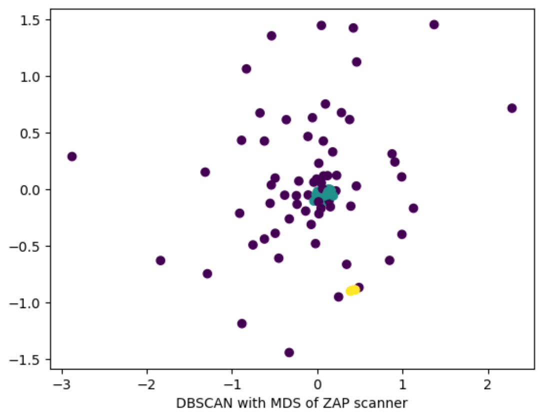


(a) t-SNE, perplexity=40 for ZAP scanner

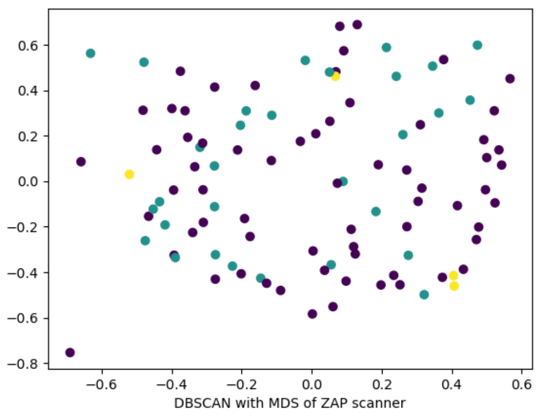


(b) t-SNE, perplexity=40 for N-Stalker scanner

Figure 5: DBSCAN

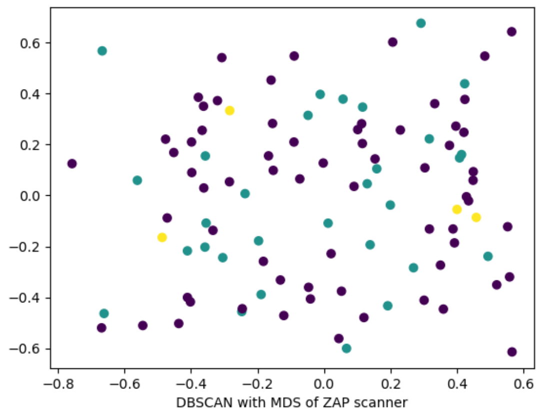


(a) MDS, metric=True, ZAP

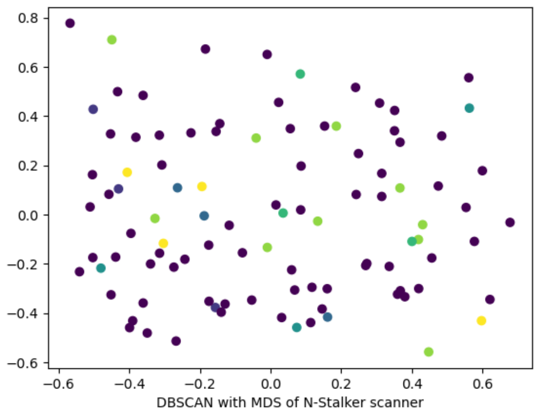


(b) MDS, metric=False, ZAP

Figure 6: DBSCAN

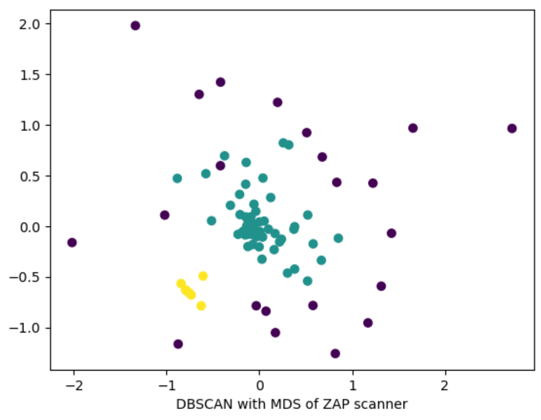


(a) MDS, metric=True, N-Stalker

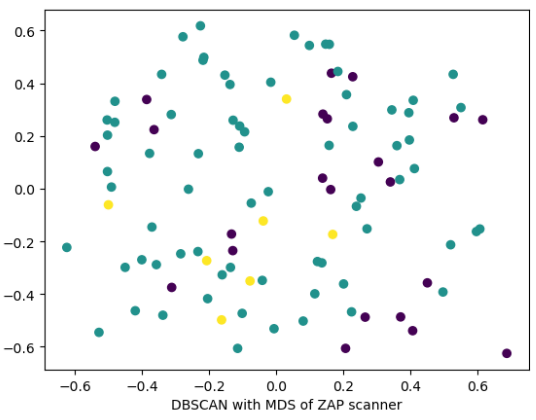


(b) MDS, metric=False, N-Stalker

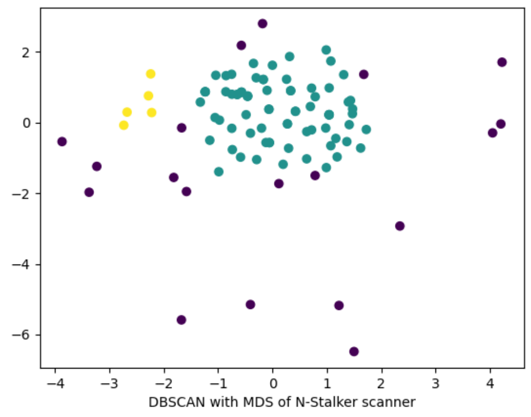
Figure 7: DBSCAN



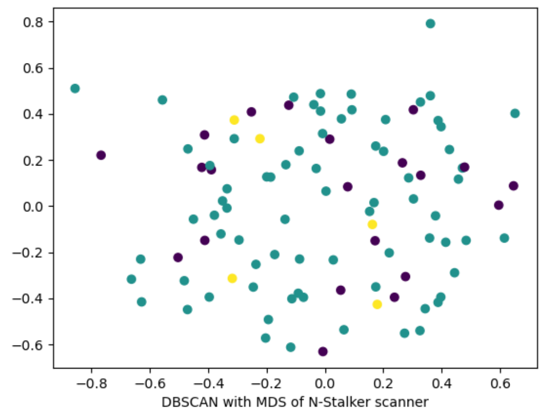
(a) eps=0.6, metric=True, ZAP



(b) eps=0.6, metric=False, ZAP



(c) eps=1.6, metric=True, N-Stalker



(d) eps=1.6, metric=False, N-Stalker

Figure 8: DBSCAN with new parameters (MDS)

For visualisation using MDS dimensionality reduction, both metrics *'True'* and *'False'* were used, purely to visualise and compare. The scatter plots for both metrics and both datasets were very inadequate. So, to experiment, few other values for *eps* were used, where *eps* = 0.6 and *eps* = 1.6 shows most distinct clusters for ZAP and N-Stalker datasets respectively, using *'True'* metric. Changing values of other parameters for MDS function, like *eps* or *n_init* did not create noticeable changes to the plots.

The DBSCAN *eps* value that showed best results in Figure 8(a),(c), contradicted the Silhouette score and Calinski-Haarabasz index, in terms of which eps value provides the best results. However, same could not be said about the use of t-SNE dimensionality reduction method. While experimenting with different *eps* values, the scatter plots with *eps* = 0.13 had the most distinct cluster. It is important to take note that the scores and index's were calculated without the outliers, but the plots were portrayed with original data. It could be implied that Figure 5(a), the darker data points are the said outliers, thus removing them will create better scores overall.

7.2 K-Means

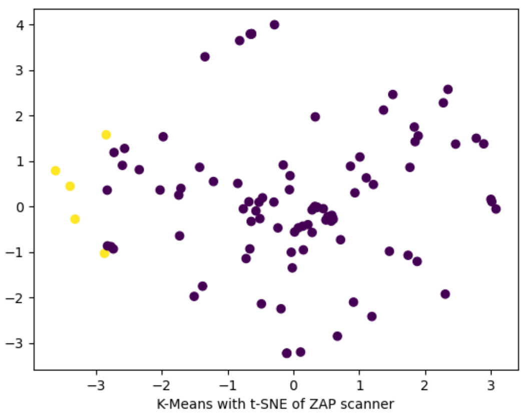
K-Means cluster evaluation was done by using 2, 3, 4 and 5 clusters. Previously elbow method showed k=4 for ZAP dataset and not a precise cluster for N-Stalker dataset, therefore couple of cluster counts were applied.

