**SQL Injection Detection Using Machine Learning**

**Submitted for**

**Statistical Machine Learning CSET211**

Submitted by:

**(E23CSEU1146) Ayush Upadhyay**

**(E23CSEU1148) Riti Dubey**

# **(E23CSEU1167) Shubham Pandey**

Submitted to

**DR. Amit Soni Arya**

**July-Dec 2024**

**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**



**INDEX**

|  |  |  |
| --- | --- | --- |
| Sr.No | Content | Page No |
| 1 | Abstract |  |
| 2 | Introduction |  |
| 3 | Related Work |  |
| 4 | Methodology |  |
| 5 | Software Required |  |
| 6 | Experimental Results |  |
| 7 | Conclusions |  |
| 8 | Code |  |
| 9 | Link |  |

# Abstract

**This project focuses on the detection of SQL Injection (SQLI) attacks using machine learning and deep learning techniques. SQLI is a cyber attack method where malicious users exploit vulnerabilities in web applications to manipulate or gain unauthorized access to databases. The goal of this project is to identify malicious SQL queries using machine learning algorithms and pretrained deep learning models like BERT. The approach improves the security of web applications by detecting malicious queries before they can be executed.**

# Introduction

**SQL Injection (SQLI) is a widespread security vulnerability that occurs when an attacker injects malicious SQL queries into input fields to manipulate a web application's database. These malicious queries may bypass authentication mechanisms, extract sensitive data, or even delete entire databases.**

**This project aims to leverage machine learning (ML) and deep learning (DL) models to identify and classify SQL queries as either malicious or genuine. By training models on a labeled dataset of SQL queries, we can develop a system that automatically detects SQLI attacks and enhances web application security.**

**For example, an attacker might input a query like userid = 'or1=1 -- to bypass authentication mechanisms.** **Machine learning models can be used to classify such queries, distinguishing them from normal queries, to prevent potential threats.**

# Related Work

**Several approaches have been proposed for detecting SQLI attacks. Notable among them are:**

1. **Ensemble Machine Learning Approaches: Ensemble learning methods like GBM and AdaBoost combine multiple models to improve accuracy. GBM builds sequential decision trees, while AdaBoost focuses on misclassified samples. Though not used in this project, XGBoost, a gradient boosting variant, shares these principles and was applied for effective SQLI detection.**
2. **Detection of Web Attacks Using Ensemble Learning: This approach focuses on both SQLI and Cross-Site Scripting (XSS) attacks, using data preprocessing techniques like tokenization and stemming.**
3. **Malicious and Benign URLs: Although not directly related to SQLI, this approach detects malicious URLs using features like query length and special character counts.**
4. **Research on SQL Injection Detection Using SVM: This method uses Support Vector Machines (SVM) to classify HTTP requests containing SQLI attempts.**

# Methodology

The methodology for SQL Injection detection using machine learning involves the following steps:

## 3.1 Data Collection

The dataset used in this project is sourced from Kaggle and consists of SQL queries labeled as either **malicious (1)** or **genuine (0)**. This dataset includes both normal SQL queries and SQLI queries with various attack techniques.

## 3.2 Data Preprocessing

Data preprocessing is crucial to the success of the model. The following steps were performed:

* **Lowercasing**: All text data was converted to lowercase to standardize inputs.
* **Tokenization**: The SQL queries were tokenized to break them down into individual words and symbols.
* **Handling Special Characters**: Unlike conventional NLP tasks, special characters in SQL queries are important, so they were retained during preprocessing.

## 3.3 Feature Extraction

To effectively distinguish between malicious and genuine SQL queries, several features were extracted, including:

* **Special Characters**: The presence of symbols such as semicolons (;), dashes (--), and single quotes (') is often indicative of SQLI attempts.
* **Keywords**: Words like OR, AND, DROP, SELECT, UNION are common in SQLI attacks.
* **Punctuation**: Certain punctuation marks are often used in SQLI attacks.
* **Word Counts**: The length of the query can help differentiate between normal and malicious queries.

Additionally, techniques like **word2vec** and **TF-IDF** were used to transform the text into n-dimensional vectors for more sophisticated feature representation.

