Exploring Software Vulnerability Prediction

Part1.Data Collection on Software Vulnerability Researchs from Scopus Platform

Overview

To assemble an extensive dataset on software vulnerability research, a detailed data collection process was implemented using the Scopus database. Scopus, renowned for its vast repository of peer-reviewed literature, served as the primary source for sourcing relevant academic papers in this specialized field. The aim was to capture a broad spectrum of research articles addressing various facets of software vulnerability, encompassing detection, management, prevention, and analysis.

Methodology

The data collection was executed through two specific queries on Scopus:

Primary Query Focused on Software Vulnerability:

- Query: ("software vulnerability" OR "software security" OR "vulnerability detection" OR "software exploit" OR "cybersecurity" AND "software")

TITLE-ABS-KEY (software AND vulnerability AND prediction) AND (LIMIT-TO (SUBJAREA , "COMP") OR LIMIT-TO (SUBJAREA , "ENGI")) AND (LIMIT-TO (DOCTYPE , "cp") OR LIMIT-TO (DOCTYPE , "ar"))

- Objective: To target papers that explicitly discuss software vulnerabilities.
- Purpose: This query was designed to fetch publications primarily contributing to the topic of software vulnerabilities, ensuring high relevance to the main subject matter.

Secondary, Broader Query:

- Query: ("software development" OR "security breach" OR "cyber threat" OR "software protection" OR "information security" AND "software")

TITLE-ABS-KEY ("software vulnerability prediction" OR "vulnerability assessment" OR "security analysis") AND PUBYEAR > 2013 AND PUBYEAR < 2025 AND (LIMIT-TO (SRCTYPE , "j") OR LIMIT-TO (SRCTYPE , "t LIMIT-TO SRCTYPE") OR EXCLUDE (SRCTYPE , "AND")) AND (

LIMIT-TO (SUBJAREA , "ENGI") OR LIMIT-TO (SUBJAREA , "t LIMIT-TO SUBJAREA") OR EXCLUDE (SUBJAREA , "AND") OR LIMIT-TO (SUBJAREA , "COMP")) AND (LIMIT-TO (DOCTYPE , "cp") OR LIMIT-TO (DOCTYPE , "t LIMIT-TO DOCTYPE") OR EXCLUDE (DOCTYPE , "AND")) AND (EXCLUDE (LANGUAGE , "AND"))

- Objective: To include a wider range of papers that indirectly contribute to the field.
- Purpose: The aim was to encompass studies that, while not exclusively centered on software vulnerabilities, offer relevant insights or data.

Results and Implications

The deployment of these two queries led to the accumulation of a rich dataset from Scopus, covering a diverse range of publications. This dataset is poised to offer a comprehensive perspective on software vulnerability research, incorporating both direct and related topics. The meticulous approach in constructing the queries ensures that the dataset serves as a solid basis for subsequent analysis and exploration in the domain of software vulnerability.

The dataset contains various columns related to software vulnerability publications. Here are the first few rows to give you an idea of the structure:

Authors	**Author full names**	**Author(s) ID**	**Title**	**Year**	**Source title**	**...**
...	...	...	...	...	...	...

For a thorough Exploratory Data Analysis (EDA), we can perform the following steps:

Basic Information: Understanding the shape of the dataset, column data types, and missing values.
Descriptive Statistics: Overview of the statistics for numerical columns.
Data Distribution: Analysis of the distribution of various features, especially categorical ones like the year of publication, source titles, and document types.
Correlation Analysis: If there are numerical features, checking for any correlation between them.

Text Analysis: Since this dataset involves a lot of textual data (like titles, abstracts), we can explore the most common words or phrases.

Let's start with the first two steps: Basic Information and Descriptive Statistics.

Exploratory Data Analysis Summary

Basic Information

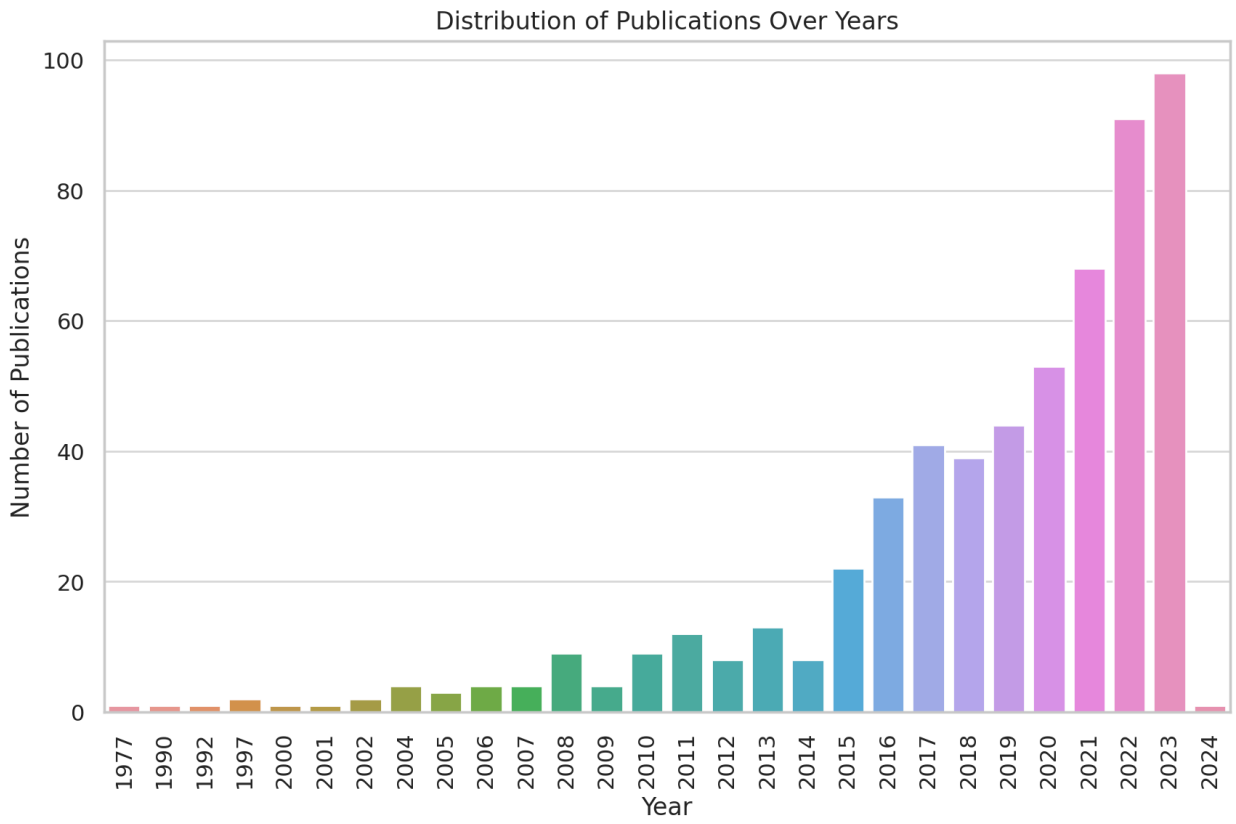
- The dataset contains 577 entries and 23 columns.
- Data types include int64 for year, float64 for page-related columns and object (which is typically string or mixed types) for the rest.
- Some columns have a significant number of missing values. For example, 'Cited by' is entirely empty, and 'Volume', 'Issue', 'Art. No.', and 'Open Access' have many missing entries.