K-Means

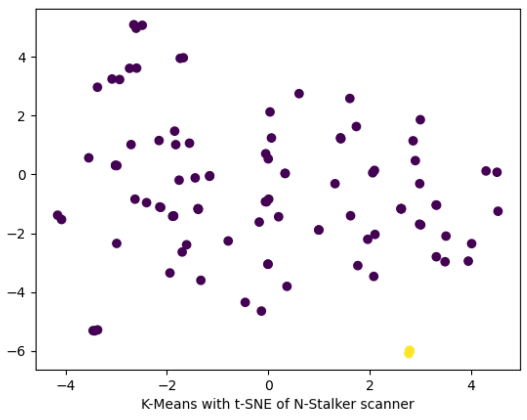
	n_clusters	2	3	4	5
ZAP	Silhouette Score	0.627	0.398	0.421	0.367
	Calinski-Harabasz index	14.794	16.268	17.349	17.147
N-Stalker	Silhouette Score	0.596	0.344	0.379	0.195
	Calinski-Harabasz index	30.398	26.759	27.379	26.644

According to the table above, it is pretty clear how 2 clusters were shown to have best Silhouette scores. However, the Calinksi-Harabasz index does not support the score for ZAP scanner dataset, but supports the elbow method plot, with best number of clusters being of 4.

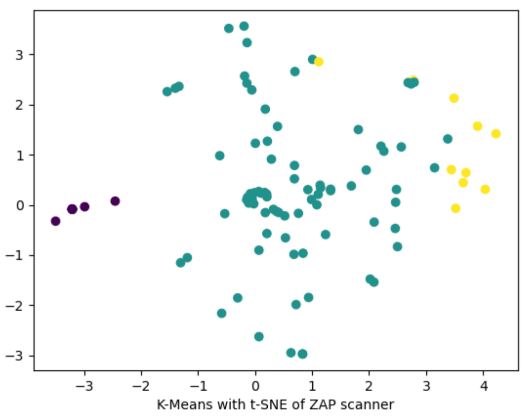
Afterwards, the scatter plots for both datasets with 2, 3 and 4 clusters were plotted, using both t-SNE and MDS Dimensionality Reductions.



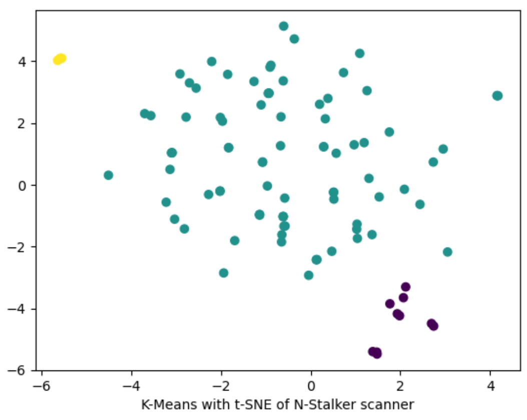
(a) k=2, ZAP



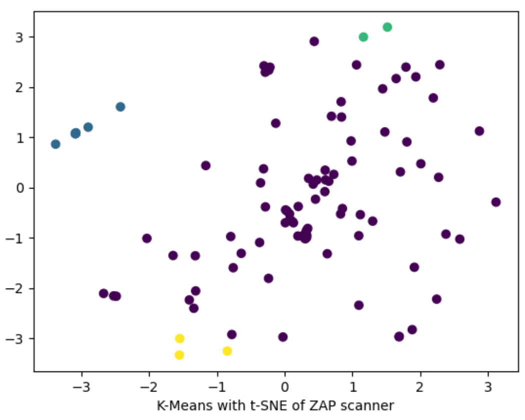
(b) k=2, N-Stalker



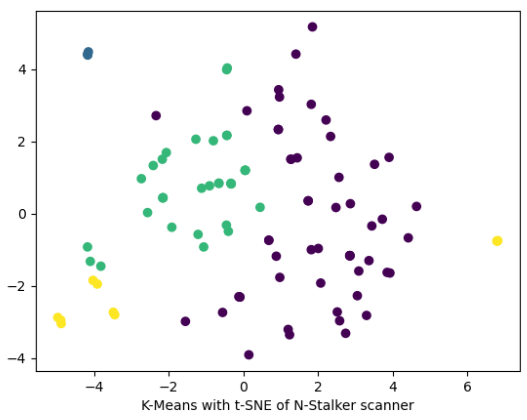
(c) k=3, ZAP



(d) k=3, N-Stalker

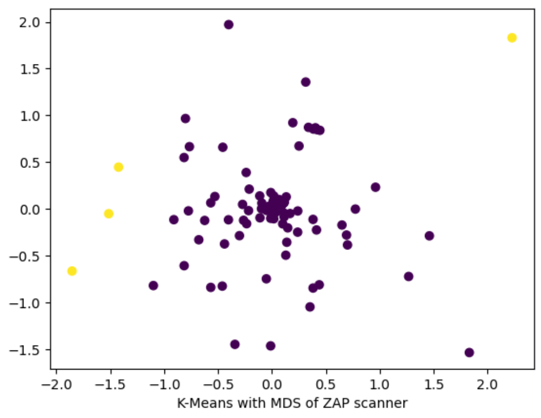


(e) k=4, ZAP

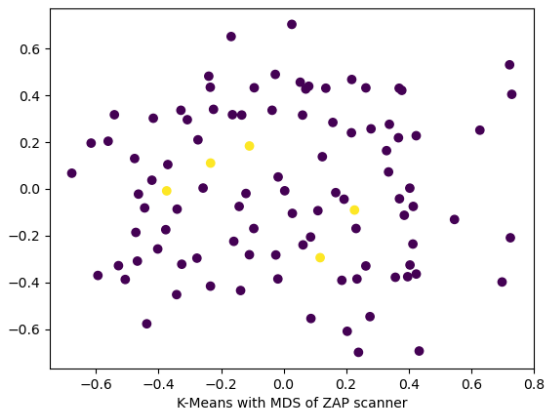


(f) k=4, N-Stalker

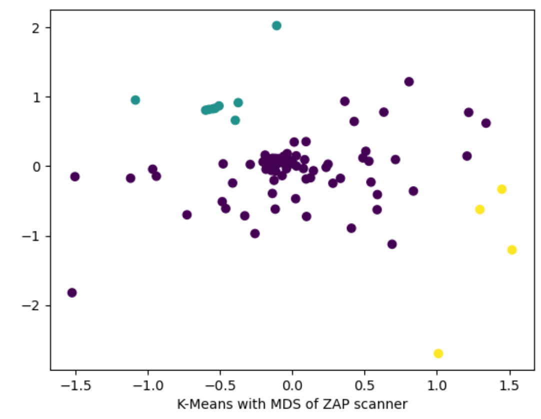
Figure 9: K-Means (t-SNE, perplexity=40)