## 3.4 Exploratory Data Analysis (EDA)

EDA was conducted to understand the distribution of the dataset, check for class imbalance, and visualize feature importance. Visualizations like word clouds, count plots, and distribution plots were created to better understand the data.

## 3.5 Model Training

Several machine learning models were tested and trained using the data. These models include:

* **Logistic Regression**
* **Support Vector Machine (SVM)**
* **XGBoost**
* **Pretrained BERT Model**

## 3.6 Model Evaluation

Models were evaluated using the following metrics:

* **Precision**: The fraction of correct positive predictions.
* **Recall**: The fraction of actual positives correctly predicted.
* **F1-Score**: The harmonic mean of precision and recall.
* **ROC-AUC**: The area under the receiver operating characteristic curve.

Additionally, confusion matrices were used to understand the misclassification rates, and **TensorBoard** was employed to visualize training metrics for deep learning models like BERT.

## Software Required

• **Software: Python, Jupyter Notebook, Scikit-learn, Pandas, Numpy, TensorFlow, and other relevant libraries**.

# Experimental Results

The evaluation metrics indicate that the model performs well in distinguishing between malicious and genuine SQL queries. Feature extraction using word2vec preserved the semantic meaning between words, which helped improve model performance. The pretrained BERT model also showed promising results, with TensorBoard scalars providing insights into the training process.

## Conclusions

**This project demonstrates that machine learning techniques can effectively detect SQLI attacks. Feature extraction plays a crucial** **role in improving model performance. Among the models tested, XGBoost with hyperparameter tuning, Logistic Regression with hyperparameter tuning, and Linear SVM with hyperparameter tuning provided the best results. The use of pretrained BERT also showed high F1-scores, highlighting its potential for real-world applications.**