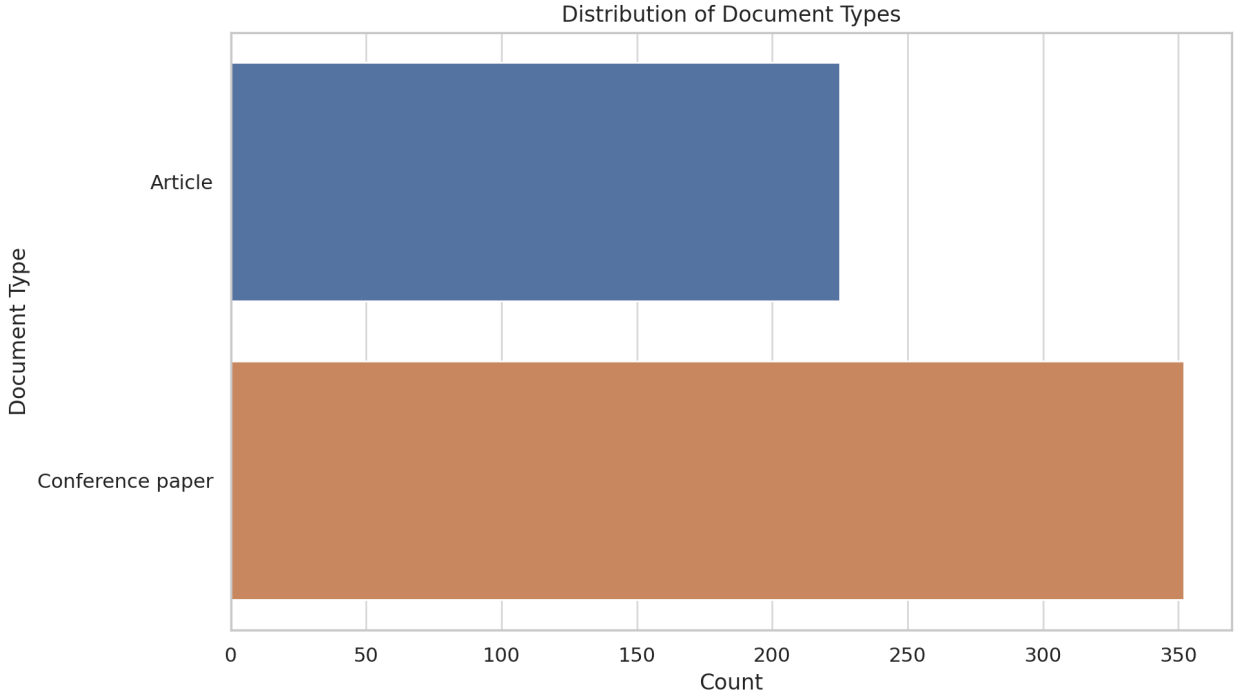
Descriptive Statistics (for numerical columns)

- Year: The publications range from 1977 to 2024, with a mean year of publication around 2018.
- Page Start: Varied starting page numbers, some unusually high, suggesting either large volumes or data entry errors.
- Page Count: On average, the documents span about 10 pages, with a maximum of 76 pages.

Next Steps

Data Distribution Analysis: We will analyze the distribution of categorical features like 'Year', 'Source title', and 'Document Type'.
Text Analysis: This would involve examining text data like titles and abstracts to identify common themes or keywords.


Distribution of Publications Over Years
Number of Publications
0
20
40
60
80
100
1977
1990
1992
1997
2000
2001
2002
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
Year


Distribution of Document Types
Document Type
Article
Conference paper
0
50
100
150
200
250
300
350
Count

Data Distribution Analysis

1. Distribution of Publications Over Years

- The dataset shows a distribution of publications from 1977 to 2024.
- There seems to be an increase in publications over recent years, indicating growing interest or research activity in the field of software vulnerability.

2. Distribution of Document Types

- The dataset includes various types of documents, such as articles, conference papers, etc.
- Some document types are more frequent than others, reflecting the common publication types in this research area.

Next Step: Text Analysis

For the text analysis, we will focus on the 'Title' and 'Abstract' columns to identify common themes or keywords. This involves processing the text data to find the most frequently occurring words, which can provide insights into the main focus areas of the research in the dataset.

Word Cloud for Titles

Text Analysis: Word Clouds

Word Cloud for Titles

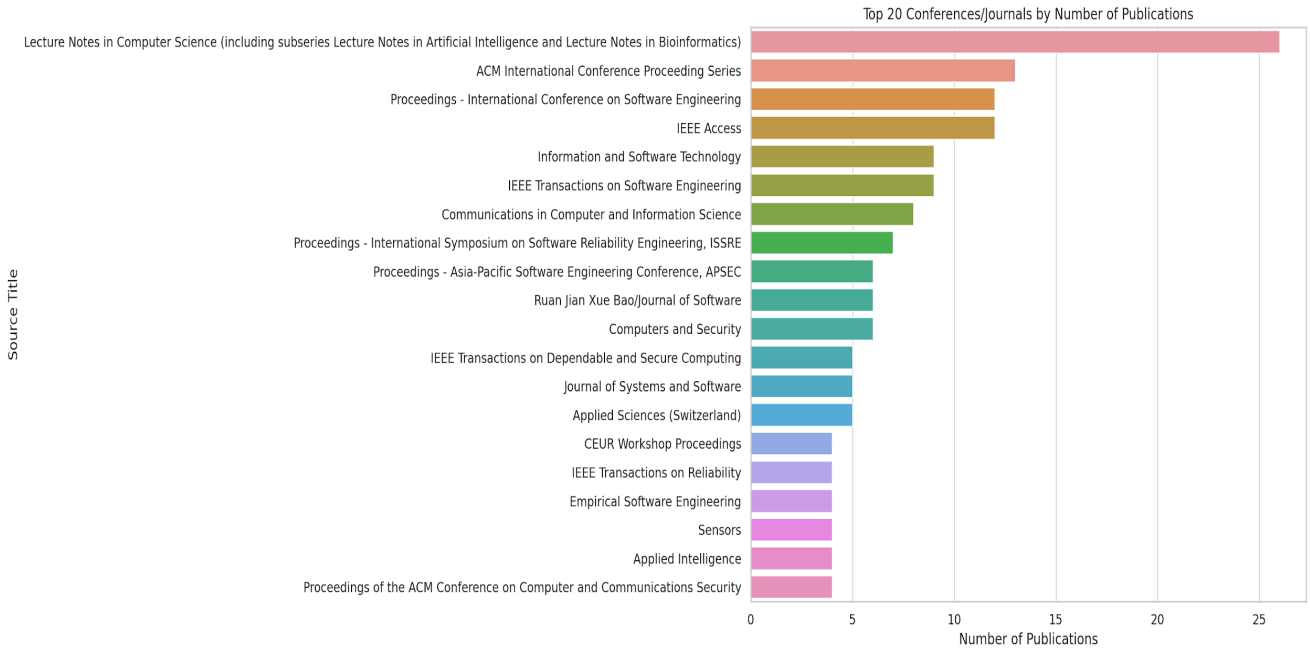
- The word cloud generated from the titles in the dataset reveals the most frequently occurring words, indicating the key topics in the research titles.
- Words like "Software", "Security", "Vulnerability", "Analysis", and "Detection" appear prominently, suggesting these are central themes in the dataset.

Word Cloud for Abstracts

- Similarly, the word cloud for abstracts highlights the most common words found in the abstracts of these publications.
- This visualization further emphasizes the focus on software security and vulnerability, along with terms like "systems", "applications", and "methods".

Conferences And Journals

The bar plot below displays the top 20 conferences or journals by the number of publications in the dataset. This visualization helps to identify which sources are most frequently publishing research in the area of software vulnerability. The 'Source Title' on the y-axis represents different conferences and journals, and the x-axis shows the count of publications in each of these sources.



The bar plot above displays the top 20 conferences or journals by the number of publications in the dataset. This visualization helps to identify which sources are most frequently publishing research in the area of software vulnerability. The 'Source Title' on the y-axis represents different conferences and journals, and the x-axis shows the count of publications in each of these sources.

TOPIC MODELING

We can perform Latent Dirichlet Allocation (LDA) on the Abstracts to identify different categories or topics within the papers. LDA is a type of statistical model that classifies text in a document to a particular topic. It assumes documents are produced from a mixture of topics, and those topics then generate words based on their probability distribution.

For this analysis, we'll follow these steps:

Preprocess the Data: Clean the abstracts by removing punctuation, lowercasing, and possibly removing stopwords.
Text Vectorization: Convert the text data into a format that can be analyzed (like a document-term matrix).
LDA Model Fitting: Apply the LDA model to identify topics.
Interpretation: Analyze the output to understand the main topics.

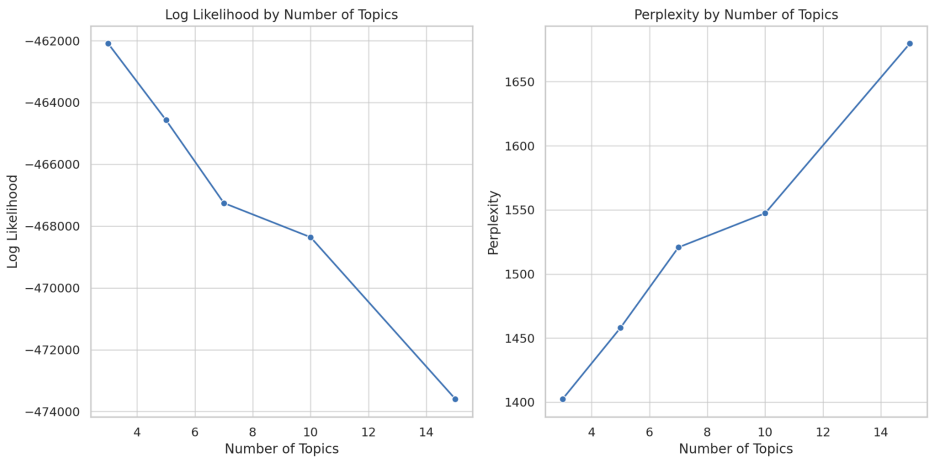
The choice of the number of topics in an LDA model is often subjective and can depend on the specific use case and dataset. However, there are methods to estimate the most appropriate number of topics for a given dataset. One common approach is to evaluate the model's performance for different numbers of topics and choose the number that gives the best balance between model complexity and explanatory power.

