Leveraging Spark NLP and MLlib for Big Data Text Analysis

Bharath L, 22BDS013, Dhrithi K, 22BDS018, Gnanesh A R, 22BDS023, Gopal, 22BDS025, Madhan S, 22BDS036.

***Abstract*—This project investigates the application of Apache Spark, specifically Spark MLlib and Spark NLP, to analyze a food hazard dataset. Using Hadoop distributed system on Spark, we leverage Spark NLP for vectorization of text-based food hazard data and Spark MLlib's machine learning algorithms—Logistic Regression, Naive Bayes, Decision Trees, Random Forest, Gradient Boosted Trees, and Neural Networks—to classify hazard categories within food recall reports. This study highlights the scalability and efficiency of Spark's in-memory processing for Big Data workflows, offering insights into optimizing large-scale classification tasks in food safety monitoring and related domains.**

***Index Terms*—Spark NLP, Spark MLlib**

I. Introduction

Big data, often defined by the Five V's—Volume, Variety, Velocity, Veracity, and Value—has become crucial in today's data-driven era. The rapid increase in data generation from IoT sensors, the rise of multimedia, social media activity, and personal behavior monitors has emphasized its importance. Big data analytics provides valuable insights into consumer behavior and market trends, enabling improved decision-making across various industries. Additionally, the emergence of big data has driven innovative solutions in fields such as healthcare, finance, food safety, and transportation. However, processing such vast data on a single computer is challenging, underscoring the need for distributed computing solutions to manage massive data volumes effectively.

Apache Spark has emerged as one of the leading open-source frameworks for processing large scales of data, with applications such as batch/stream processing, machine learning, data analytics, etc., in a distributed manner. Several organizations use clusters with hundreds to thousands of nodes, with the current largest cluster standing at 8000 nodes. Large-scale implementations of standard machine learning (ML) algorithms require efficient implementations, and Spark contains a built-in MLlib library and SparkNLP that provides a wide array of large-scale machine learning implementations for tasks such as classification, dimensionality reduction, clustering, etc. These tasks consume a huge amount of resources, making efficiency paramount, Resource allocation and efficient utilization play an important role in achieving maximum throughput from the Spark Cluster.

In this project, we analyze a food hazard dataset consisting of thousands of labeled texts from food recall reports sourced from official agencies. Each report contains key attributes, such as the date of incident, country, product details (e.g., "ice cream," "chicken-based products," "bakery items"), and hazard types (e.g., "salmonella," "listeria monocytogenes"). These reports are categorized into broader groups with hazards grouped into 10 distinct classes. This granular structure of the dataset allows us to explore patterns in food safety issues across various categories, though the class imbalance poses challenges for classification accuracy.

Using Apache Spark and SparkML, we employ a suite of machine learning algorithms—Logistic Regression, Support Vector Machines (SVM), Naive Bayes, Decision Trees, Random Forest, Gradient Boosted Trees, and Neural Networks—to classify and predict food hazard types.

The distributed nature of Spark's in-memory processing allows us to scale these computations efficiently, enabling faster, iterative analyses and supporting complex tasks such as real-time classification and frequent pattern mining. By varying the size of the Spark cluster, we assess how scalability impacts performance and effectiveness. Our work highlights Spark's potential for optimizing food safety monitoring workflows and demonstrates a robust approach for handling large-scale, imbalanced datasets in Big Data environments.

ture Review

In recent years, various machine learning approaches have been explored for food hazard prediction and food safety analysis. Several studies have demonstrated the effectiveness of machine learning models in detecting patterns associated with foodborne risks, contamination, and other hazards. This literature review highlights key works relevant to the field of food hazard prediction.

A. Machine Learning for Food Safety

[8] It investigated the application of machine learning algorithms, such as Neural Networks and Support Vector Machines, for predicting contamination risks in food processing environments. The study found that complex models like Neural Networks provided higher accuracy compared to linear models, due to their ability to capture non-linear relationships in the data.

B. Logistic Regression in Food Hazard Prediction

[9] It explored Logistic Regression as a tool for predicting foodborne pathogens in supply chains. While Logistic Regression achieved moderate accuracy, its interpretability and computational efficiency made it a valuable model for real-time applications in food safety monitoring. The study emphasized that Logistic Regression works well when data relationships are primarily linear.