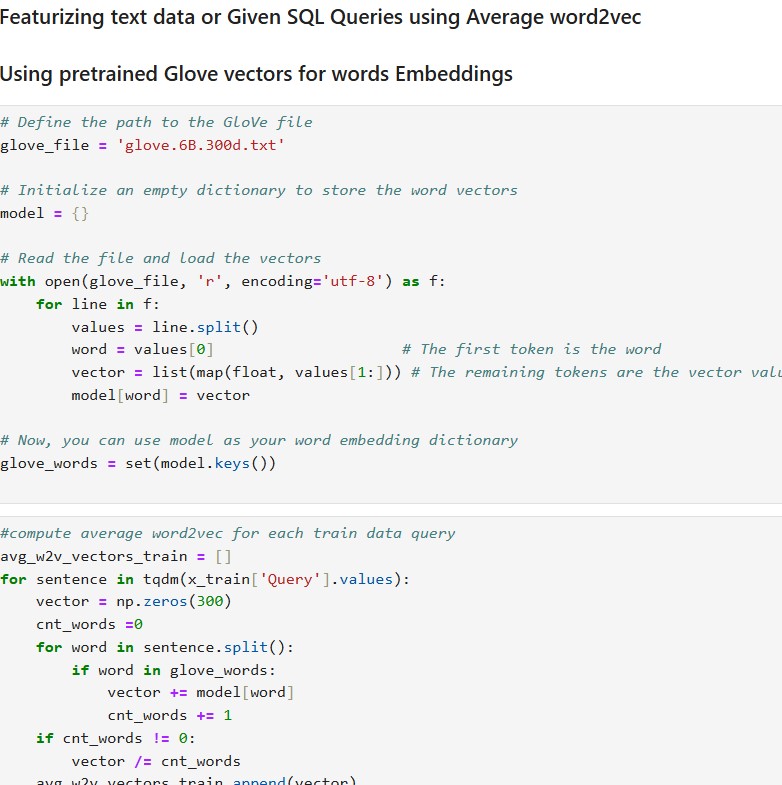
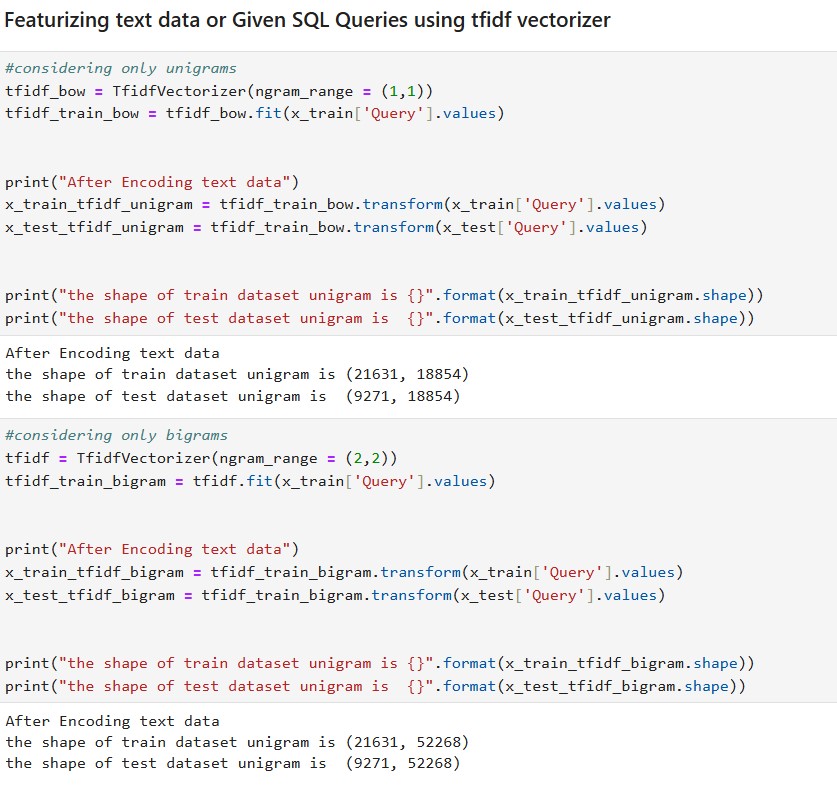