A popular metric for this is the Log-Likelihood and the Perplexity of the model. Generally, higher log-likelihood and lower perplexity values indicate better performance. However, it's also important to ensure that the topics are interpretable and meaningful.

Let's run the LDA model with different numbers of topics and evaluate their log-likelihood and perplexity to determine a suitable number of topics for this dataset.
The evaluation of the LDA model with different numbers of topics yielded the following results:

Num of Topics	Log Likelihood	Perplexity
3	-462095.13	1402.55
4	-463789.67	1440.32
5	-464571.08	1458.08
6	-465504.43	1479.58
7	-467254.10	1520.73
8	-467301.44	1521.86
10	-468360.62	1547.35
15	-473592.38	1679.64

- Log Likelihood still tends to decrease as the number of topics increases, with the highest for 3 topics.
- Perplexity follows a similar pattern, being lowest for 3 topics and increasing with more topics.



Here are the plots displaying the Log Likelihood and Perplexity for different numbers of topics in the LDA model:

Log Likelihood by Number of Topics:

- This plot shows how the log likelihood changes as the number of topics increases. A higher log likelihood generally indicates a better model fit.

Perplexity by Number of Topics:

- This plot shows the perplexity for different numbers of topics. Lower perplexity values typically indicate a better model.

Based on the Log Likelihood and Perplexity values you've provided, the two best topic numbers appear to be 3 and 4. These numbers have the highest log likelihood (which is better when higher) and the lowest perplexity (which is better when lower) among the options, suggesting they provide a good balance between model complexity and fit.

Let's perform LDA with these two topic numbers and observe the topics for each. This will give us a clearer idea of how the different numbers of topics categorize the documents. I will then provide the topics for both 3 and 4 topics.

Here are the topics identified by the LDA model for both 3 and 4 topics:

3 Topics

Topic 0:

- Key Words: software, vulnerability, vulnerabilities, prediction, code, security, metrics, models, using, data
- Interpretation: Focuses broadly on software vulnerabilities, their prediction, and related security measures.

Topic 1:

- Key Words: analysis, vulnerability, data, seismic, software, prediction, model, results, risk, paper
- Interpretation: Seems to relate to the analysis of vulnerabilities, potentially in a seismic context or involving risk assessment.

Topic 2:

- Key Words: software, vulnerability, security, vulnerabilities, prediction, code, model, models, learning, data
- Interpretation: Centers around software vulnerability and security, with an emphasis on prediction and learning models.

4 Topics

Topic 0:

- Key Words: software, vulnerabilities, vulnerability, prediction, code, metrics, models, security, using, vulnerable
- Interpretation: Similar to Topic 0 in 3 Topics, focusing on software vulnerabilities and predictive models.

Topic 1:

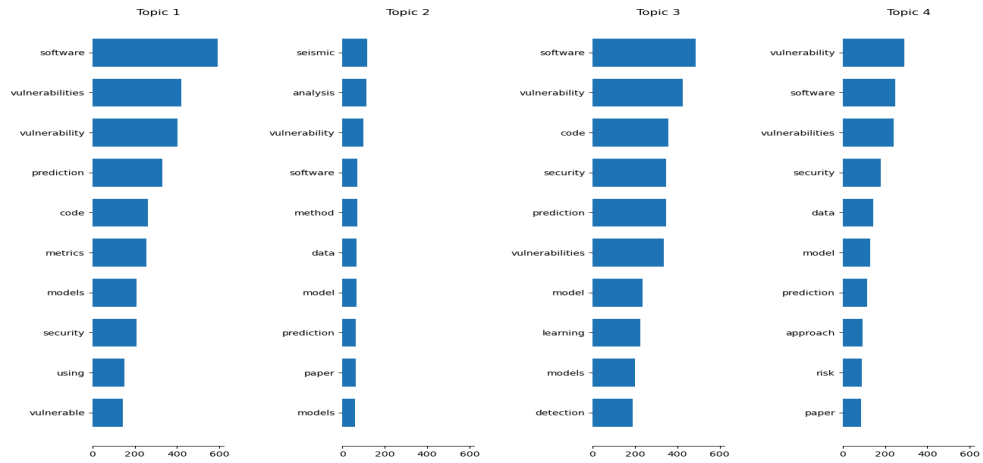
- Key Words: seismic, analysis, vulnerability, software, method, data, model, prediction, paper, models
- Interpretation: Similar to Topic 1 in 3 Topics, but with a clearer emphasis on seismic data and methods in software vulnerability.

Topic 2:

- Key Words: software, vulnerability, code, security, prediction, vulnerabilities, model, learning, models, detection
- Interpretation: Focuses on the technical aspects of software security, including code analysis, vulnerability detection, and learning models.

Topic 3:

- Key Words: vulnerability, software, vulnerabilities, security, data, model, prediction, approach, risk, paper
- Interpretation: Discusses software vulnerabilities and security, including risk assessment and prediction approaches.

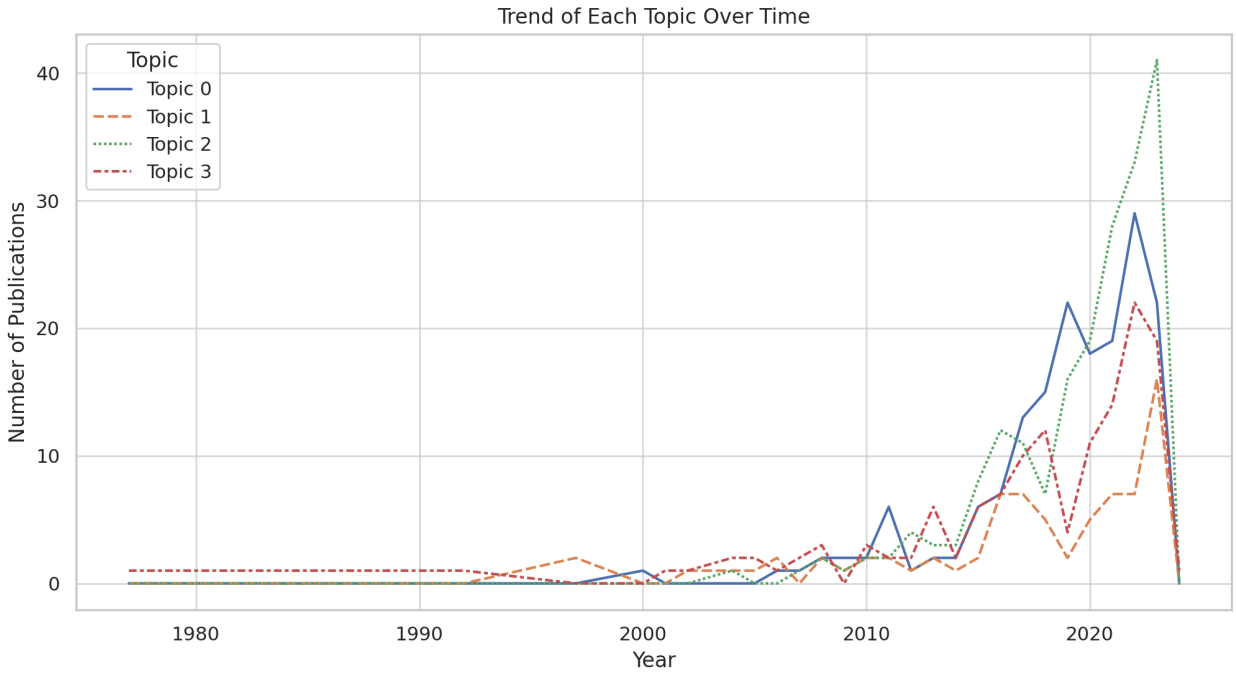


OverTime Analysis of Topics

To create a new column in the data DataFrame that assigns the most relevant topic to each record, we'll use the results from the LDA model. Each document (in our case, each abstract) gets a

distribution over the topics, and we'll assign the topic with the highest probability to each document.

