ANOMASHIELD : REAL-TIME DATA POISONING DETECTION IN DISTRIBUTED MACHINE LEARNING

Gourishetty Sindhusha
Assistant Professor
Dept. of Computer Science and Engineering
Jyothishmathi Institute of Technology and Science
(JNTUH)
Karimnagar, Telanagana, India
gourishettysindhusha@gmail.com

Ravinder Mogili
Professor and Head
Dept. of Computer Science and Engineering
Jyothishmathi Institute of Technology and Science
(JNTUH)
Karimnagar, Telanagana, India
mogili.ravinder@jits.ac.in

Srija Thota
UG Student
Dept. of Computer Science and Engineering
Jyothishmathi Institute of Technology and Science
(JNTUH)
Karimnagar, Telanagana, India
thotasrj2004@gmail.com

Sai Kumar Jangiti
UG Student
Dept. of Computer Science and Engineering
Jyothishmathi Institute of Technology and Science
(JNTUH)
Karimnagar, Telanagana, India
22271a05a0saimudhiraj@gmail.com

Vishnu Kandikatala
UG Student
Dept. of Computer Science and Engineering
Jyothishmathi Institute of Technology and Science
(JNTUH)
Karimnagar, Telanagana, India
22271a05c3vishnu@gmail.com

Manaswini Katakam
UG Student
Dept. of Computer Science and Engineering
Jyothishmathi Institute of Technology and Science
(JNTUH)
Karimnagar, Telanagana, India
22.579manaswini@gmail.com

***Abstract*— Data poisoning attacks have a significant effect on the accuracy and reliability of machine learning models. Data poisoning is a type of attack where malicious data is injected into the model. In this paper, a new tool named AnomaShield is proposed. It is a web-based tool that detects anomalous rows in datasets. AnomaShield makes use of four approaches: Z-Score, IQR, Isolation Forest, and One-Class SVM, to detect anomalous rows. Based on majority votes, it determines which rows are anomalous. AnomaShield is built using Python and Django, and Chart.js and Bootstrap are used at the frontend. The experimental results show that the tool was able to classify the data with an accuracy of 92%, effectively detecting the anomalies in the data injected during the experiment. The tool was efficient in detecting the anomalies using the integration method.**

Keywords—Data Poisoning, Anomaly Detection, Ensemble Methods, One Class SVM, Isolation Forest, Web based systems, Data Quality Assurance

I. INTRODUCTION

People use Machine learning systems a lot in areas, like finance, healthcare and education. The success of Machine learning systems relies on the quality of the training data. It matters. I have seen models produce predictions when the training data is bad. If malicious or extreme records get into datasets models can make predictions, unstable outputs or unsafe decisions [1].

The scripts used for data cleaning, which are used for inspection, are not effective in dealing with data poisoning attacks. As the size of the data increases, the need for data validation mechanisms increases [3].

To solve these problems using AnomaShield which is a web-based application designed to identify potentially poisoned or unusual rows in tabular datasets before training machine learning models. It combines multiple anomaly detection methods and presents the results through a clear, user-friendly interface. The system focuses on tabular datasets with numeric features and supports users with limited expertise in statistics or machine learning, even when the number or nature of poisoned rows is unknown.

Beyond detecting anomalies, AnomaShield emphasizes transparency by allowing users to see which detection methods flagged each row and make informed data-cleaning decisions. The application runs in a browser, where users can upload CSV or Excel files. It validates file format and size, then applies four anomaly detection techniques and combines their outputs using a consensus strategy to provide reliable and interpretable results.

II. LITERATURE SURVEY

A. Data Poisoning in Machine Learning Systems

Data poisoning is now a challenge in the world of machine learning, especially in critical domains such as finance, healthcare, and education. In their paper, Biggio and Roli (2018) [1] discussed their overview of adversarial machine learning. The authors discussed how machine learning models are being attacked during the training phase. The authors also emphasized that the assumption of clean and

independent data in the training phase of machine learning models does not hold in adversarial settings. Therefore, there is a need to enhance the validation process to ensure the reliability of the data. Recent surveys on poisoning attacks and defences have emphasized the importance of data sanitisation in enhancing the robustness of machine learning models, wherein suspicious data is eliminated before model training [2].