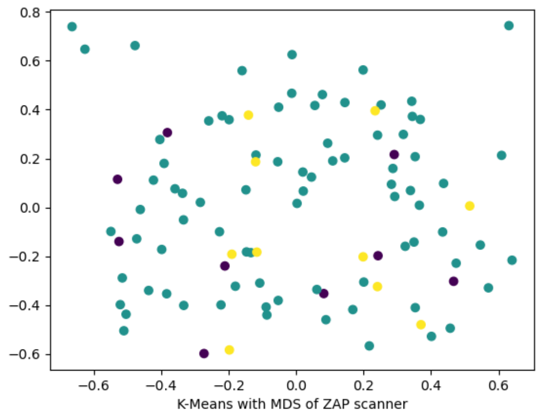
(a) k=2, $metric =' True'$



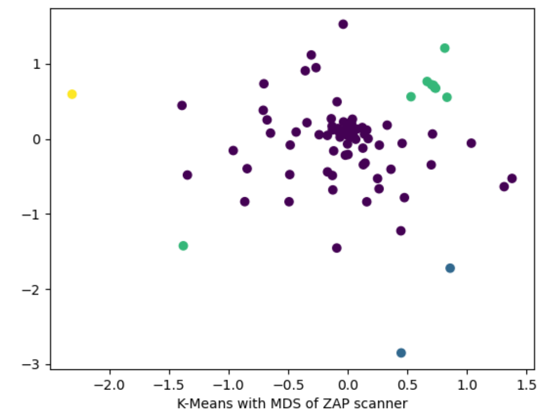
(b) k=2, $metric =' False'$



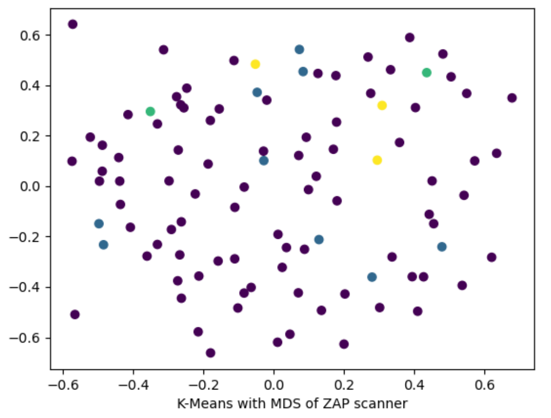
(c) k=3, $metric =' True'$



(d) k=3, $metric =' False'$

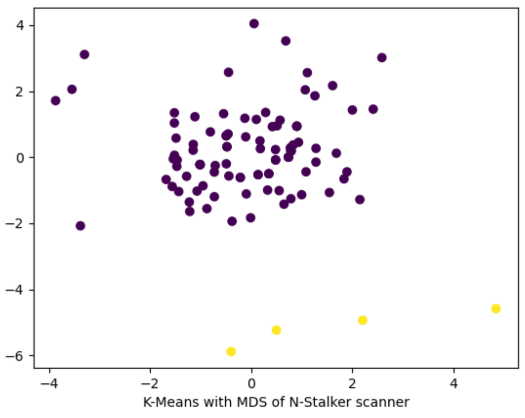


(e) k=4, $metric =' True'$

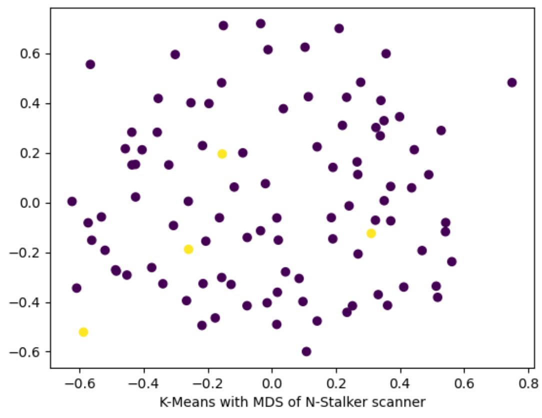


(f) k=4, $metric =' False'$

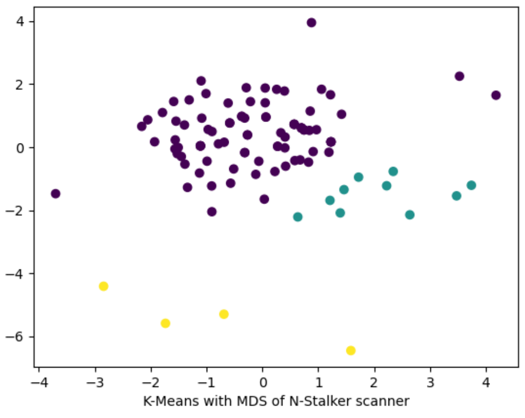
Figure 10: K-Means (MDS) for ZAP



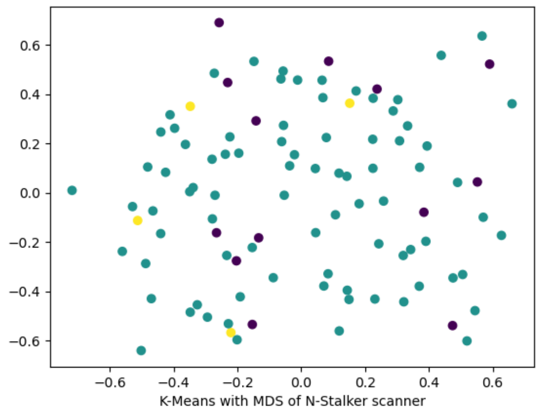
(a) k=2, $metric =' True'$



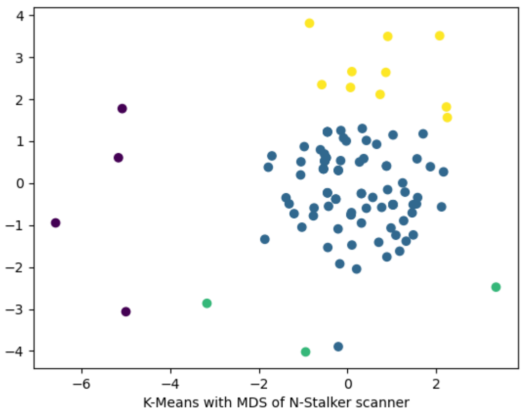
(b) k=2, $metric =' False'$



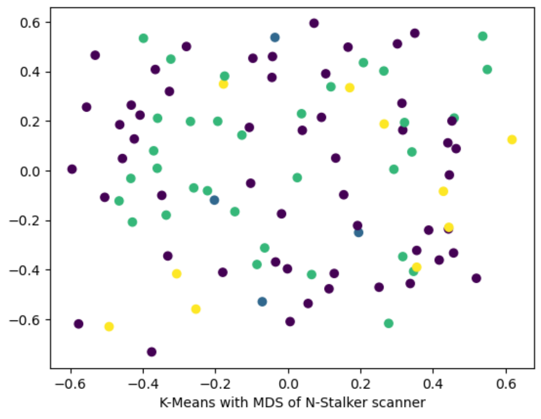
(c) k=3, $metric =' True'$



(d) k=3, $metric =' False'$



(e) k=4, $metric =' True'$



(f) k=4, $metric =' False'$

Figure 11: K-Means (MDS) for N-Stalker

The parameters for t-SNE and MDS were kept the same as for DBSCAN clustering for the same reasons as well. For clustering using K-Means algorithm, according to Figure 9, ZAP scanner provides best scatter plots with 3 clusters (Figure 9(c)), likewise for N-Stalker too (Figure 9(d)). For the use of MDS dimensionality reduction, ZAP portrays most clear clusters using $metric =' True'$, yet none of the cluster count provides satisfactory plots, but for dataset of N-Stalker, Figure 11(c) have better clusters of count 3.

7.3 Gaussian Mixture

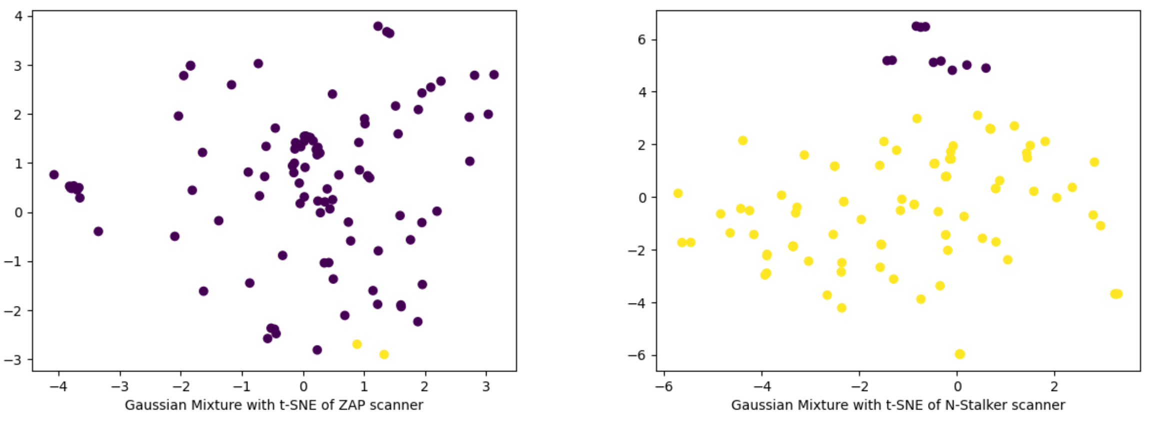
Gaussian Mixture algorithm used the parameter $'n\_components'$, and the range of [2,8] was chosen, resulting in such table for evaluating the cluster count:

Gaussian Mixture

	n_components	2	3	4	5	6	7	8
ZAP	Silhouette Score	0.677	0.531	0.375	0.407	0.331	0.377	0.361
	Calinski-Harabasz index	13.5	13.91	11.92	11.8	10.69	14.44	13.03
N-Stalker	Silhouette Score	0.424	0.147	0.168	0.179	0.155	0.157	0.147
	Calinski-Harabasz index	12.14	23.22	21.88	25.26	22.8	19.59	18.84

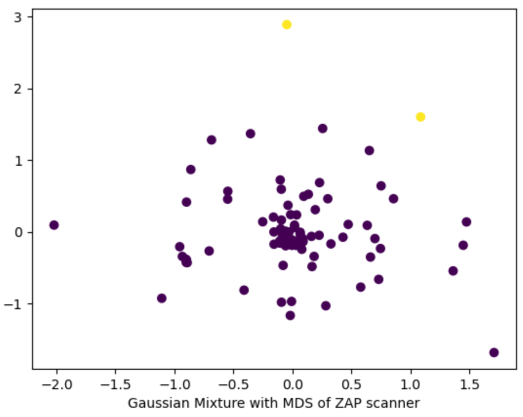
This table is to show, 2 cluster provided best Silhouette scores, yet Calinski-Harabasz index did not support and showed $gm = 7$ and $gm = 5$ were better for ZAP and N-Stalker respectively.

Consequently, Gaussian Mixture went through the same procedure as K-Means and DBSCAN algorithms to be visualized in terms of the dimensionality reduction t-SNE and MDS choice of parameters. The $gm = 2$ is used to for visualisation.