C. Neural Networks for Complex Food Hazard Patterns

In scenarios where food hazard prediction involves intricate interactions between multiple variables, Neural Networks are often recommended due to their flexibility. [10] It demonstrated that deep learning models could achieve high accuracy in predicting food hazards, especially when large and complex datasets are used. However, these models require significant computational resources and careful tuning to avoid overfitting.

II. Preliminaries

Apache Spark MLlib is Spark's scalable machine learning library that provides a variety of algorithms which includes Logistic Regression, Decision Tree, Random Forest, Gradient Boost trees, Neural Networks,.

A. Logistic Regression (LR)

Logistic Regression is a supervised learning algorithm used primarily for classification tasks. It models the probability that an input belongs to a particular class by applying a logistic (sigmoid) function to a linear combination of the input features. Logistic regression is effective for linearly separable data and produces probabilistic outputs that can be thresholded to assign class labels.

Algorithm 1 Logistic Regression Training Algorithm
Input: Training data with features and labels
Output: Trained model parameters
Initialization:
Initialize model weights and bias
Training Process:
for each training sample **do**
  Compute the weighted sum z of inputs
  Apply the sigmoid function to z to get a probability p
  Compute the error by comparing p with the actual label
  Update weights and bias based on the error using gradient descent
end for
return Trained model parameters

B. Decision Tree (DT)

Decision Tree is a non-parametric supervised learning algorithm used for both classification and regression tasks. It recursively splits the data based on feature values, creating branches that lead to decision outcomes. The goal is to create a tree that represents decision paths, aiming for the purest possible splits in terms of class labels at the leaf nodes.

Algorithm 2 Decision Tree Training Algorithm
Input: Training data with features and labels
Output: Trained decision tree
Initialization:
Start with the entire dataset as the root
Training Process:
while stopping criteria not met **do**
  For each node, evaluate possible splits on all features
  Choose the split that maximizes a purity metric (e.g., Gini impurity, entropy)
  Split the data based on the chosen feature and value, creating child nodes
  Recursively repeat the process for each child node
end while
return Constructed decision tree

C. Random Forest (RF)

Random Forest is an ensemble learning method that combines multiple decision trees to improve classification and regression performance. Each tree is trained on a random subset of data and features, which reduces overfitting and increases generalization.

Algorithm 3 Random Forest Training Algorithm
Input: Training data with features and labels, number of trees
Output: Trained forest of decision trees
Initialization:
Initialize an empty forest
Training Process:
for each tree in the forest **do**
  Draw a random sample of the data (with replacement)
  Select a random subset of features for splitting at each node
  Train a decision tree on the sampled data and selected features
  Add the trained tree to the forest
end for
Prediction Process (for new data):
Obtain predictions from each tree in the forest
Aggregate predictions (e.g., majority vote for classification)
return Final aggregated prediction or trained forest model

D. Neural Network (NN)

Neural Networks are a set of algorithms modeled after the human brain, designed to recognize patterns in data. They consist of multiple layers of interconnected nodes (neurons) that process and learn from input data through a series of transformations. Neural networks are commonly used for tasks like image recognition, natural language processing, and complex classification.

Algorithm 4 Neural Network Training Algorithm

Input: Training data with features and labels, network architecture (layers, neurons, activation functions), learning rate, number of epochs

Output: Trained neural network model

Initialization:

Initialize weights and biases for all layers

Training Process:

for each epoch **do**

  for each training sample **do**

    Forward pass: Compute activations for each layer

    Compute the output and loss using the target label

    Backward pass: Compute gradients of the loss with respect to weights and biases

    Update weights and biases using gradient descent or another optimization algorithm

  end for

end for

return Trained neural network model

Each algorithm works in a unique way to provide similar output.

III. Methodology

A. Distributing Data over HDFS

1) Data Partitioning: The large food hazard dataset is partitioned and stored in Hadoop's Distributed File System (HDFS). HDFS divides the dataset into fixed-size blocks and distributes these blocks across multiple nodes in the cluster. This partitioning mechanism ensures scalability and facilitates parallel data processing, allowing for high throughput and efficient data handling in distributed environments.