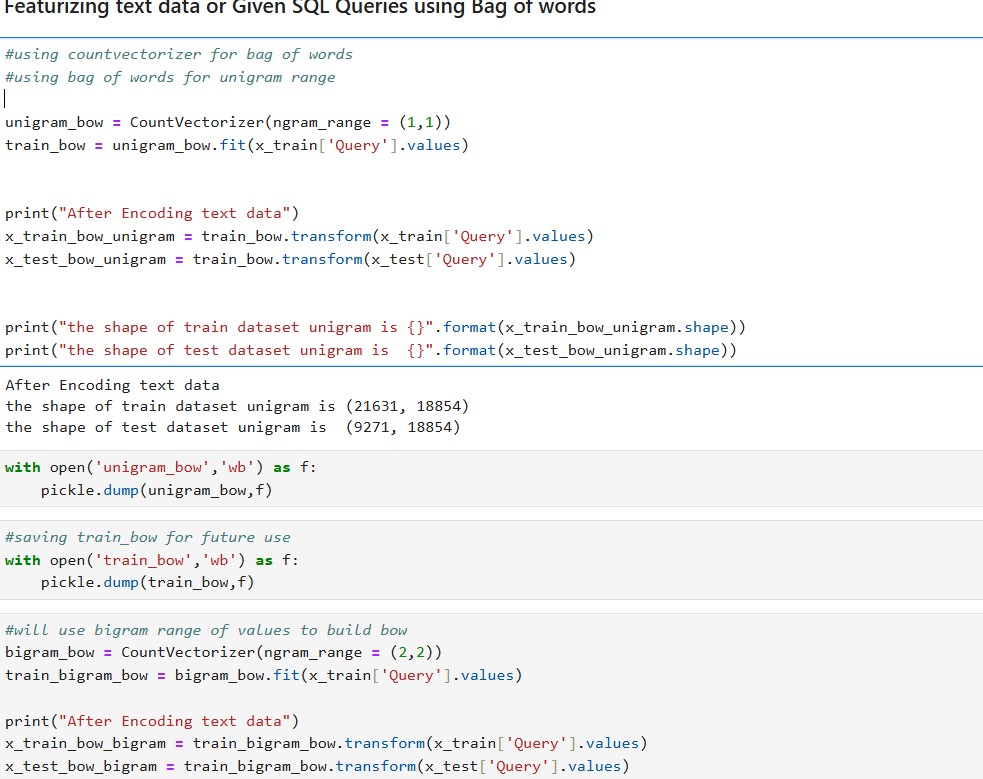
## Future Scope