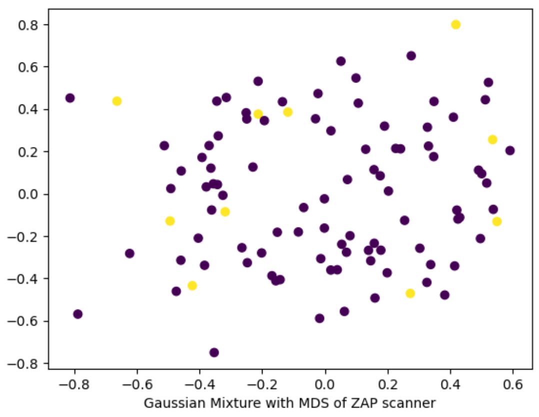


(a) gm=2, ZAP (b) gm=2, N-Stalker

Figure 12: Gaussian Mixture (t-SNE, $perplexity = 40$)

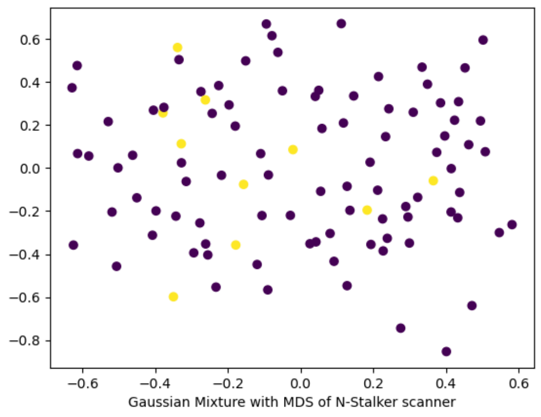


(a) gm=2, $metric =' True'$

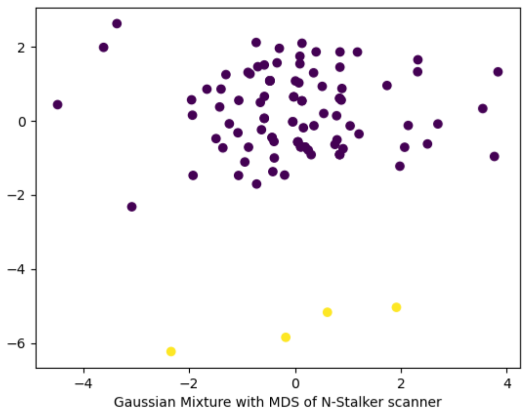


(b) gm=2, $metric =' False'$

Figure 13: Gaussian Mixture (MDS) for ZAP



(a) gm=2, $metric =' True'$



(b) gm=2, $metric =' False'$

Figure 14: Gaussian Mixture (MDS) for N-Stalker

The Gaussian Mixture portrays similar clusters for ZAP and N-Stalker dataset using 2 clusters - Figure 12 with t-SNE dimensionality reduction. However using MDS, Figure 13(a) and Figure 14(b), clusters are more clear.

7.4 Cluster Intersection

The cluster intersection was calculated of DBSCAN, K-Means and Gaussian Mixture clustering algorithms, using the parameters mentioned in the table. The parameters for t-SNE and MDS were kept exactly same as for all the other calculations.

The table shows a range of results, from 8% to 44% difference of cluster predictions of rows. It should be noted that DBSCAN is the only algorithm to keep its results constant through out many initiations, which is due to the nature of DBSCAN, compared to K-Means and Gaussian Mixture as mentioned in previous section. Unfortunately, throughout

	The percentage of rows to have different cluster prediction		
DBSCAN	eps=0.13 & eps=0.9	eps=0.6 & eps=1.6	
	44.00%	39.00%	
K_Means	n_clusters=2	n_clusters=3	n_clusters=4
	18.00%	26.00%	33.00%
Gaussian Mixture	n_components=2	n_components=3	n_components=4
	8.00%	32.00%	40.00%

many initializations the percentage difference for K-Means and Gaussian Mixture clustering differed from the ones noted in the table even up to the 90%. However with the right initialization the results are to show the cluster prediction for the rows of the two datasets are fairly similar.

7.5 Result Conclusions

In the table down below, the best results, in terms of Silhouette score, for DBSCAN, K-Means and Gaussian Mixture clustering were chosen to their corresponding parameters.

		DBSCAN	K-Means	Gaussian Mixture
ZAP	Silhouette Score	0.88	0.627	0.677
	Calinski-Harabasz Index	378.31	14.794	13.5
N-Stalker	Silhouette Score	0.97	0.596	0.424
	Calinski-Harabasz Index	22086.29	30.398	12.14

It is portrayed that DBSCAN produces significantly best Silhouette scores and Calinski-Harabasz Index for both ZAP and N-Stalker scanners. The reason for this could be the effect outliers have on both K-Means and Gaussian Mixture clustering methods.

8 General Conclusions

100 websites of educational institution were tested for vulnerabilities using automated software tools. Zed Attack Proxy(ZAP), a free open source tool created by OWASP, and N-Stalker X Free Edition with over 20,000 databases, were used for this research.

1. After thorough investigation of many research papers and further study of many more software tools, the scanners that were most suitable for the research object, were: Zed Attack Proxy(ZAP), a free open source tool created by OWASP, and N-Stalker. that is a paid tool, however the X Free Edition was chosen.

2. The 100 websites were chosen within educational institutions, varying from universities, colleges, schools and nurseries. For the reasons of, the educational sector being a vital part of every persons life, thus its security is at its most importance. 200 scans and plus were performed, each taking from 30mins to multiple of hours. 1,140,521 of total Urls were scanned, identifying three levels of severity, high, medium and low, with

there being a total of 1169, 2323 and 899 total vulnerabilities detected respectively. Within N-Stalker 104 vulnerabilities were detected, and 34 with ZAP scanner.

3. The analysis was performed through clustering methods: DBSCAN, K-Means and Gaussian Mixture and comparison was done between these methods. DBSCAN performed the best representing highest Silhouette and Calinski-Harabasz indexes for both ZAP and N-Stalker scanners. ZAP and N-Stalker scanner encountered silhouette scores of 0.88, 0.97 and 378.31 and 22086.29 Calinski-Harabasz index respectively. These results were notably higher than of other clustering algorithms.

   It was determined that, from 8% to 44% of the websites fell into different clusters using ZAP and N-Stalker scanners. 8% belonging to Gaussian Mixture with 2 clusters. Through multiple internalisation's, it was found Gaussian Mixture was able to cluster the websites into the near same clusters across both datasets. That is to say that same websites were found to have the similar type of the vulnerabilities within ZAP and N-Stalker datasets.

Cluster intersections show that we can assume that two type of scanners detect same type of vulnerabilities.

Arriving at an interpretation that scanner performance is satisfactory, yet not exceptional and could be better with smaller percentage in differences of cluster intersections. Few attributes that could of effected this and may of been different is the nature of the scanners, one being, the path they chose for each Url base. Moreover, the software tools to be free, could imply its lesser quality, as less was invested into them than those that are paid. The control over false positives was also not taken by ZAP scanner, whereas it was by N-Stalker, leading to less accurate results.

Ultimately, many vulnerabilities were found across the websites within the Lithuanian educational institutions, which poorly reflects on the seriousness taken of the security of the websites that are browsed through. The automated scans is one of the options that a creator of the website could use to track the vulnerabilities present and take necessary actions to make browsing within the website a safer experience for the intended users.

9 Future Work

For future work, more scanners and more websites would also improve the credibility of conclusions working with automated testing. Nevertheless, penetrative testing could be considered. Manually looking through the source code for the vulnerabilities would eliminate any false positives that automated scanners don't take into account.

10 Acknowledgments

I would like to thank my supervisor Professor Julius Venskus for assisting me throughout the whole of research and appreciate the constant expert advice and support that was provided.

References

[1] Fachrizal Aksan, Michał Jasiński, Tomasz Sikorski, Dominika Kaczorowska, Jacek Rezmer, Vishnu Suresh, Zbigniew Leonowicz, Paweł Kostyła, Jarosław Szymańda, and Przemysław Janik. Clustering methods for power quality measurements in virtual power plant. *Energies*, 14(18), 2021.

[2] Mohammed S. Al-Sanea and Ahmad A. Al-Daraiseh. Security evaluation of saudi arabia's websites using open source tools. In *2015 First International Conference on Anti-Cybercrime (ICACC)*, pages 1–5, 2015.

[3] Luai Al Shalabi, Zyad Shaaban, and Basel Kasasbeh. Data mining: A preprocessing engine. *Journal of Computer Science*, 2(9):735–739, 2006.

[4] Jason Bau, Elie Bursztein, Divij Gupta, and John Mitchell. State of the art: Automated black-box web application vulnerability testing. In *2010 IEEE Symposium on Security and Privacy*, pages 332–345, 2010.

[5] Stefano Calzavara, Mauro Conti, Riccardo Focardi, Alvise Rabitti, and Gabriele Tolomei. Machine learning for web vulnerability detection: the case of cross-site request forgery. *IEEE Security & Privacy*, 18(3):8–16, 2020.

[6] Joshua M. Dudik, Atsuko Kurosu, James L. Coyle, and Ervin Sejdić. A comparative analysis of dbscan, k-means, and quadratic variation algorithms for automatic identification of swallows from swallowing accelerometry signals. *Computers in Biology and Medicine*, 59:10–18, 2015.

[7] Laura Falk, Atul Prakash, and Kevin Borders. Analyzing websites for user-visible security design flaws. In *Proceedings of the 4th Symposium on Usable Privacy and Security*, pages 117–126, 2008.