B. Isolation Forest for Anomaly Detection

Based on research done by Liu and others in 2008 [3], the Isolation Forest algorithm was developed, which is usually chosen to detect anomalies in spreadsheet-type information. Unlike other algorithms, it does not classify the information, instead creating chaotic tree paths on the fly and seeing where each piece stands out. The faster it is removed, the more interesting its story is, and the faster it disappears, the more normal it will be. If it splits away quickly, it will be an anomaly. The speed at which it splits away is important, as other algorithms can handle a lot of information easily. This algorithm does not require human oversight and finds things that do not fit the trend, making it good for a lot of organized information. The best part of this algorithm is that it detects unusual behaviors across various tools that can be used to find outliers.

C. One Class SVM for Novelty Detection

Back in 2001, Schölkopf and his team came up with a new technique that would be known as the One-Class Support Vector Machine. Unlike other techniques that seek to classify both normal and abnormal cases, this technique only targets the normal cases. It can be likened to a situation where a net is cast around the normal cases, and anything that does not fall within that net will be considered abnormal. This technique can be very effective even in situations where the data is found in a complex space with many different dimensions. The effectiveness of this technique, however, depends on the right choice of parameters and the correct configuration of the kernel. Due to its flexibility in dealing with different kinds of patterns in the data, this technique can be combined with other techniques for detecting abnormalities.

D. Density Based Outliers Detection Methods

With the introduction of Breunig et al. in 2000 [5], the density-based methods like Local Outlier Factor (LOF) have been used for the detection of anomalies based on the density of the point in the data set. This method detects the anomalies based on the small-scale outlying points that can be ignored by other stats-based methods. The detection of anomalies improves by using all the techniques for the detection of anomalies. This is a good technique to use when the anomalies in the data set vary or go unnoticed. Using a single technique for the detection of anomalies may cause you to lose some signals, but using all the techniques will solve the problem for you.

III. Problem Statement

Today's machine learning models are very reliant on the quality of their training data. If the data set has poisoned, corrupted, or highly anomalous data rows, the trained model may end up being biased, erratic, or even malicious. In today's business world, data is often gathered from various sources and saved as CSV or Excel files without thoroughly examining each row of the data set. It is very time-consuming and often not done at all because of the time constraints involved.

The problem that AnomaShield aims to solve is how to automatically identify and flag potentially poisoned or anomalous data rows in a tabular data set before it is used to train a machine learning model. The solution should be able to handle popular file formats, employ more than one method of identification, and provide easily interpretable results that can be understood by both technical and non-technical individuals.

IV. Existing System and Proposed Sysyem

A. Existing Methods and Limitations

Data cleaning in machine learning problems is usually carried out by using spreadsheets, single scripts, or manual data cleaning. Data analysts will look at summary statistics, sort data, and remove data points that are considered to be outliers or missing. Although these methods are sufficient for small data, they are not adequate for large and complex data. Manual data cleaning is a time-consuming, unreliable, and error-prone process. Single-method solutions for data cleaning, such as using statistical boundaries alone, can either remove valid data points or fail to detect poisoned data rows.

B. Need for an Automated Approach

Manual data cleaning is not scalable, reliable, or reproducible. Anomaly detection in multiple features is not possible through manual data analysis.

C. Proposed System

The proposed system, AnomaShield – Data Poison Detection System, is a web-based application that identifies anomalous or poisoned rows in tabular data. Users can upload their CSV or Excel files. The system uses various detection techniques, including Z-score, IQR, Isolation Forest, and One-Class SVM. The results are then combined using a consensus rule to identify the anomalous rows. The results are then presented in a dashboard, and users can download a cleaned dataset with the anomalous rows removed.

D. Advantages of Proposed System

The proposed system is fully automated and requires less human intervention. The system uses various detection techniques to ensure accuracy and reliability. The application of the consensus rule in the proposed system minimizes the possibility of false positives, and the web-based application makes the system user-friendly. The cleaned datasets can be directly used in machine learning applications to ensure accuracy and reliability.

V. System Architecture

The AnomaShield design is based on a standard web application design with a simple path from input to output. The design consists of a number of components that can be listed as follows:

User and upload interface

- The user interacts with the web application using a browser and logs in.
- The user selects a CSV or Excel file from the home page and clicks on the upload/analyze button.
- The browser uploads the file to the server using HTTP.

Backend Upload and Validation

- The Django backend receives the file and validates it (file type and size).
- If valid, the file is stored in a specific directory on the server.
- An upload record is stored in the database with details such as the file name, upload time, and status

Detection Engine

- The backend transmits the uploaded dataset to the detection engine, which reads the file, extracts the numeric columns, and applies Z-score, IQR, Isolation Forest, and One-Class SVM to highlight the anomalous rows.
- The results of each algorithm are then combined using a consensus rule to identify the rows that are labeled as suspicious or poisoned.