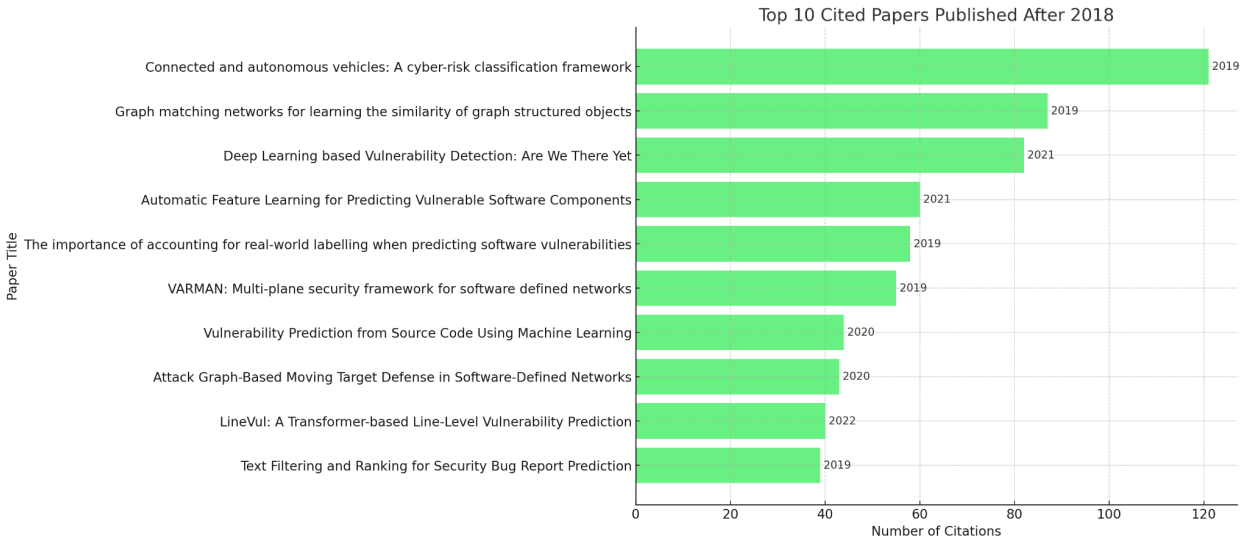
After assigning the topics, we'll create a plot showing the trend of each topic over time. This will require the `Year` column in the dataset to show how the prevalence of each topic has evolved over the years.



The line plot above shows the trend of each topic over time, based on the number of publications per year assigned to each topic. In the plot:

- Each line represents one of the four topics identified by the LDA model.
- The x-axis represents the year of publication.
- The y-axis shows the number of publications associated with each topic for each year.

This visualization helps in understanding how the prevalence or focus on each topic has evolved over the years within the dataset. For example, a rising trend in a particular topic might indicate increasing research interest or relevance in that area over time.



"Connected and autonomous vehicles: A cyber-risk classification framework" (2019, 121 citations):

- Focus: Addresses cybersecurity risks in the context of connected and autonomous vehicles (CAVs).
- Contribution: Proposes a comprehensive framework to classify cyber risks associated with CAVs, considering various attack vectors and potential impacts. This framework is designed to help stakeholders understand and mitigate cybersecurity threats in this rapidly advancing area of automotive technology.

"Graph matching networks for learning the similarity of graph structured objects" (2019, 87 citations):

- Focus: Investigates how to determine the similarity between objects that have a graph-structured format.
- Contribution: Introduces Graph Matching Networks (GMNs) as a novel approach for learning similarity measures in graph-structured data. This methodology is applicable in various domains, including social network analysis, bioinformatics, and recommendation systems.

"Deep Learning based Vulnerability Detection: Are We There Yet?" (2021, 82 citations):

- Focus: Evaluates the current status and effectiveness of deep learning techniques in detecting software vulnerabilities.
- Contribution: Offers a critical analysis of the advancements and limitations in applying deep learning for vulnerability detection. The paper discusses challenges such as data quality, model interpretability, and the generalization of results.

"Automatic Feature Learning for Predicting Vulnerable Software Components" (2021, 60 citations):

- Focus: On developing methods for automatically learning features that can predict software vulnerabilities.
- Contribution: Presents an approach that leverages machine learning to identify features within software components that are indicative of vulnerabilities. This assists in proactive vulnerability detection and enhances software security practices.

"The importance of accounting for real-world labelling when predicting software vulnerabilities" (2019, 58 citations):

- Focus: Critiques the gap between academic research and real-world applications in software vulnerability prediction.
- Contribution: Highlights the discrepancies between academic models and their practical utility, stressing the need for realistic labelling of data. This insight is vital for developing more accurate and applicable vulnerability prediction models.

"VARMAN: Multi-plane security framework for software-defined networking" (2019, 55 citations):

- Focus: Enhances security in software-defined networking (SDN).
- Contribution: Introduces VARMAN, a multi-layered security framework specifically designed for SDN environments. The paper discusses how this framework can provide comprehensive security coverage across different planes of an SDN architecture.

"Vulnerability Prediction from Source Code Using Deep Learning: A Systematic Literature Review" (2020, 44 citations):

- Focus: Reviews the use of deep learning for predicting vulnerabilities from source code.
- Contribution: Provides a systematic overview of existing literature, identifying the trends, methodologies, and gaps in current research. This review is instrumental for researchers aiming to understand the state-of-the-art and future directions in this area.

"Attack Graph-Based Moving Target Defense in Software Defined Networks" (2020, 43 citations):

- Focus: On developing moving target defense strategies in the context of software-defined networks.
- Contribution: Explores the use of attack graphs as a basis for implementing moving target defenses. This approach aims to dynamically alter the attack surface, making it more challenging for attackers to exploit vulnerabilities.

"LineVul: A Transformer-based Line-Level Vulnerability Detector" (2022, 40 citations):

- Focus: Aims at detecting vulnerabilities at the line level in source code.
- Contribution: Presents LineVul, which employs a transformer-based model to identify vulnerabilities in individual lines of code. This granular approach enhances the precision of vulnerability detection.

"Text Filtering and Ranking for Security Bug Report Prediction" (2019, 39 citations):

- Focus: On improving the prediction and identification of security-related issues in bug reports.
- Contribution: Develops methods for effectively filtering and ranking text within bug reports. This assists in more accurately predicting and managing security bugs, thereby improving overall software reliability and security.

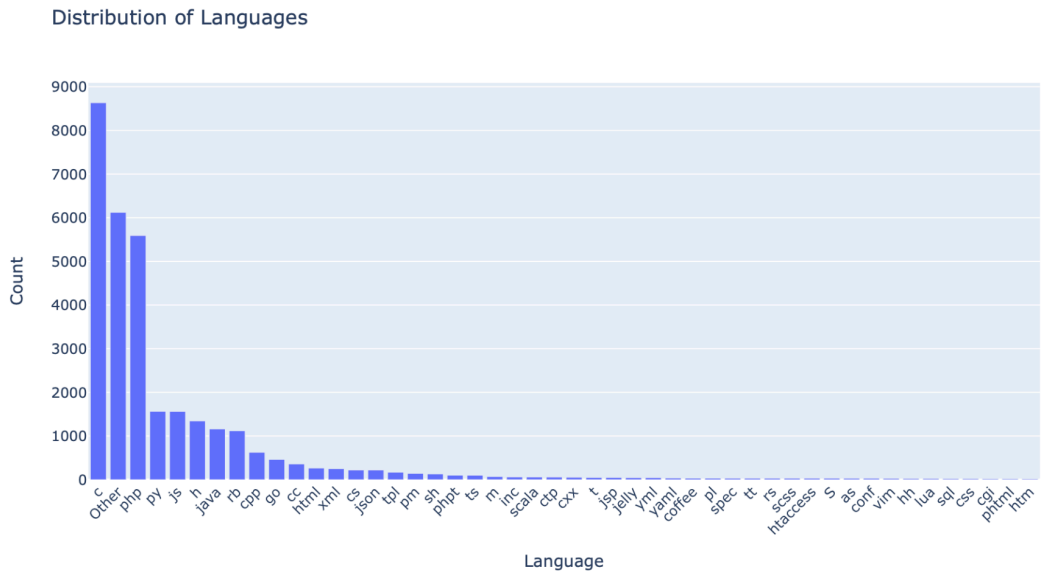
Each of these papers represents a significant contribution to the field of software security, offering innovative approaches, critical analyses, or comprehensive reviews to address various challenges and opportunities in this evolving domain.