[8] Christos Faloutsos and King-Ip Lin. Fastmap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. *SIGMOD Rec.*, 24(2):163–174, may 1995.

[9] Shashank Gupta and Brij Bhooshan Gupta. Cross-site scripting (xss) attacks and defense mechanisms: classification and state-of-the-art. *International Journal of System Assurance Engineering and Management*, 8(1):512–530, 2017.

[10] William G Halfond, Jeremy Viegas, Alessandro Orso, et al. A classification of sql-injection attacks and countermeasures. In *Proceedings of the IEEE international symposium on secure software engineering*, volume 1, pages 13–15. IEEE, 2006.

[11] Henderi Henderi, Tri Wahyuningsih, and Efana Rahwanto. Comparison of min-max normalization and z-score normalization in the k-nearest neighbor (knn) algorithm to test the accuracy of types of breast cancer. *International Journal of Informatics and Information Systems*, 4(1):13–20, 2021.

[12] Nafiseh Hosseini, Fatemeh Fakhar, Behzad Kiani, and Saeid Eslami. Enhancing the security of patients' portals and websites by detecting malicious web crawlers using machine learning techniques. *International journal of medical informatics*, 132:103976, 2019.

[13] Denis V Ivanov, Dmitry A Moskvin, and GS Kubrin. Detection of security threats to modern websites. *Automatic Control and Computer Sciences*, 53(8):963–968, 2019.

[14] Mark Joswiak, You Peng, Ivan Castillo, and Leo H. Chiang. Dimensionality reduction for visualizing industrial chemical process data. *Control Engineering Practice*, 93:104189, 2019.

[15] Sung-Soo Kim, Sunhee Kwon, and Dianne Cook. Interactive visualization of hierarchical clusters using mds and mst. *Metrika*, 51(1):39–51, 2000.

[16] Alexander Lis. Comparison and analysis of web vulnerability scanners. B.S. thesis, 2019.

[17] Rina Elizabeth López de Jiménez. Pentesting on web applications using ethical - hacking. In *2016 IEEE 36th Central American and Panama Convention (CONCAPAN XXXVI)*, pages 1–6, 2016.

[18] Binu Melit Devassy and Sony George. Dimensionality reduction and visualisation of hyperspectral ink data using t-sne. *Forensic Science International*, 311:110194, 2020.

[19] Max Mignotte. Mds-based multiresolution nonlinear dimensionality reduction model for color image segmentation. *IEEE transactions on neural networks*, 22(3):447–460, 2011.

[20] Ismail Bin Mohamad and Dauda Usman. Standardization and its effects on k-means clustering algorithm. *Research Journal of Applied Sciences, Engineering and Technology*, 6(17):3299–3303, 2013.

[21] Md Moniruzzaman, Farida Chowdhury, and Md Sadek Ferdous. Measuring vulnerabilities of bangladeshi websites. In *2019 International Conference on Electrical, Computer and Communication Engineering (ECCE)*, pages 1–7, 2019.

[22] Sangeeta Nagpure and Sonal Kurkure. Vulnerability assessment and penetration testing of web application. In *2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA)*, pages 1–6, 2017.

[23] Nosa Omar Salih Omar et al. *A Proposed Method for Vulnerability Discovery of Sudanese Government Websites.* PhD thesis, Sudan University of Science & Technology, 2018.

[24] Eva Patel and Dharmender Singh Kushwaha. Clustering cloud workloads: K-means vs gaussian mixture model. *Procedia Computer Science*, 171:158–167, 2020. Third International Conference on Computing and Network Communications (CoCoNet'19).

[25] C Saranya and G Manikandan. A study on normalization techniques for privacy preserving data mining. *International Journal of Engineering and Technology (IJET)*, 5(3):2701–2704, 2013.

[26] Ketan Rajshekhar Shahapure and Charles Nicholas. Cluster quality analysis using silhouette score. In *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 747–748, 2020.

[27] Meshal Shutaywi and Nezamoddin N. Kachouie. Silhouette analysis for performance evaluation in machine learning with applications to clustering. *Entropy*, 23(6), 2021.

[28] N. Karthikeyani Visalakshi and J. Suguna. K-means clustering using max-min distance measure. In *NAFIPS 2009 - 2009 Annual Meeting of the North American Fuzzy Information Processing Society*, pages 1–6, 2009.

[29] Xu Wang and Yusheng Xu. An improved index for clustering validation based on silhouette index and calinski-harabasz index. *IOP Conference Series: Materials Science and Engineering*, 569(5):052024, jul 2019.

11 Appendix A

Generolo Jono Žemaičio Lietuvos karo akademija www.lka.lt
ISM Vadybos ir ekonomikos universitetas www.ism.lt
Kazimiero Simonavičiaus universitetas https://ksu.lt
Klaipėdos universitetas www.ku.lt
LCC tarptautinis universitetas https://lcc.lt
Lietuvos sporto universitetas www.lsu.lt
Lietuvos sveikatos mokslu universitetas www.lsmuni.lt
Mykolo Romerio universitetas www.mruni.eu
Telšiu Vyskupo Vincento Borisevičiaus kunigu seminarija www.tks.lt
Vilniaus dailės akademija www.vda.lt
Vilniaus Gedimino technikos universitetas https://vilniustech.lt
Vilniaus Šv. Juozapo kunigu seminarija www.seminarija.lt
Vilniaus universitetas www.vu.lt
Vilniaus universiteto Verslo mokykla www.vm.vu.lt/
Vytauto Didžiojo universitetas www.vdu.lt
Alytaus kolegija https://alytauskolegija.lt/en/
Kauno kolegija www.kaunokolegija.lt
Kauno mišku ir aplinkos inžinerijos kolegija www.kmaik.lt
Kauno technikos kolegija www.ktk.lt
Klaipėdos valstybinė kolegija www.kvk.lt
Kolpingo kolegija https://www.kolpingokolegija.lt/
Lietuvos aukštoji jūreivystės mokykla www.lajm.lt
Lietuvos verslo kolegija https://www.ltvk.lt/
Marijampolės kolegija https://marko.lt
Panevėžio kolegija https://panko.lt
Socialiniu mokslu kolegija www.smk.lt
Dieveniškiu technologiju ir verslo mokykla www.dieveniskiutvm.lt
Amatu mokykla Sodžiaus meistrai“ https://sodziausmeistrai.lt
Aukštadvario žemės ūkio mokykla www.gimnazija.aukstadvaris.lm.lt
Elektrėnu profesinio mokymo centras www.epmc.lt
Mokykla Art of Beauty“ www.mokyklaaob.lt
Balbieriškis Jonavos politechnikos mokykla www.jpm.lt
Kaišiadoriu technologiju ir verslo mokykla www.zum.kaisiadorys.lm.lt
Karaliaus Mindaugo Profesinio Mokymo Centras www.kaupa.lt
Kauno Dainavos darbo rinkos mokymo centras www.kaunodrmc.lt
Kauno maisto pramonės mokykla https://mpcentras.lt
Alytaus profesinio rengimo centras www.aprc.lt
Šakiu Varpo mokykla https://www.varpas.lt/
Plungės Ryto pagrindinė mokykla https://ryto.plunge.lm.lt/
Šakiu Varpo mokykla www.varpas.lt/
Plungės Ryto pagrindinė mokykla www.ryto.plunge.lm.lt
Biržu Aušros” pagrindinė mokykla https://birzuausra.lt