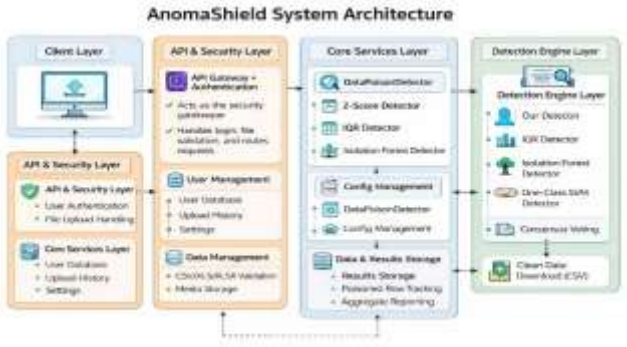
Database Storage

- Summary statistics such as total rows, flagged rows, clean rows, and detection counts for each algorithm are stored in the database.
- Row-level results, which include row index, detection status, algorithms that detected the row, and original row data, are also stored for future use.

Dashboard and Output Layer

- The dashboard presents the detection result in summary cards, graphs, and tables for flagged and clean rows.
- The user can download a cleaned dataset with only the non-flagged rows for machine learning processing.

.



System Architecture of AnomaShield

VI. Proposed Method Implementation and Algorithms

A. Z-Score Method

The Z-score method tells the distance of a data point from the mean value in terms of standard deviations. The system computes the mean and standard deviation for each numeric column and determines the Z-score for each data point.
If the absolute Z-score of any data point is greater than a certain value (for example, 4.0), the corresponding row is marked as a possible outlier. This technique is applicable when the data distribution is normally distributed.

$$Z = \frac{x - \mu}{\sigma}$$

where x= value, μ= mean, σ= standard deviation.

Rows with $|Z|$greater than a threshold (e.g., 4.0) are flagged as outliers.

B. IQR Method

The Interquartile Range (IQR) method is applied to identify outliers using quartiles. The system computes Q1, Q3, and IQR for each numeric column and specifies an acceptable range using a multiplier (for example, 2.5 × IQR).

$$IQR = Q3 - Q1$$

where $k$is a multiplier (e.g., 2.5).

Values outside this range are marked as anomalous.

C. Isolation Forest

- Isolation Forest is a tree algorithm that uses isolation to identify data as anomalous. Data that is easier to isolate is identified as anomalous.
- In the system, numeric data is processed using an Isolation Forest model from scikit-learn, and rows that are marked as anomalies are flagged using this method.

D. One Class SVM

- One-Class SVM is a model that identifies the boundary of normal data and identifies data that is beyond the boundary as anomalous. Numeric data is scaled before using the model.
- The model identifies each row as normal or anomalous, and rows identified as anomalies are flagged for further analysis.

E. Consensus Logic

- Each model flags rows independently as anomalous. The system then identifies how many models flagged each row.
- A row is identified as poisoned only when a certain number of conditions are met (for example, two models). This makes the system less vulnerable to false positives and more accurate.

F. Clean Dataset Generation

- After final classification, rows identified as clean are gathered, and flagged rows are discarded.
- The system produces a cleaned CSV file that holds only rows that were not flagged, ready to be used directly for machine learning or analysis.

VII. Result Analysis

The system was tested with sample datasets that included numeric features and introduced anomalies. Smaller datasets were used to test the correctness of the system, and larger datasets were used to see the performance of the system.

- The system was able to identify rows with extreme values and anomalies. Z-Score and IQR were able to identify rows with extreme values, and Isolation Forest and One-Class SVM were able to identify rows with anomalies.
- The consensus method was able to reduce false positives by choosing rows that were identified by multiple methods.
- The dashboard results included summary statistics, charts, and tables that made it easy for users to analyze the flagged and clean rows.