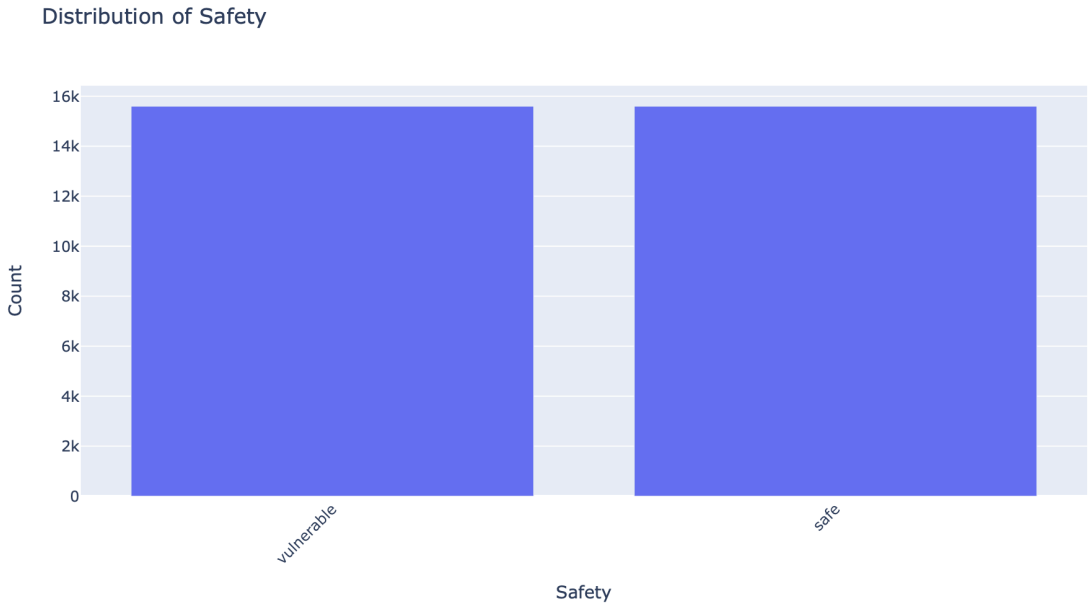
Part2.CVEfixesPreprocessed

Let's perform some exploratory data analysis (EDA) on the CVEfixes dataset to gain insights and provide you with some code for new ideas. The dataset appears to have three columns: 'code', 'language', and 'safety'. Here are some EDA tasks we can perform:

Basic Summary Statistics: Get an overview of the dataset, including summary statistics for numerical columns (if any).
Data Distribution: Explore the distribution of data in each column, including the frequency of unique values in categorical columns.
Missing Values: Check for missing values and handle them if necessary.
Data Visualization: Create visualizations such as histograms, bar plots, or pie charts to better understand the data.

Feature Relationships: Explore relationships between columns and identify any patterns or correlations.





LSTM Model for Code-Language Classification

Introduction:

This document presents a Long Short-Term Memory (LSTM) model designed for code-language classification. The objective is to classify code snippets based on their programming language. The model combines both code and language features to improve classification accuracy.

Model Architecture:

The LSTM model architecture consists of several key components:

1. **Embedding Layer:**
    - The embedding layer is responsible for converting input sequences into dense vectors. In this model, it maps words to 128-dimensional vectors.
2. **Spatial Dropout:**
    - SpatialDropout1D is employed to drop entire 1D feature maps, preventing overfitting by introducing noise in the embeddings.
3. **Bidirectional LSTM Layers:**
    - Two Bidirectional LSTM layers are stacked to capture sequential dependencies from both directions. The first LSTM layer returns sequences to the next layer.
4. **Dense Layers:**
    - Dense layers with ReLU activation functions are used to introduce non-linearity and reduce dimensionality.
5. **Dropout:**
    - Dropout layers are included to further prevent overfitting by randomly dropping connections during training.
6. **Output Layer:**
    - The output layer uses the sigmoid activation function to produce binary classification results.

Training and Results:

The model is trained on a dataset consisting of code snippets and their corresponding programming languages. Key training details include:

- **Optimizer:** Adam with a learning rate of 0.001
- **Loss Function:** Binary Crossentropy
- **Batch Size:** 64
- **Number of Epochs:** 20

Results:

The model's performance is evaluated on a validation set, and the results for each epoch are displayed. Key metrics include accuracy and loss. The final accuracy on the validation set is approximately 48.72%, indicating moderate performance. It is important to note that the performance may vary based on the dataset characteristics.

Recommendations for Improvement:

1. **Hyperparameter Tuning:**
   - Further experimentation with hyperparameters, such as learning rate, batch size, and layer configurations, may lead to better performance.
2. **Data Augmentation:**
   - Augmenting the dataset or introducing variations in code snippets may improve the model's ability to generalize.
3. **Additional Preprocessing:**
   - Exploring different text preprocessing techniques and tokenization strategies may enhance the model's understanding of code snippets.

Conclusion:

The LSTM model presented in this document serves as a baseline for code-language classification. Further refinements and experimentation are encouraged to achieve higher accuracy. Understanding the limitations and nuances of the dataset is crucial for optimizing model performance.

Part3.

The `Global_Dataset.xlsx` file has been successfully loaded. The dataset contains the following columns:

1. `ID`: A unique identifier for each entry.
2. `CVE-ID`: The Common Vulnerabilities and Exposures (CVE) identifier.
3. `CVSS-V3`: The Common Vulnerability Scoring System version 3 score.
4. `CVSS-V2`: The Common Vulnerability Scoring System version 2 score.
5. `SEVERITY`: The severity of the vulnerability.
6. `DESCRIPTION`: A description of the vulnerability.
7. `CWE-ID`: The Common Weakness Enumeration identifier.

For the Exploratory Data Analysis (EDA), we can perform several tasks to understand the dataset better:

1. **Basic Statistics**: Get an overview of numerical data like CVSS scores.
2. **Data Distribution**: Analyze the distribution of severity, CVSS scores, and the number of vulnerabilities over time.
3. **Missing Values**: Check for and handle any missing values in the dataset.
4. **Text Analysis**: Explore the `DESCRIPTION` field to understand common terms and phrases.

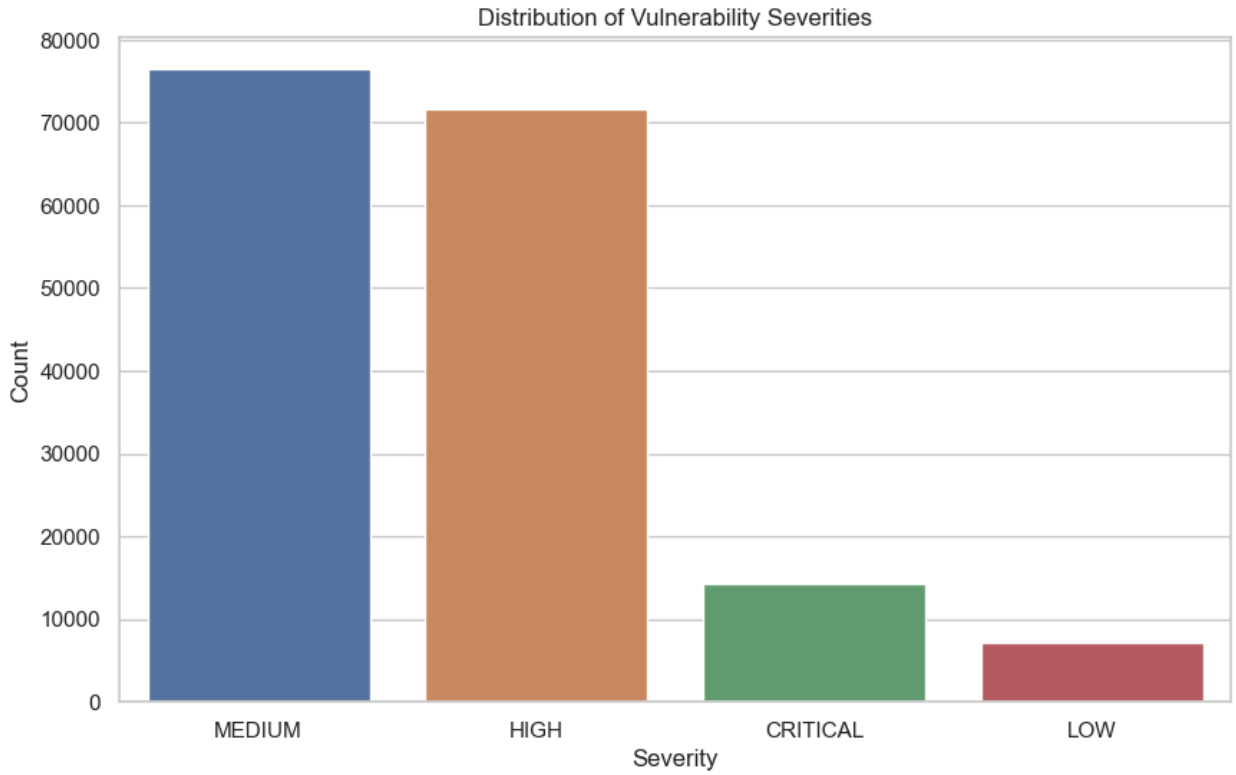
Let's start with basic statistics and data distribution. We'll check for missing values and get an overview of the numerical columns. Then, we'll look at the distribution of vulnerabilities by severity and over time.