Ukmergės Dukstynos pagrindinė mokykla www.dukstyna.lt/naujienos.htm
Vilniaus Žygimanto Augusto progimnazija www.zaugusto.lt
Druskininku Atgimimo“ mokykla www.atgimimomokykla.lt
Gargždu Kranto“ progimnazija www.krantopm.lt
Joniškio Saulės” pagrindinė mokykla https://saule.joniskis.lm.lt
Panevėžio Mykolo Karkos pagrindinė mokykla www.karkosm.lt
Ignalinos r. Didžiasalio Ryto“ gimnazijos Dūkšto skyrius www.dukstomokykla.lt
Biržu r. Vabalninko Balio Sruogos gimnazija www.vabalninkomokykla.lt
Biržu ,,Atžalyno” pagrindinės mokykla www.atzalynas.birzai.lm.lt
Šiauliu universitetinė gimnazija www.sugimnazija.lt
Skaistgirio gimnazija www.skaistgiris.joniskis.lm.lt
Vytauto Didžiojo universiteto Atžalyno“ progimnazija https://vduprogimnazija.lt
KTU inžinerijos licėjus www.inzinerijoslicejus.ktu.edu
Kauno Vyturio“ gimnazija https://vyturys.kaunas.lm.lt
Kauno Juozo Grušo meno gimnazija https://grusas.kaunas.lm.lt
Kauno Juozo Urbšio progimnazija https://urbsys.kaunas.lm.lt
Jurgio Dobkevičiaus progimnazija www.dobkevicius.kaunas.lm.lt
Kauno Kovo 11-osios gimnazija https://kovo11gimnazija.lt
Kauno Palemono gimnazija www.palemonas.kaunas.lm.lt
Šaukėnu Vlado Pūtvio-Putvinskio gimnazija www.saukenugimnazija.lt
Vaivorykštės tako gimnazija http://vaivorykstestakas.lt
Klaipėdos Vydūno gimnazija www.vyduno.lt
Mažeikiu Sodu pagrindinė mokykla https://sodu.mazeikiai.lm.lt
Neringos gimnazija http://neringosgimnazija.lt
Lygumu pagrindinė mokykla www.lygumai.pakruojis.lm.lt
Lygumu pagrindinė mokykla https://vaskai.pasvalys.lt
Kuliu gimnazija http://www.kuliai.lt
Vilniaus Simono Stanevičiaus progimnazija https://staneviciaus.lt/
Vilniaus Sofijos Kovalevskajos gimnazija http://www.sofijoskovalevskajosmokykla.lt/
Antazavės Juozo Gruodžio pagrindinė mokykla http://antazavesmokykla.lt/
Ugdymo ir Laisvalaikio Centras AVILIUKAS” http://www.aviliukas.com/
Mamos delne Vaiku Darželis https://mamosdelne.lt/
Vaiku darželis Pasaku namukas“ https://www.pasakunamukas.lt/
PĖDUTĖS” Vaiku Lopšelis – Darželis https://www.pedutes.lt/
Krikščioniškas vaiku darželis https://krikscioniskasdarzelis.lt/
MŪSU PĖDUTĖS https://www.musupedutes.lt/
Ademilis“ https://www.ademilis.lt/
Parko jonvabaliai“ — privatus darželis Pilaitėje https://parkojonvabaliai.lt/
MINI DARŽELIAI https://www.minidarzeliai.lt/
Šilutės lopšelis - darželis Raudonkepuraitė“ https://www.raudonkepuraite.lt/
Vilniaus nevalstybinis vaiku lopšelis – darželis Rudnosiukas”. https://rudnosiukas.eu/
Kauno lopšelis-darželis Rokutis“ https://www.ldrokutis.lt/
Lopšelis-darželis Žiogelis“ https://darzelis.lt/
Vilniaus lopšelis-darželis Nykštukas“ http://www.nykstukasld.lt/
Kauno menu darželis ETIUDAS https://etiudas.kaunas.lm.lt/

Kauno lopšelis-darželis "Klevelis" https://www.klevelis.lt/
Kauno Montesori mokykla-darželis ,,Žiburėlis" https://mokyklaziburelis.lt/
Vilniaus lopšelis darželis "Žemyna" http://www.zemyna.vilnius.lm.lt/
Lopšelis – darželis "Žara" https://www.zara.kaunas.lm.lt/
Kauno lopšelis-darželis Tukas" https://www.darzelistukas.lt/
Kauno mokykla-darželis Šviesa "https://www.mokyklasviesa.lt/
Spindulys – Darželis https://www.spindulys.kaunas.lm.lt/en/
Darželis-mokykla "Saulutė" https://www.saulute.vilnius.lm.lt/
Kauno mokykla-darželis "RŪTELĖ" https://www.mokyklarutele.lt/
Kauno lopšelis-darželis Giliukas" https://www.kaunogiliukas.lt/
Alytaus lopšelis-darželis "Du gaideliai" https://www.alytausdugaideliai.lt/
Alytaus lopšelis-darželis ,,Girinukas" https://www.ldgirinukas.lt/
Alytaus lopšelis-darželis LINELIS https://alytauslinelis.lt/
Birštono vaiku lopšelis-darželis ,,Vyturėlis" http://www.vyturelis.com/
Biržu lopšelis-darželis "DRUGELIS" https://www.lddrugelis.lt/

12 Appendix B

N-Stalker

Website link	URLs	High	Medium	Low
http://antazavesmokykla.lt/	258	103	20	0
http://neringosgimnazija.lt	500	23	32	0
http://vaivorykstestakas.lt	58	2	4	0
http://www.aviliukas.com/	20	0	2	1
http://www.epmc.lt	500	49	65	1
http://www.kuliai.lt	500	6	178	0
http://www.nykstukasld.lt/	123	0	7	1
http://www.sofijoskovalevskajosmokykla.lt/	184	6	5	0
http://www.vyturelis.com/	75	0	2	0
http://www.zemyna.vilnius.lm.lt/	240	3	6	0
https://alytauskolegija.lt/lt/	3	1	3	0
https://alytauslinelis.lt/	500	5	5	0
https://birzuausra.lt	500	0	4	1
https://darzelis.lt/	500	4	168	1
https://etiudas.kaunas.lm.lt/	500	0	7	1
https://grusas.kaunas.lm.lt	500	1	5	0
https://kovo11gimnazija.lt	500	12	26	2
https://krikscioniskasdarzelis.lt	180	1	5	1
https://ksu.lt	500	1	3	1
https://lcc.lt	500	1	5	2
https://lsmu.lt	500	114	101	5
https://ltvk.lt	291	0	0	0
https://mamosdelne.lt/	124	0	3	0
https://marko.lt	500	0	5	2
https://mokyklaziburelis.lt/	502	1	6	49
https://mpcentras.lt	500	2	8	0
https://panko.lt	500	2	4	1
https://parkojonvabaliai.lt/	99	2	6	1
https://rudnosiukas.eu/	158	0	3	0
https://ryto.plunge.lm.lt	2	0	1	0
https://saule.joniskis.lm.lt	105	0	5	0
https://sodu.mazeikiai.lm.lt	500	0	3	0
https://sodziausmeistrai.lt	500	1	1	0
https://staneviciaus.lt/	500	1	9	11
https://urbsys.kaunas.lm.lt	500	0	8	1
https://vaskai.pasvalys.lt	500	1	5	0
https://vduprogimnazija.lt	500	1	8	1
https://vilniustech.lt	500	1	3	2

https://vyturys.kaunas.lm.lt	500	4	7	1
https://www.ademilis.lt/	88	1	5	0
https://www.alytausdugaideliai.lt/	500	8	6	1
https://www.aprc.lt	281	0	8	2
https://www.darzelistukas.lt/	502	13	20	1
https://www.dieveniskiutvm.lt/index.php/lt/	266	2	8	2
https://www.dukstyna.lt/naujienos.htm	500	0	2	0
https://www.gimnazija.aukstadvaris.lm.lt	500	2	5	0
https://www.ism.lt	500	6	3	1
https://www.jpm.lt	219	244	69	3
https://www.kaunodrmc.lt/	117	0	2	1
https://www.kaunogiliukas.lt/	399	5	5	0
https://www.kaunokolegija.lt	500	2	11	1
https://www.kaupa.lt	500	2	5	1
https://www.klevelis.lt/	108	2	5	0
https://www.kmaik.lt	500	0	4	1
https://www.kolpingokolegija.lt/	500	4	7	0
https://www.ktk.lt	500	0	2	1
https://www.ku.lt	500	2	2	0
https://www.kvk.lt	500	0	2	1
https://www.lajm.lt	500	4	8	17
https://www.lddrugelis.lt/	500	5	6	0
https://www.ldgirinukas.lt/	397	15	20	1
https://www.ldrokutis.lt/	500	5	5	1
https://www.lka.lt	500	0	4	1
https://www.lsu.lt	500	3	8	0
https://www.minidarzeliai.lt/	155	2	7	1
https://www.mokyklaaob.lt	500	29	39	2
https://www.mokyklarutele.lt/	500	1	4	51
https://www.mokyklasviesa.lt/	500	4	4	0
https://www.mruni.eu	500	21	28	2
https://www.musupedutes.lt/	57	0	3	0
https://www.pasakunamukas.lt/	63	4	3	1
https://www.pedutes.lt/	32	0	0	0
https://www.raudonkepuraite.lt/	367	0	5	1
https://www.seminarija.lt	500	0	3	1
https://www.spindulys.kaunas.lm.lt/lt/	249	1	4	0
https://www.tks.lt	252	227	39	0
https://www.varpas.lt	500	8	14	0
https://www.vda.lt	500	0	4	2
https://www.vdu.lt/lt/	500	1	4	1
https://www.vm.vu.lt	500	22	35	1

https://www.vu.lt	500	0	0	0
https://www.zara.kaunas.lm.lt/	338	14	18	0
https://www.atgimimomokykla.lt	500	1	6	0
https://www.atzalynas.birzai.lm.lt	500	23	272	73
https://www.dobkevicius.kaunas.lm.lt	422	1	8	1
https://www.dukstomokykla.lt	500	2	10	6
https://www.inzinerijoslicejus.ktu.edu	500	0	353	2
https://www.karkosm.lt	500	1	6	0
https://www.krantopm.lt	399	3	4	0
https://www.lygumai.pakruojis.lm.lt	500	0	5	0
https://www.palemonas.kaunas.lm.lt	500	0	9	0
https://www.saukenugimnazija.lt	500	3	7	1
https://www.saulute.vilnius.lm.lt/?page_id=277&lang=lt	43	3	4	0
https://www.skaistgiris.joniskis.lm.lt	210	0	6	2
https://www.smk.lt	500	0	6	1
https://www.sugimnazija.lt	500	1	4	1
https://www.vabalninkomokykla.lt	77	0	6	0
https://www.vyduno.lt	500	3	6	0
https://www.zaugusto.lt	500	1	11	3
https://www.zum.kaisiadorys.lm.lt	75	0	2	1