**Future work can focus on:**

1. **Enhancing feature extraction techniques.**
2. **Exploring more advanced deep learning models for better accuracy.**
3. **Implementing real-time SQLI detection systems.**
4. **Expanding the dataset to include more diverse SQL queries.**

# Code

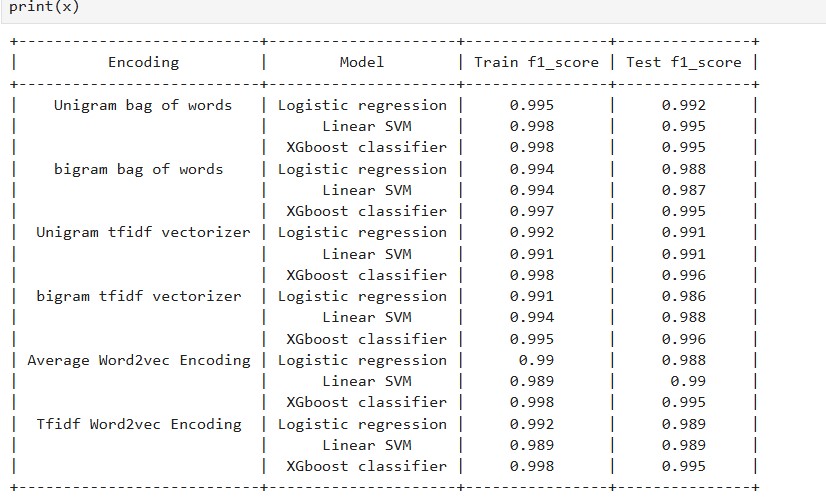
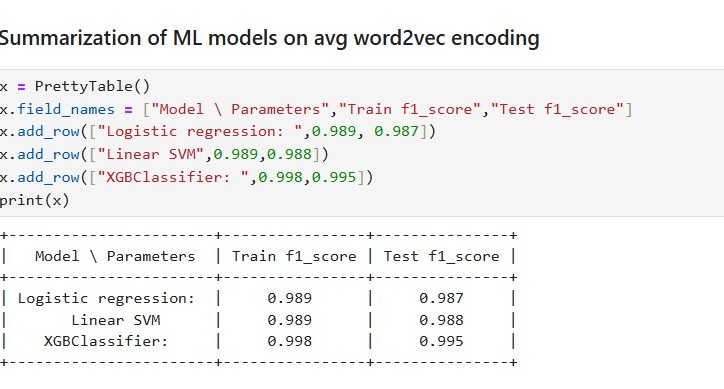
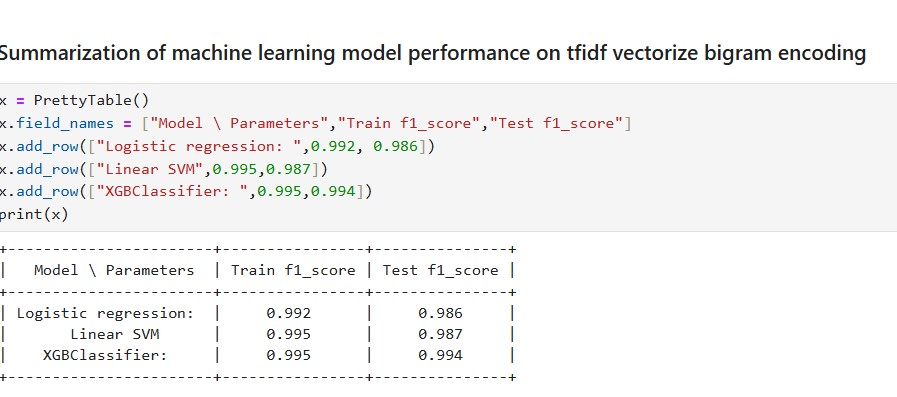
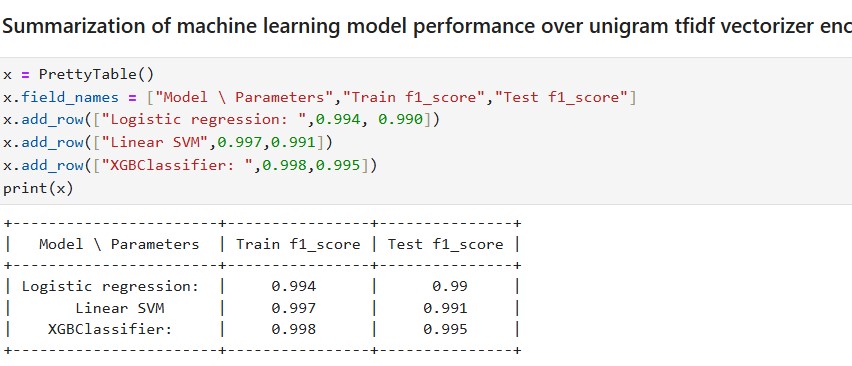
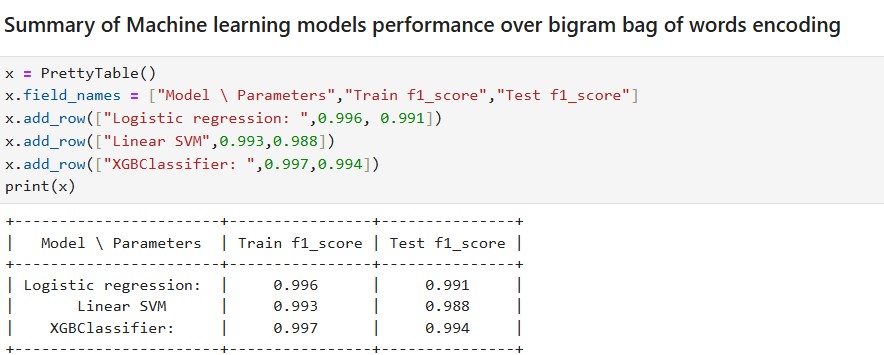
## Feature Extraction