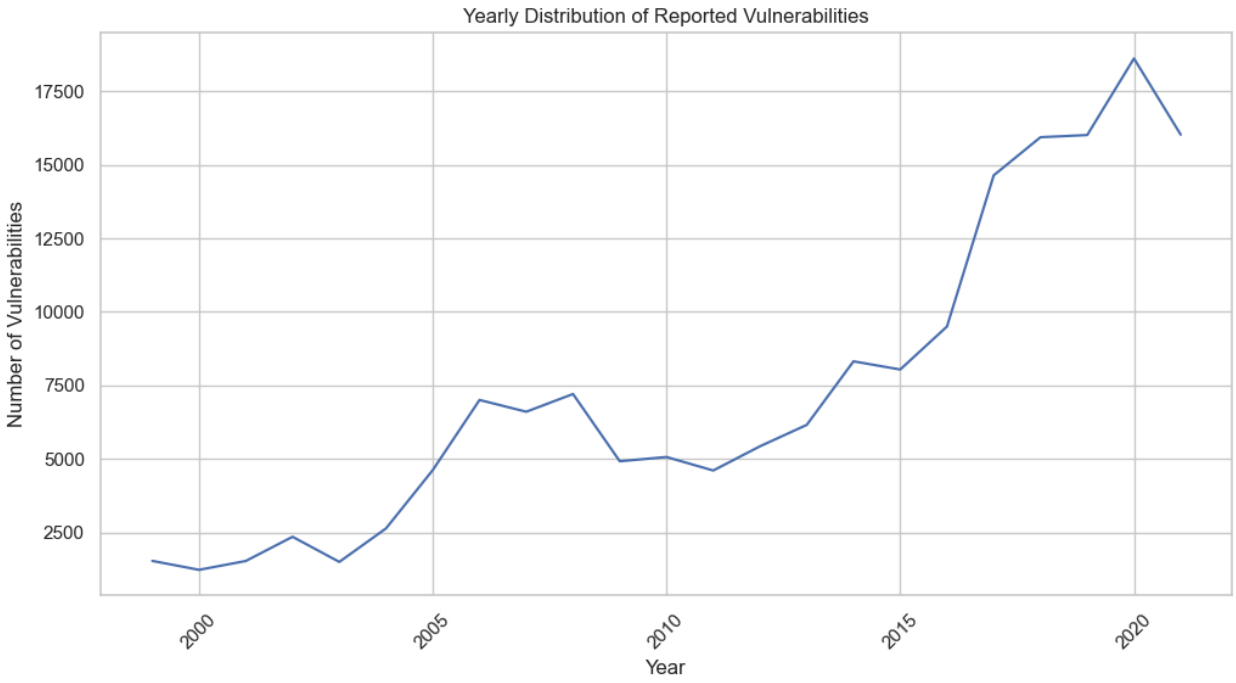
Here's an overview of the Exploratory Data Analysis (EDA) for the `Global_Dataset.xlsx`:

1. **Basic Statistics:**

- The dataset contains 169,566 entries.
- The ID column, which is a unique identifier, ranges from 1 to 169,566.

2. **Missing Values:**
   - There are no missing values in the dataset across all the columns.
3. **Severity Distribution:**
   - The most common severity rating is 'MEDIUM' (76,460 instances), followed by 'HIGH' (71,617 instances).
   - There are 14,198 instances of 'CRITICAL' severity and 7,205 of 'LOW' severity.
   - 86 entries have no severity rating.
4. **Yearly Distribution of Vulnerabilities (for the first 10 years):**
   - Starting from 1999, the number of reported vulnerabilities has generally increased year over year.
   - The dataset includes vulnerabilities reported from 1999 to 2021, with a noticeable increase in entries from 2005 onwards.

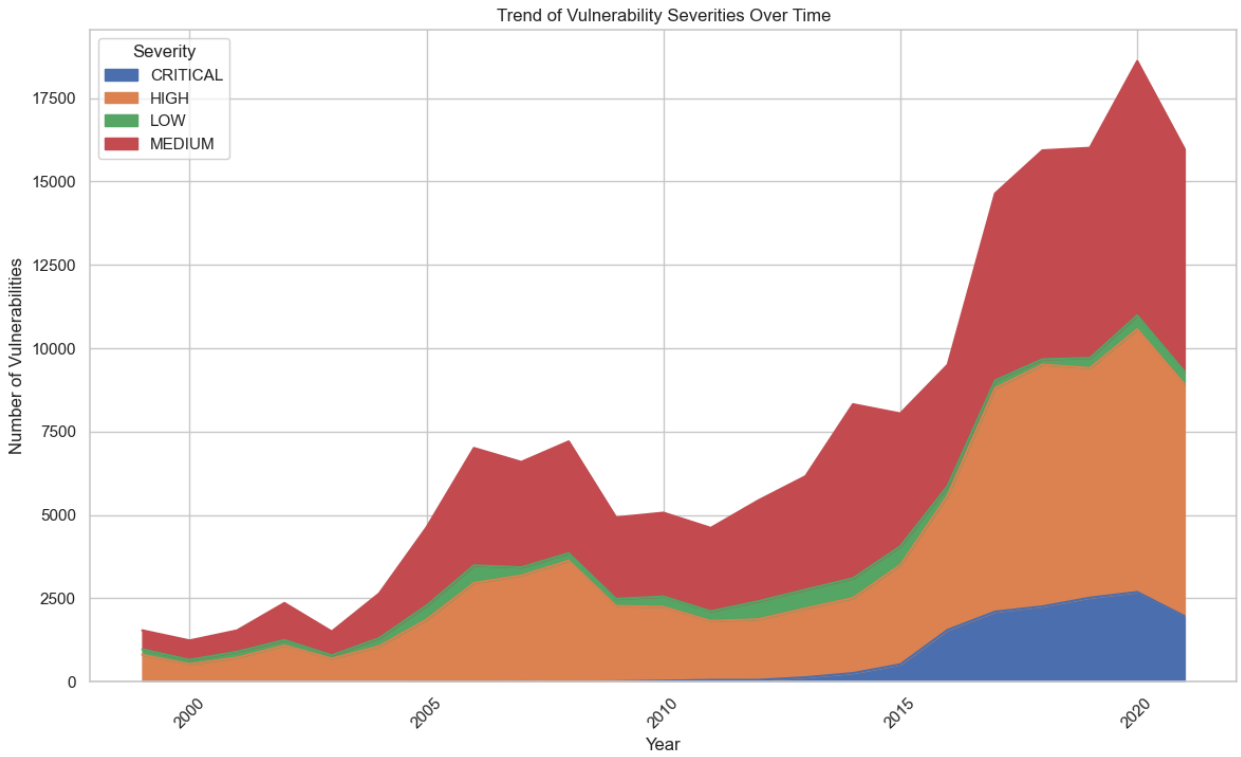




Here are two plots that provide visual insights into the dataset:

1. **Distribution of Vulnerability Severities:**
    - This plot shows the count of vulnerabilities for each severity level. The most common severities are 'MEDIUM' and 'HIGH', with 'CRITICAL' and 'LOW' being less frequent.
2. **Yearly Distribution of Reported Vulnerabilities:**
    - This line plot illustrates the number of vulnerabilities reported each year. There is a clear trend of increasing vulnerability reports over time, particularly from the mid-2000s onward.

These plots help in visually analyzing the distribution of data in terms of severity and the trend over time, offering a quick way to understand the characteristics of the vulnerabilities reported.



The area plot above illustrates the trend of vulnerability severities over time, spanning from 1999 to 2021. This visualization provides insights into how the different severity levels of reported vulnerabilities have changed annually. Key observations include:

1. A general increase in the total number of vulnerabilities reported each year, especially noticeable from the mid-2000s.
2. Variations in the proportion of severity levels over time, with 'MEDIUM' and 'HIGH' severities being the most common.
3. The presence of 'CRITICAL' and 'LOW' severities, although in smaller quantities compared to 'MEDIUM' and 'HIGH'.

This trend analysis helps in understanding the evolving nature of cybersecurity threats and the importance of continuous monitoring and updating of security measures.

Document: Vulnerability Severity Prediction Using Machine Learning

Introduction

This document outlines the development of a machine learning model to predict the severity of software vulnerabilities based on their descriptions. The model utilizes a dataset containing details of vulnerabilities, including their Common Vulnerabilities and Exposures (CVE) IDs, descriptions, and severity levels.