Fig 1: Dashboard

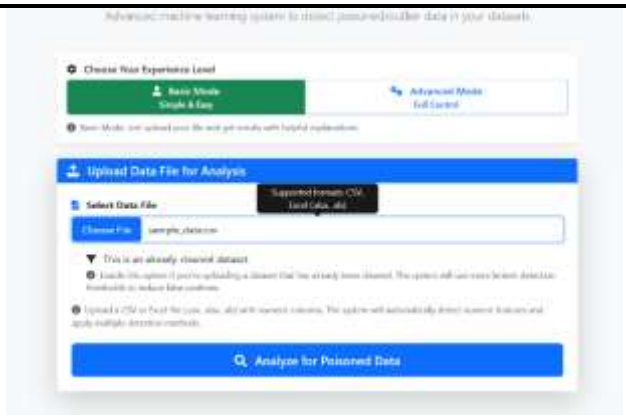


Fig 2: Uploading the dataset to verify whether dataset is poisoned or not

Fig 3:Result

Fig 4: Pie chart and Bargraph

VIII. Conclusion

AnomaShield – Data Poison Detection System is an efficient web-based system for identifying the existence of anomalous rows in tabular data. By incorporating a combination of multiple detection algorithms and a consensus algorithm, the system improves the accuracy of detection and assists in identifying anomalies even before the training of machine learning models. The system supports popular file formats and delivers a clean dataset ready for use.

The system reduces the time and effort required for data cleaning and assists in easier visualization of the existence of anomalies in the dataset. The system can be utilized by students, analysts, and developers to assist in better model performance and data quality. Although it is not a replacement for human analysis, it assists users in focusing on the most anomalous rows of a dataset.

The system can be developed further in the future by adding more algorithms for detection, supporting mixed data types, supporting large datasets, and integrating the system with data pipelines. These points can further assist in improving the usability of the system in real-world applications.

IX. Future scope

Additional Detection Methods:
Future versions can implement more advanced methods such as LOF, DBSCAN, or autoencoders to improve the detection method.

Mixed Data Types:
Extending the system to handle categorical and mixed-type data will make it suitable for a wider range of real-world datasets

API and Automation Support:
The addition of a REST API will make it easier to integrate with other software programs.

Scalability Improvements:
Asynchronous processing will enable the program to handle large amounts of data more effectively. This would prevent timeouts and support high-volume data processing.

Reporting and Configuration Improvements:
The addition of reporting and configuration capabilities will enable users to better understand the output and configure the detection.

X. References

[1] B. Biggio and F. Roli, “Wild patterns: Ten years after the rise of adversarial machine learning,” *Pattern Recognition*, vol. 84, pp. 317–331, 2018.
[2] A. Hong, X. He, W. Pan, et al., “Threats to Training: A Survey of Poisoning Attacks and Defenses on Machine Learning Systems,” *ACM Computing Surveys*, 2022. (Available via ACM Digital Library, DOI: 10.1145/3538707).
[3] F. T. Liu, K. M. Ting, and Z.-H. Zhou, “Isolation Forest,” in *Proceedings of the 2008 IEEE International Conference on Data Mining*, pp. 413–422, IEEE, 2008.
[4] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, “Estimating the support of a high-dimensional distribution,” *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001.
[5] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, “LOF: Identifying density-based local outliers,” in *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, pp. 93–104, ACM, 2000.
[6] J. Steinhardt, P. W. W. Koh, and P. Liang, “Certified defenses for data poisoning attacks,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017, pp. 3517–3529.
[7] B. Nelson, B. I. P. Rubinstein, L. Huang, A. D. Joseph, S. J. Lee, S. Rao, and J. D. Tygar, “Query strategies for evading convex-inducing classifiers,” *Journal of Machine Learning Research*, vol. 13, pp. 1293–1332, 2012.
[8] A. Paudice, L. Muñoz-González, and E. C. Lupu, “Detection of adversarial training examples in poisoning attacks through anomaly detection,” in *Proc. IEEE European Symposium on Security and Privacy*, 2018, pp. 346–362.
[9] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection: A survey,” *ACM Computing Surveys*, vol. 41, no. 3, pp. 1–58, 2009.

[10] M. Goldstein and S. Uchida, "A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data," *PLOS ONE*, vol. 11, no. 4, 2016.
[11] C. C. Aggarwal, *Outlier Analysis*, 2nd ed., Springer, 2017.
[12] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016. (See chapter on robustness and regularization.)
[13] H. Xiao, H. Xiao, and C. Eckert, "Adversarial label-flip attack and defense for graph neural networks," in *Proc. IEEE International Conference on Data Mining (ICDM)*, 2019.
[14] S. Wang, M. Tang, J. Liu, and J. Bu, "Data poisoning attacks against online learning," in *Proc. IEEE International Conference on Data Mining (ICDM)*, 2018.
[15] D. Dua and C. Graff, "UCI Machine Learning Repository," University of California, Irvine, 2017. [Online].