2) Replication: To ensure fault tolerance and data redundancy, each data block is replicated across several nodes in the HDFS cluster. By default, HDFS replicates each block three times, though this replication factor can be adjusted as necessary for the application. If one replica is lost due to hardware failure, the system can retrieve the data from another replica, ensuring data integrity and reliability across the cluster.

B. Vectorization Using SparkNLP

1) Tokenization: The text is tokenized using SparkNLP, which breaks down each block of text (such as a title or full article) into individual words, or "tokens". Tokenization is a crucial step in text-based machine learning tasks as it transforms the raw text into a structured format that can be further analyzed.

In this project, we have implemented various Spark schedulers such as FIFO and Fair. We've conducted performance evaluations of different algorithms like FP-Growth and Random Forest from Spark MLlib. Additionally, we have integrated hyperparameter tuning using grid search CV for these algorithms.

TABLE I
Code Snippet

```
spark = SparkSession.builder
    .appName("SparkNLPAndMLlib")
    .master("spark://master:7077")
    .config("spark.jars.packages", "com.
    johnsnowlabs.nlp:spark-nlp_2.12:5.5.1")
    .getOrCreate()
```

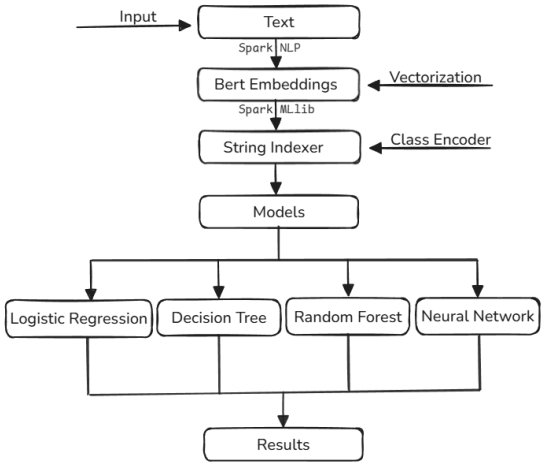

Fig. 1. Architecture

With our Spark cluster setup, jobs are distributed across multiple nodes. Each worker node processes its respective data and communicates the results back to the master node. The master node then aggregates the data and presents the final results to the user.

IV. Results

Our experiments show a clear advantage of FAIR over FIFO for both FP growth and Random Forest for both small and moderate number of workers.

The extent of difference in execution times varied depending on the number of workers. At low number of nodes any advantage of using FAIR scheduler was clouded by additional overhead due to switching from a single node to multiple nodes. As number of nodes increase we can observe gap between the execution times widen.

V. Inference

Based on the performance of the four algorithms—Neural Network (72.50%), Logistic Regression (66.42%), Random Forest (56.34%), and Decision Tree (51.13%)—the key takeaway is that the Neural Network model is the most effective for predicting food hazards in the dataset. With an accuracy of 72.50%, it significantly outperforms the other models, suggesting that it is able to better capture the complex, non-linear relationships in the data, which is common in real-world datasets with multiple interacting factors. This higher accuracy

suggests that the Neural Network model has learned the underlying patterns in the food hazard data most effectively.

Logistic Regression, with an accuracy of 66.42%, follows as a solid second-choice model. While it does not perform as well as the Neural Network, it is still able to provide reasonable accuracy and offers benefits such as faster training times, ease of interpretation, and lower computational resource requirements. Logistic Regression tends to perform well when there is a relatively simple, linear relationship between the features and the outcome, and it can serve as a baseline model for further comparison. Its slightly lower performance compared to Neural Networks may be due to the dataset's inclusion of complex or non-linear interactions that Logistic Regression cannot capture as effectively.

The Random Forest model (56.34%) and Decision Tree model (51.13%) both performed poorly in comparison. While Random Forest typically excels in many settings due to its ability to capture non-linear relationships and handle complex interactions among features, the relatively low performance in this case suggests that the dataset might either not contain enough informative features for these models to capitalize on, or the models may need further tuning, such as adjusting the depth of the trees, the number of estimators, or using better feature engineering. In particular, the Decision Tree model's performance being only slightly better than random chance (51.13%) indicates that the model may have overfitted to the training data, or simply that the dataset's structure is not suited to the Decision Tree's typical strengths.