Dataset Overview

- **Source**: SD_Vulnerability_Dataset.xlsx
- **Contents**: CVE-ID, CVSS scores (V2 and V3), severity, descriptions, CWE-IDs, and CWE names.
- **Sample Size**: 5,000 records (sampled from the full dataset for efficient processing).

Preprocessing and Feature Engineering

1. **Label Encoding**: Severity labels ('CRITICAL', 'HIGH', 'MEDIUM', 'LOW') were numerically encoded for model compatibility.
2. **Text Vectorization**: The 'DESCRIPTION' field, containing textual data, was transformed into a numerical format using the TF-IDF (Term Frequency-Inverse Document Frequency) technique.

Model Development

- **Algorithm**: Random Forest Classifier.
- **Rationale**: Chosen for its robustness and effectiveness with varied data types.
- **Training and Testing**: The dataset was split into a 70-30 ratio for training and testing, respectively.

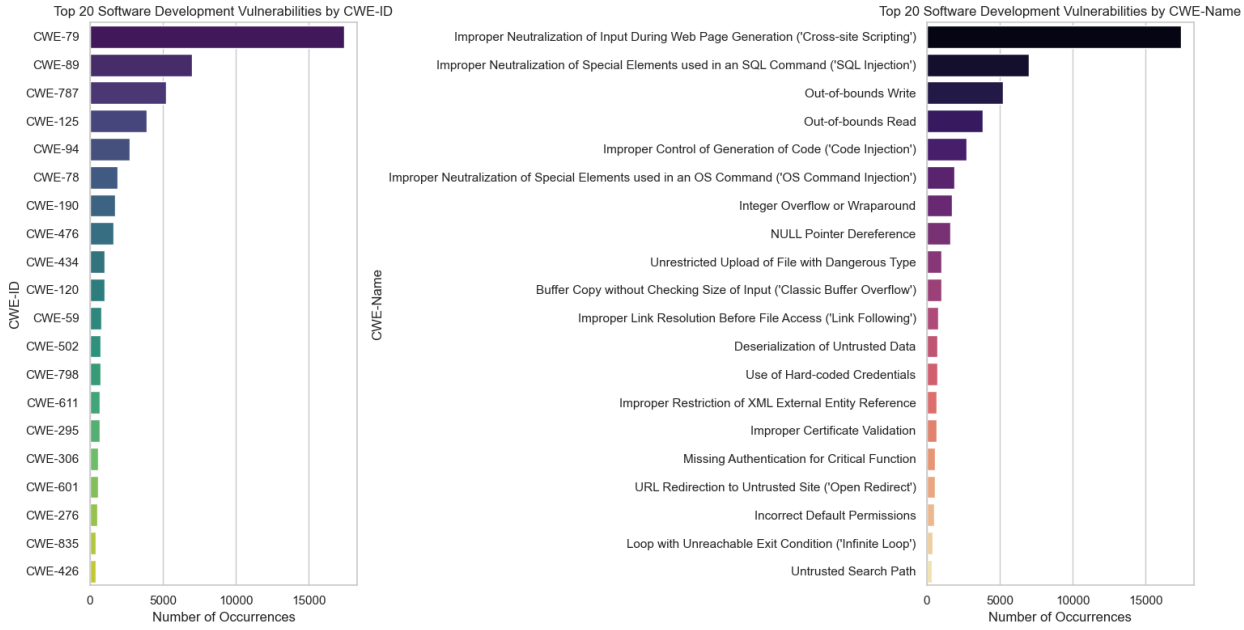
Model Evaluation

- **Evaluation Metrics**: Precision, recall, and F1-score for each severity class; overall accuracy.
- **Performance**: Varied performance across different severity classes, with the model showing higher accuracy in predicting 'LOW' and 'CRITICAL' severities.

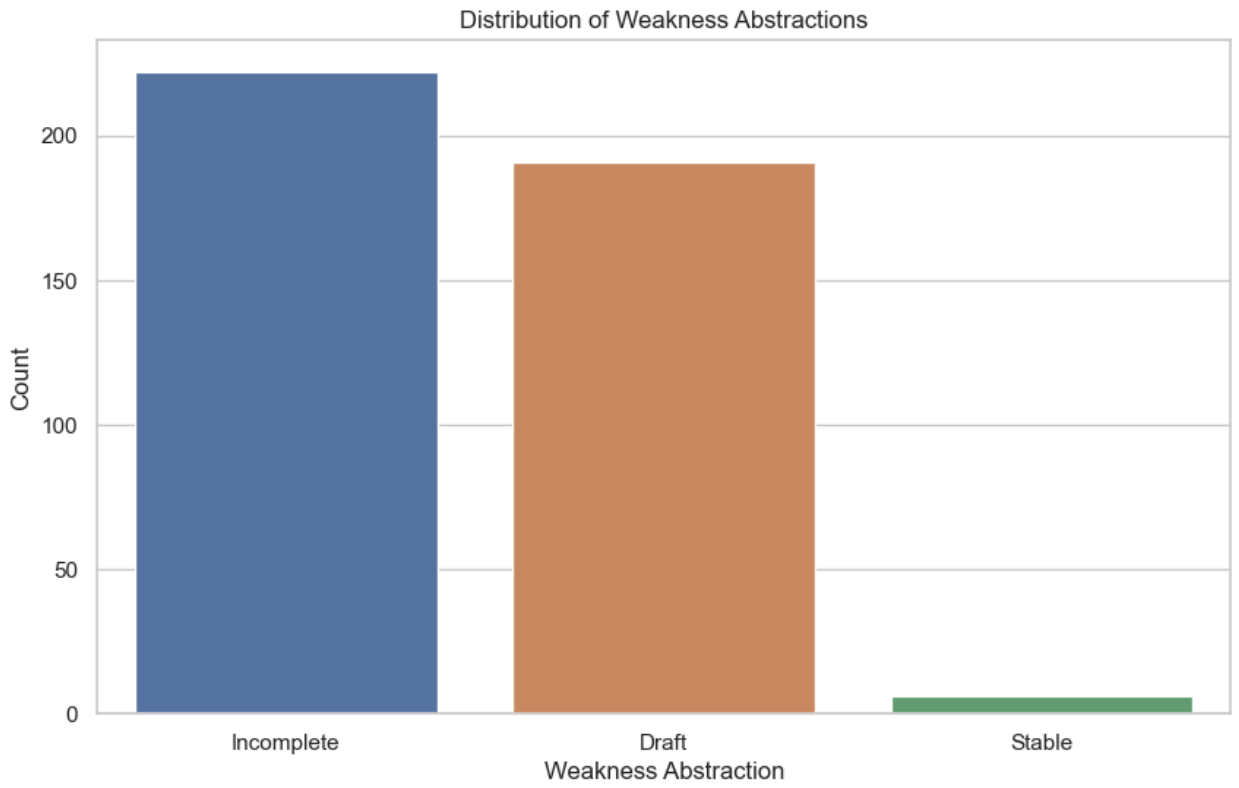
Conclusions and Recommendations

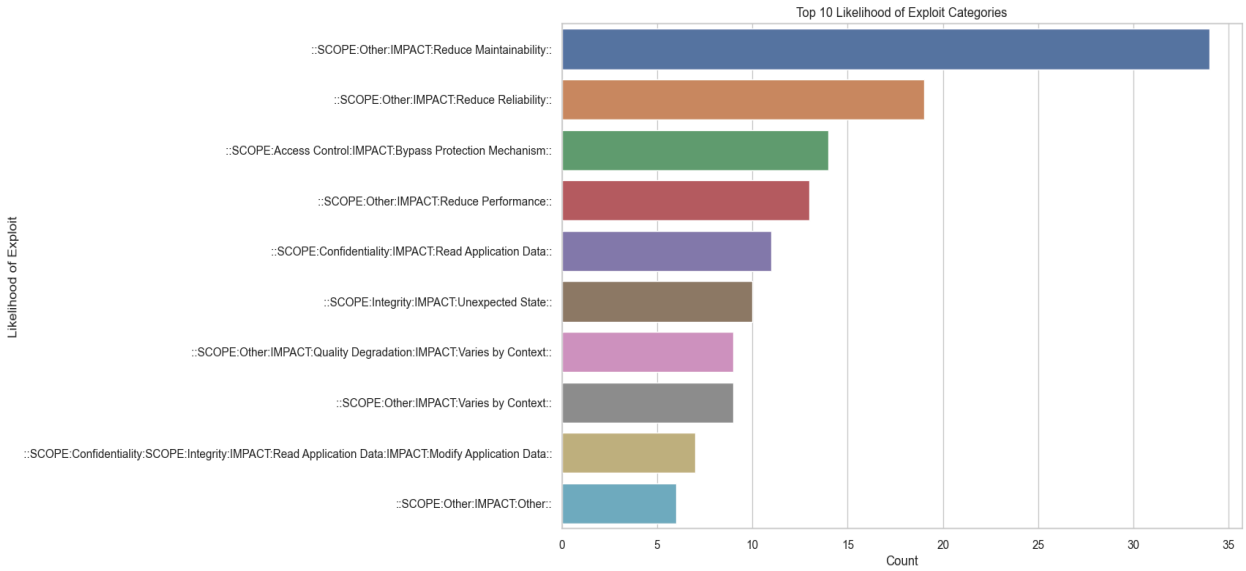
The model demonstrates a reasonable degree of accuracy in predicting the severity of vulnerabilities based on descriptions. However, there is scope for improvement, particularly in balancing performance across different severity classes. Future work may include:

- **Experimenting with Different Classifiers**: Exploring other algorithms like SVM, Naive Bayes, or neural networks.
- **Hyperparameter Tuning**: Optimizing the current model for better performance.
- **Feature Enhancement**: Incorporating additional features like CVSS scores and CWE-IDs.