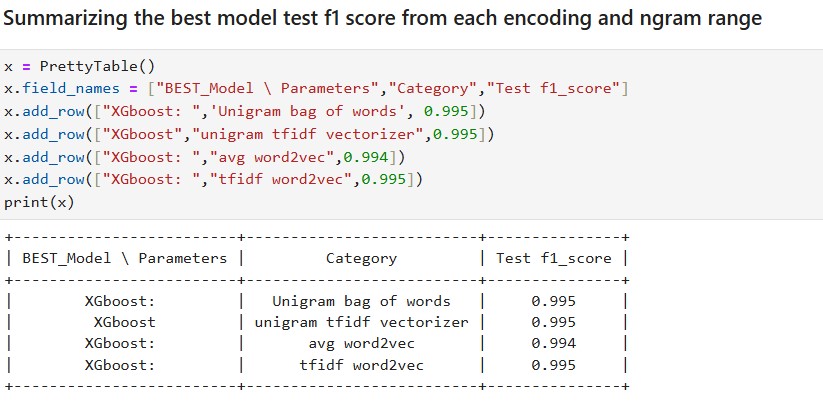


## Generalized Function for Logistic Regression and Linear SVM with Performance Metrics



## Summary of Machine learning models performance





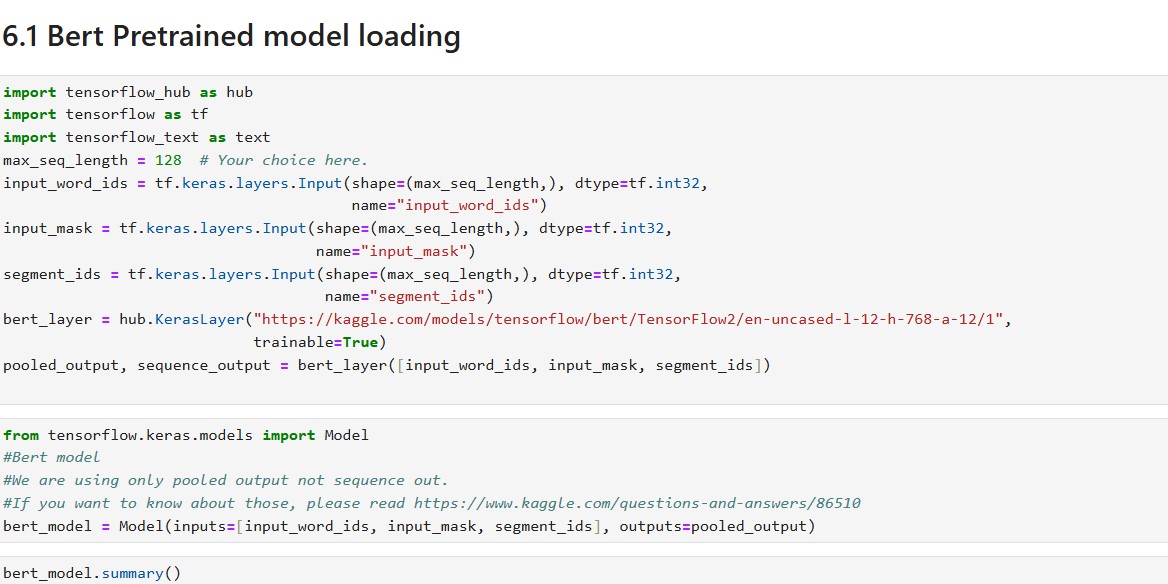
Conclusions :

**In each category XGboost classifier given the best f1-score**

**Bigrams provide less f1 score than unigrams we can see from the above table.**

**we got the test f1 score of 0.995 for three models. we can choose any one of them for classification.**[**¶**](http://localhost:8888/notebooks/SQL-INJECTION-MACHINE-LEARNING-MODEL.ipynb?#we-got-the-test-f1-score-of-0.995-for-three-models.-we-can-choose-any-one-of-them-for-classification.)

## Pretrained BERT Model



Observation :

**As we can see from the above training we got best f1-score of 0.9921 which is very good from ML models i.e 0.99**[¶](http://localhost:8888/notebooks/Deep_Learning_Model.ipynb?#As-we-can-see-from-the-above-training-we-got-best-f1-score-of-0.9921-which-is-very-good-from-ML-models-i.e-0.99)

## References

[**http://ieeexplore.ieee.org/document/6993127**](http://ieeexplore.ieee.org/document/6993127)

[**https://scholarworks.sjsu.edu/cgi/viewcontent.cgi?article=1649&contex t=etd\_projects**](https://scholarworks.sjsu.edu/cgi/viewcontent.cgi?article=1649&context=etd_projects)

[**https://www.researchgate.net/publication/377708236\_A\_Machine\_Lear ning\_Methodology\_for\_Detecting\_SQL\_Injection\_Attacks**](https://www.researchgate.net/publication/377708236_A_Machine_Learning_Methodology_for_Detecting_SQL_Injection_Attacks)

### GitHub Link

[Ritidube/SQL-INJECTION-DETECTION: This project uses machine learning to detect SQL injection attacks by classifying SQL queries as safe or malicious. SQL injection is a common method for unauthorized database access. The model aims to enhance security, protecting sensitive data and helping organizations prevent potential threats.](https://github.com/Ritidube/SQL-INJECTION-DETECTION)

POWERPOINT PRESENTATION LINK:

<https://www.canva.com/design/DAGWt7Fx9nE/KYRN_bokXG3scCTiP7bi7A/view?utm_content=DAGWt7Fx9nE&utm_campaign=designshare&utm_medium=link&utm_source=editor>

Submitted by:

**(E23CSEU1146) Ayush Upadhyay**

**(E23CSEU1148) Riti Dubey**

# **(E23CSEU1167) Shubham Pandey**