ZAP

Website link	URLs	High	Medium	Low
http://antazavesmokykla.lt/	11467	1	4	8
http://neringosgimnazija.lt	2501	2	4	5
http://vaivorykstestakas.lt	5275	1	4	3
http://www.aviliukas.com/	358	0	4	5
http://www.epmc.lt	11268	0	4	9
http://www.kuliai.lt	10483	1	4	1
http://www.nykstukasld.lt/	334	1	4	4
http://www.sofijoskovalevskajosmokykla.lt/	10993	1	5	8
http://www.vyturelis.com/	1116	0	2	1
http://www.zemyna.vilnius.lm.lt/	818	0	4	3
https://alytauskolegija.lt/en/	19262	0	4	5
https://alytauslinelis.lt/	4442	2	4	4
https://birzuausra.lt	13961	2	4	6
https://darzelis.lt/	2330	2	4	8
https://etiudas.kaunas.lm.lt/	4095	1	4	6
https://grusas.kaunas.lm.lt//	23558	1	4	4
https://kovo11gimnazija.lt	2401	1	4	11
https://krikscioniskasdarzelis.lt/	2072	1	5	8
https://ksu.lt	3376	1	4	4
https://lcc.lt	2365	0	3	10
https://lsmu.lt	203905	1	7	11
https://ltvk.lt	29203	1	3	9
https://mamosdelne.lt/	541	0	3	4
https://marko.lt	20256	2	4	8
https://mokyklaziburelis.lt/	2659	1	4	12
https://mpcentras.lt	2146	1	5	8
https://panko.lt	4816	0	4	7
https://parkojonvabaliai.lt/	315	0	4	7
https://rudnosiukas.eu/	461	1	2	4
https://ryto.plunge.lm.lt	6574	1	6	5
https://saule.joniskis.lm.lt	20439	2	4	3
https://sodu.mazeikiai.lm.lt	4051	1	4	4
https://sodziausmeistrai.lt	5489	2	3	6
https://staneviciaus.lt/	5447	2	4	14
https://urbsys.kaunas.lm.lt	32596	2	3	4
https://vaskai.pasvalys.lt	7929	2	3	3
https://vduprogimnazija.lt	3024	2	4	10
https://vilniustech.lt	30421	2	4	8
https://vyturys.kaunas.lm.lt	25840	1	4	5
https://www.ademilis.lt/	297	0	4	4
https://www.alytausdugaideliai.lt/	1659	2	5	6
https://www.aprc.lt	4402	1	6	6

https://www.darzelistukas.lt/	9349	2	4	8
https://www.dieveniskiutvm.lt/index.php/lt/	722	0	7	6
https://www.dukstyna.lt/naujienos.htm	2147	2	4	2
https://www.gimnazija.aukstadvaris.lm.lt	5131	2	5	1
https://www.ism.lt	13586	1	4	7
https://www.jpm.lt	2084	2	8	6
https://www.kaunodrmc.lt/	281	0	6	6
https://www.kaunogiliukas.lt/	5016	0	4	4
https://www.kaunokolegija.lt	42637	2	5	11
https://www.kaupa.lt	14128	2	5	8
https://www.klevelis.lt/	1415	2	4	7
https://www.kmaik.lt	4122	1	6	9
https://www.kolpingokolegija.lt/	16763	2	6	9
https://www.ktk.lt	5757	2	2	5
https://www.ku.lt	9295	1	3	7
https://www.kvk.lt	18845	2	7	11
https://www.lajm.lt	5201	0	3	9
https://www.lddrugelis.lt/	6930	1	4	5
https://www.ldgirinukas.lt/	1574	1	6	5
https://www.ldrokutis.lt/	4572	2	4	6
https://www.lka.lt	15293	2	6	9
https://www.lsu.lt	12863	2	4	7
https://www.minidarzeliai.lt/	431	0	4	7
https://www.mokyklaaob.lt	2104	2	4	9
https://www.mokyklarutele.lt/	3442	2	4	13
https://www.mokyklasviesa.lt/	6173	1	4	3
https://www.mruni.eu	94336	2	6	10
https://www.musupedutes.lt/	1888	0	2	2
https://www.pasakunamukas.lt/	227	0	3	6
https://www.pedutes.lt/	677	0	3	4
https://www.raudonkepuraite.lt/	3407	1	4	8
https://www.seminarija.lt	2567	2	2	6
https://www.spindulys.kaunas.lm.lt/lt/	3113	2	4	4
https://www.tks.lt	602	0	4	5
https://www.varpas.lt	3077	1	5	3
https://www.vda.lt	59090	2	4	10
https://www.vdu.lt/lt/	14725	1	5	6
https://www.vm.vu.lt	1820	1	4	9
https://www.vu.lt	22352	2	6	8
https://www.zara.kaunas.lm.lt/	1951	2	4	2
https://www.atgimimomokykla.lt	daug	1	3	4
https://www.atzalynas.birzai.lm.lt	16005	1	4	6
https://www.dobkevicius.kaunas.lm.lt	1763	1	3	5
https://www.dukstomokykla.lt	22528	2	4	7

https://www.inzinerijoslicejus.ktu.edu	6519	0	4	10
https://www.karkosm.lt	20249	1	4	4
https://www.krantopm.lt	2151	2	6	5
https://www.lygumai.pakruojis.lm.lt	5366	2	3	3
https://www.palemonas.kaunas.lm.lt	9381	2	4	4
https://www.saukenugimnazija.lt	5471	2	4	6
https://www.saulute.vilnius.lm.lt/?page_id=277&lang=lt	6481	0	4	3
https://www.skaistgiris.joniskis.lm.lt	4358	1	4	5
https://www.smk.lt	6133	2	6	10
https://www.sugimnazija.lt	2244	1	4	6
https://www.vabalninkomokykla.lt	1470	1	4	3
https://www.vyduno.lt	22425	2	4	4
https://www.zaugusto.lt	1696	1	5	11
https://www.zum.kaisiadorys.lm.lt	915	1	6	3