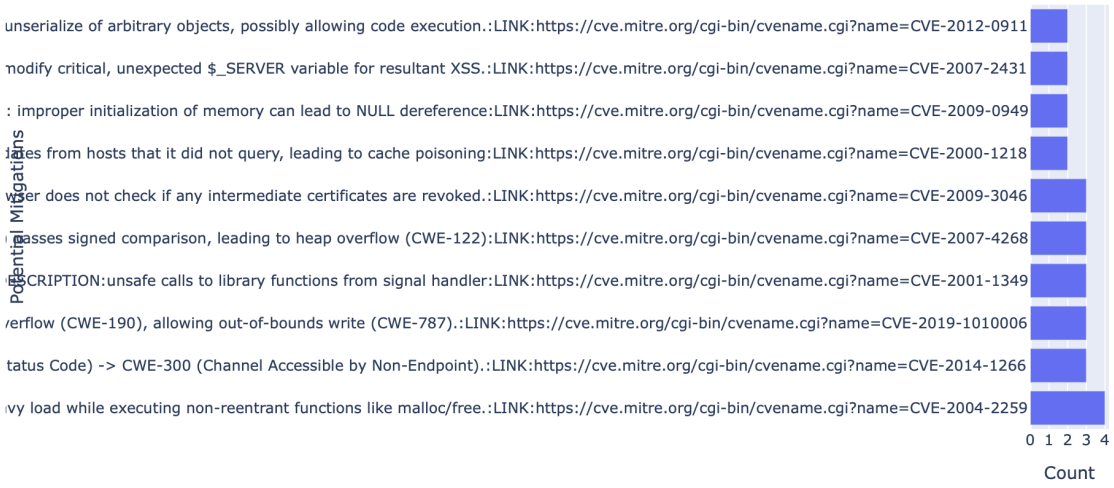
Exploratory Data Analysis of SD_CWEs.csv Dataset - Summary of Results with Additional Insights

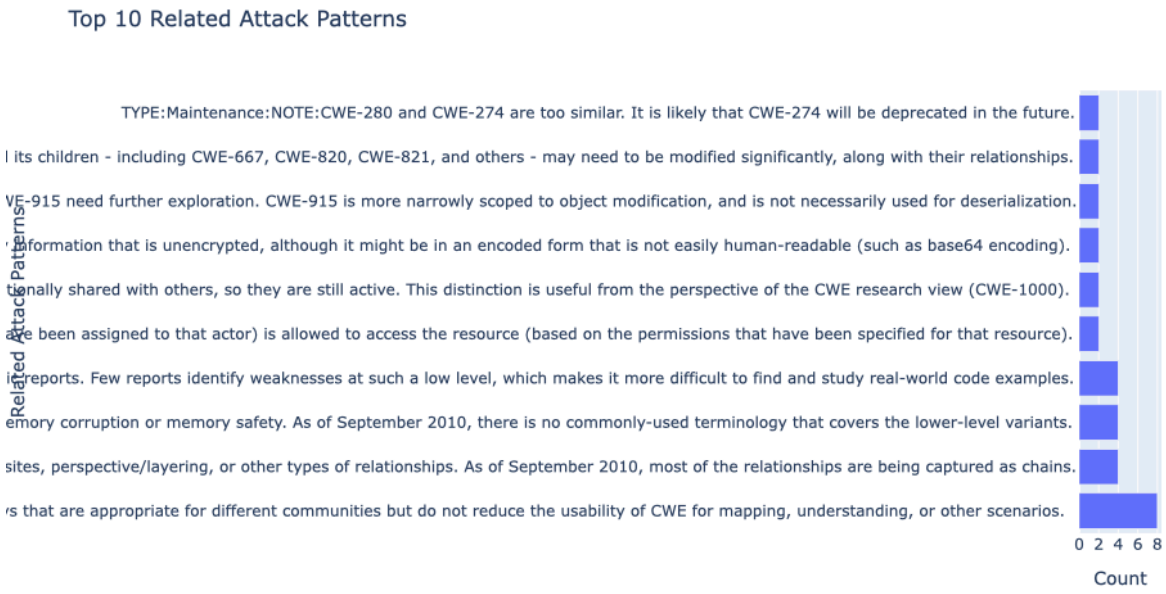

Distribution of Weakness Abstractions
Count
200
150
100
50
0
Incomplete
Draft
Stable
Weakness Abstraction


Top 10 Likelihood of Exploit Categories
Likelihood of Exploit
::SCOPE:Other:IMPACT:Reduce Maintainability::
::SCOPE:Other:IMPACT:Reduce Reliability::
::SCOPE:Access Control:IMPACT:Bypass Protection Mechanism::
::SCOPE:Other:IMPACT:Reduce Performance::
::SCOPE:Confidentiality:IMPACT:Read Application Data::
::SCOPE:Integrity:IMPACT:Unexpected State::
::SCOPE:Other:IMPACT:Quality Degradation:IMPACT:Varies by Context::
::SCOPE:Other:IMPACT:Varies by Context::
::SCOPE:Confidentiality:SCOPE:Integrity:IMPACT:Read Application Data:IMPACT:Modify Application Data::
::SCOPE:Other:IMPACT:Other::
0
5
10
15
20
25
30
35
Count

Top 10 Affected Resources

Top 10 Potential Mitigations





Overview

This extended document encapsulates the results of the Exploratory Data Analysis (EDA) performed on the SD_CWEs.csv dataset. It includes insights into Potential Mitigations, Affected Resources, Related Attack Patterns, the 'Likelihood of Exploit,' and the Distribution of Weakness Abstractions.

Dataset

The SD_CWEs.csv dataset is an extensive collection of data on software weaknesses, detailing potential mitigations, affected resources, related attack patterns, likelihood of exploits, and weakness abstractions.

EDA Results

1. **Potential Mitigations:**
    - Identified the top 10 most frequently cited mitigation strategies.
    - These strategies highlight prevalent approaches to countering software vulnerabilities.
2. **Affected Resources:**
    - Revealed the top 10 resources most commonly impacted by weaknesses.
    - This analysis aids in pinpointing critical areas in software systems that require enhanced security.
3. **Related Attack Patterns:**
    - Showcased the top 10 common attack patterns associated with the weaknesses.
    - Understanding these patterns is vital for developing targeted defense strategies.
4. **Likelihood of Exploit:**
    - A simplified plot was generated to display the top 10 'Likelihood of Exploit' categories.

- This visualization provides a quick understanding of how likely different weaknesses are to be exploited.

5. **Distribution of Weakness Abstractions:**
    - Analyzed the distribution of different types of weakness abstractions.
    - This distribution offers insights into the common nature of documented software weaknesses.

Visualizations

- Horizontal bar plots were created to visually represent the top 10 categories in Potential Mitigations, Affected Resources, Related Attack Patterns, and 'Likelihood of Exploit'.
- An additional bar plot was used to illustrate the distribution of Weakness Abstractions.

Conclusions

The comprehensive EDA provides a multi-faceted view of the software vulnerabilities landscape. The identified trends in mitigations, resources at risk, related attack patterns, likelihood of exploits, and weakness abstractions are instrumental in guiding security practices and policies in software development and cybersecurity.

Resources :

https://www.kaggle.com/datasets/girish17019/cvefixes-vulnerable-and-fixed-code

https://www.kaggle.com/datasets/krooz0/cve-and-cwe-mapping-dataset