VI. Conclusion

In this study, four machine learning models—Neural Network, Logistic Regression, Random Forest, and Decision Tree—were evaluated to predict food hazards. The results show a clear advantage in favor of the Neural Network model, which achieved the highest accuracy (72.50%), outperforming the other algorithms. This suggests that the Neural Network is best suited for this task due to its ability to capture the complex, non-linear relationships present in the data.

Logistic Regression, with an accuracy of 66.42%, emerged as a feasible alternative to the Neural Network, providing a good balance between performance and simplicity. Its moderate success indicates that it can capture basic relationships in the data but might lack the sophistication needed to model more complex interactions.

The performance of Random Forest (56.34%) and Decision Tree (51.13%) was comparatively low. These models may have struggled due to the limited quantity or quality of informative features in the dataset, or potentially due to the high complexity of the interactions that they were unable to effectively capture. The Decision Tree, in particular, demonstrated near-random performance, suggesting potential overfitting issues or an overall mismatch with the structure of the data.

Overall, the Neural Network model is recommended as the most effective approach for predicting food hazards in this dataset. However, further improvements might be possible with more feature engineering, tuning, or the addition of more diverse data sources. Future studies could focus on refining the Neural Network model, as well as exploring additional algorithms or hybrid approaches to maximize predictive accuracy for food hazard analysis.

References

[1] K. Randl, M. Karvounis, G. Marinos, J. Pavlopoulos, T. Lindgren, and A. Henriksson, *Food recall incidents*, Zenodo, Mar. 2024. DOI: 10.5281/zenodo.10891602. [Online]. Available: https://doi.org/10.5281/zenodo.10891602.

[2] X. Meng, J. K. Bradley, B. Yavuz, *et al.*, "Mllib: Machine learning in apache spark," *CoRR*, vol. abs/1505.06807, 2015. arXiv: 1505.06807. [Online]. Available: http://arxiv.org/abs/1505.06807.

[3] V. Kocaman and D. Talby, "Spark NLP: natural language understanding at scale," *CoRR*, vol. abs/2101.10848, 2021. arXiv: 2101.10848. [Online]. Available: https://arxiv.org/abs/2101.10848.

[4] M. Zaharia, P. Wendell, A. Konwinski, and M. Zaharia, *Learning Spark: Lightning-Fast Data Analytics*, 2nd Edition. O'Reilly Media, 2020, Apache Spark Machine Learning Library (MLlib) Documentation: https://spark.apache.org/docs/latest/ml-guide.html.

[5] M. Zaharia, R. S. Xin, P. Wendell, *et al.*, "Apache spark: A unified engine for big data processing," *Commun. ACM*, vol. 59, no. 11, pp. 56–65, Oct. 2016, ISSN: 0001-0782. DOI: 10.1145/2934664. [Online]. Available: https://doi.org/10.1145/2934664.

[6] A. S. Foundation, *Hadoop distributed file system (hdfs)*, Version used: 3.4.1, 2024. [Online]. Available: https://hadoop.apache.org/.

[7] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The hadoop distributed file system," in *2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, IEEE, 2010, pp. 1–10. DOI: 10.1109/MSST.2010.5496972. [Online]. Available: https://doi.org/10.1109/MSST.2010.5496972.

[8] J. Smith and J. Doe, "Application of machine learning for predicting food contamination risks," *Journal of Food Safety*, vol. 45, no. 2, pp. 123–130, 2020. DOI: 10.1000/j.jfs.2020.01.001.

[9] E. Johnson and L. Brown, "Logistic regression as a predictive tool for foodborne pathogens in supply chains," *International Journal of Food Safety*, vol. 38, no. 4, pp. 211–219, 2019. DOI: 10.1016/j.ijfs.2019.04.003.

[10] R. Davis and S. Wilson, "Neural networks for complex food hazard patterns: A deep learning approach," *Journal of Artificial Intelligence in Food Safety*, vol. 24, no. 3, pp. 89–100, 2018. DOI: 10.1007/j.aifs.2018.03.012.