13 Appendix C

```
import pandas as pd
from matplotlib import pyplot as plt
from sklearn import mixture
from sklearn.manifold import TSNE, MDS
from sklearn.metrics import silhouette_score, calinski_harabasz_score
from sklearn.mixture import GaussianMixture
from sklearn.preprocessing import MinMaxScaler
from sklearn.cluster import DBSCAN, KMeans

# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Load the Data
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
df1 = pd.read_excel(r'C:\Users\***\***\ZAP-Final.xlsx')
df2 = pd.read_excel(r'C:\Users\***\***\N-Stalker-Final.xlsx')
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Apply min-max scaling:
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
X1 = df1.columns[0:34]
X2 = df2.columns[0:103]
scaler =MinMaxScaler()
scaled_data1 =scaler.fit_transform(df1[X1])
scaled_data2 =scaler.fit_transform(df2[X2])
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# t-SNE
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
tsne1 =TSNE(n_components=2, perplexity=40, learning_rate=70, n_iter=360)
tsne2 =TSNE(n_components=2, perplexity=40, learning_rate=70, n_iter=360)
X_tsne1 =tsne1.fit_transform(scaled_data1)
X_tsne2 =tsne2.fit_transform(scaled_data2)
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# MDS
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
mds1 =MDS(n_components=2, metric=False, eps=1e-3, n_init=1)
mds2 =MDS(n_components=2, metric=False, eps=1e-3, n_init=1)
X_MDS1 =mds1.fit_transform(scaled_data1)
X_MDS2 =mds2.fit_transform(scaled_data2)
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# DBSCAN clustering method
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
dbscan1 =DBSCAN(eps=0.6, min_samples=3)
dbscan2 =DBSCAN(eps=1.6, min_samples=3)
dbscan1.fit(scaled_data1)
dbscan2.fit(scaled_data2)
labelsD1 =dbscan1.labels_
labelsD2 =dbscan2.labels_
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# DBSCAN number of clusters per eps for ZAP
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
score = []
for n in range(10, 300):
    dbscan1 =DBSCAN(eps=n/100, min_samples=3)
```

```
    dbscan1.fit(scaled_data1)
    score.append(max(dbscan1.labels_))
    print(n, max(dbscan1.labels_))
plt.plot(range(10, 300), score)
plt.xticks(range(10, 300))
plt.xlabel("Epsilon * 100")
plt.ylabel("Cluster Count")
plt.show()
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# DBSCAN number of clusters per eps for N-Stalker
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
score = []
for n in range(10, 300):
    dbscan2 =DBSCAN(eps=n/100, min_samples=3)
    dbscan2.fit(scaled_data2)
    score.append(max(dbscan2.labels_))
    print(n, max(dbscan2.labels_))
plt.plot(range(10, 300), score)
plt.xticks(range(10, 300))
plt.xlabel("Epsilon * 100")
plt.ylabel("Cluster Count")
plt.show()
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Removing the outliers for Silhouette Score and C-H Index for DBSCAN
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
outlier_detection =DBSCAN(
  eps=0.17,
  metric="euclidean",
  min_samples=3,
  n_jobs=-1)
outliers1 =outlier_detection.fit_predict(scaled_data1)
outliers2 =outlier_detection.fit_predict(scaled_data2)
df_clean1 =scaled_data1[outliers1 !=-1]
df_clean2 =scaled_data2[outliers2 !=-1]
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Silhouette score without the outliers for DBSCAN
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
scoreD1 =silhouette_score(df_clean1, outliers1[outliers1 !=-1])
scoreD2 =silhouette_score(df_clean2, outliers2[outliers2 !=-1])
print(f"Silhouette score: {scoreD1:.3f}")
print(f"Silhouette score: {scoreD2:.3f}")
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Calinski-Harabasz index without the outliers for DBSCAN
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
scoreD1 =calinski_harabasz_score(df_clean1, outliers1[outliers1 !=-1])
scoreD2 =calinski_harabasz_score(df_clean2, outliers2[outliers2 !=-1])
print(f"Calinski-Harabasz index: {scoreD1:.3f}")
print(f"Calinski-Harabasz index: {scoreD2:.3f}")
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Scatter plot of DBSCAN clusters
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
plt.scatter(X_tsne1[:, 0], X_tsne1[:, 1], c=labelsD1)
plt.xlabel("DBSCAN with t-SNE of ZAP scanner")
plt.show()
```

```
plt.scatter(X_tsne2[:, 0], X_tsne2[:, 1], c=labelsD2)
plt.xlabel("DBSCAN with t-SNE of N-Stalker scanner")
plt.show()
plt.scatter(X_MDS1[:, 0], X_MDS1[:, 1], c=labelsD1)
plt.xlabel("DBSCAN with MDS of ZAP scanner")
plt.show()
plt.scatter(X_MDS2[:, 0], X_MDS2[:, 1], c=labelsD2)
plt.xlabel("DBSCAN with MDS of N-Stalker scanner")
plt.show()
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# K-Means
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
kmeans1 =KMeans(n_clusters=4)
kmeans2 =KMeans(n_clusters=4)
kmeans1.fit(scaled_data1)
kmeans2.fit(scaled_data2)
labelsK1 =kmeans1.labels_
labelsK2 =kmeans2.labels_
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Elbow Method
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
kmeans_kwargs ={
    "init": "random",
    "n_init": 10,
    "random_state": 1,
}
sse = []
for k in range(1, 11):
    kmeans =KMeans(n_clusters=k, **kmeans_kwargs)
    kmeans.fit(scaled_data1)
    sse.append(kmeans.inertia_)
plt.plot(range(1, 11), sse)
plt.xticks(range(1, 11))
plt.xlabel("Number of Clusters for ZAP")
plt.ylabel("SSE")
plt.show()

sse = []
for k in range(1, 11):
    kmeans =KMeans(n_clusters=k, **kmeans_kwargs)
    kmeans.fit(scaled_data2)
    sse.append(kmeans.inertia_)
plt.plot(range(1, 11), sse)
plt.xticks(range(1, 11))
plt.xlabel("Number of Clusters for ZAP")
plt.ylabel("SSE")
plt.show()
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Silhouette score for K-Means
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
scoreK1 =silhouette_score(scaled_data1, labelsK1)
scoreK2 =silhouette_score(scaled_data2, labelsK2)
print(f"Silhouette score: {scoreK1:.3f}")
print(f"Silhouette score: {scoreK2:.3f}")
```

```
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Calinski-Harabasz index for K-Means
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
scoreK1 =calinski_harabasz_score(scaled_data1, labelsK1)
scoreK2 =calinski_harabasz_score(scaled_data2, labelsK2)
print(f"Calinski-Harabasz index: {scoreK1:.3f}")
print(f"Calinski-Harabasz index: {scoreK2:.3f}")
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Scatter plot of K-Means clusters
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
plt.scatter(X_tsne1[:, 0], X_tsne1[:, 1], c=labelsK1)
plt.xlabel("K-Means with t-SNE of ZAP scanner")
plt.show()
plt.scatter(X_tsne2[:, 0], X_tsne2[:, 1], c=labelsK2)
plt.xlabel("K-Means with t-SNE of N-Stalker scanner")
plt.show()
plt.scatter(X_MDS1[:, 0], X_MDS1[:, 1], c=labelsK1)
plt.xlabel("K-Means with MDS of ZAP scanner")
plt.show()
plt.scatter(X_MDS2[:, 0], X_MDS2[:, 1], c=labelsK2)
plt.xlabel("K-Means with MDS of N-Stalker scanner")
plt.show()
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Gaussian Mixture
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
gm1 = GaussianMixture(n_components=2)
gm2 = GaussianMixture(n_components=2)
gm1.fit(scaled_data1)
gm2.fit(scaled_data2)
labelsG1 =gm1.predict(scaled_data1)
labelsG2 =gm2.predict(scaled_data2)
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# BIC & NIC
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
bic = []
for n in range(1, 10):
    gmm =mixture.GaussianMixture(n_components=n)
    gmm.fit(scaled_data1)
    bic.append(gmm.bic(scaled_data1))
aic = []
for n in range(1, 10):
    gmm =mixture.GaussianMixture(n_components=n)
    gmm.fit(scaled_data1)
    aic.append(gmm.aic(scaled_data1))
plt.plot(bic, label='BIC')
plt.plot(aic, label='AIC')
plt.legend()
plt.xlabel("n_components")
plt.show()

bic = []
for n in range(1, 10):
    gmm =mixture.GaussianMixture(n_components=n)
    gmm.fit(scaled_data2)
```

```
    bic.append(gmm.bic(scaled_data2))
aic = []
for n in range(1, 10):
    gmm =mixture.GaussianMixture(n_components=n)
    gmm.fit(scaled_data2)
    aic.append(gmm.aic(scaled_data2))
plt.plot(bic, label='BIC')
plt.plot(aic, label='AIC')
plt.legend()
plt.xlabel("n_components")
plt.show()
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Silhouette score for Gaussian Mixture
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
scoreG1 =silhouette_score(scaled_data1, labelsG1)
scoreG2 =silhouette_score(scaled_data2, labelsG2)
print(f"Silhouette score: {scoreG1:.3f}")
print(f"Silhouette score: {scoreG2:.3f}")
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Calinski-Harabasz index for Gaussian Mixture
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
scoreG1 =calinski_harabasz_score(scaled_data1, labelsG1)
scoreG2 =calinski_harabasz_score(scaled_data2, labelsG2)
print(f"Calinski-Harabasz index: {scoreG1:.2f}")
print(f"Calinski-Harabasz index: {scoreG2:.2f}")
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Scatter plot of Gaussian Mixture clusters
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
plt.scatter(X_tsne1[:, 0], X_tsne1[:, 1], c=labelsG1)
plt.xlabel("Gaussian Mixture with t-SNE of ZAP scanner")
plt.show()
plt.scatter(X_tsne2[:, 0], X_tsne2[:, 1], c=labelsG2)
plt.xlabel("Gaussian Mixture with t-SNE of N-Stalker scanner")
plt.show()
plt.scatter(X_MDS1[:, 0], X_MDS1[:, 1], c=labelsG1)
plt.xlabel("Gaussian Mixture with MDS of ZAP scanner")
plt.show()
plt.scatter(X_MDS2[:, 0], X_MDS2[:, 1], c=labelsG2)
plt.xlabel("Gaussian Mixture with MDS of N-Stalker scanner")
plt.show()
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Create a new column in each dataframe to store the cluster predictions & Compare the
                                             cluster predictions for each row in the two
                                             dataframes
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
df1['cluster'] =labelsD1
df2['cluster'] =labelsD2
differences =0
for i in range(len(df1)):
    if df1.iloc[i]['cluster'] !=df2.iloc[i]['cluster']:
        differences +=1
percent_different =differences /len(df1) *100
print(f"{percent_different:.2f}% of the rows have different cluster predictions with
                                             DBSCAN clustering.")
```

```
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
df1['cluster'] =labelsK1
df2['cluster'] =labelsK2
differences =0
for i in range(len(df1)):
    if df1.iloc[i]['cluster'] !=df2.iloc[i]['cluster']:
        differences +=1
percent_different =differences /len(df1) *100
print(f"{percent_different:.2f}% of the rows have different cluster predictions with
                                                 K-Means clustering.")
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
df1['cluster'] =labelsG1
df2['cluster'] =labelsG2
differences =0
for i in range(len(df1)):
    if df1.iloc[i]['cluster'] !=df2.iloc[i]['cluster']:
        differences +=1
percent_different =differences /len(df1) *100
print(f"{percent_different:.2f}% of the rows have different cluster predictions with
                                                 Gaussian Mixture clustering.")
```