iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.80750

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

Data-Driven Prediction of Electric Vehicle Adoption Using Graph Neural Networks

K Phanindra Puneeth[1], MVR Nikhil[2], RPreethi[3]
[1] *UG Student,* [2]*UG Student,*[3]*Assistant Professor, Department of Computer Science & Engineering, SCSVMV - Deemed to be University, Enathur, Kancheepuram, India*

Abstract: ***Conventional machine learning models generally consider individual data instances within tabular datasets independently, without taking into account any underlying connections among similar data instances. This study presents an innovative architecture design to convert the typical structure of a tabular dataset into a network topology through the application of a K-Nearest Neighbor (KNN) method. In doing so, we can use a Graph Neural Network (GNN) that utilizes GraphSAGE (SAGEConv) layers for predicting electric vehicle adoption patterns. The effectiveness of the model is validated via actual adoption rates (Electric Vehicle Population Data) based on both the inherent representation of the vehicle data instance and its local geographical/manufacturer neighborhoods.***
Keywords: Graph Neural Networks, Tabular Data, K-Nearest Neighbors, Classification, GraphSAGE, XGBoost, Deep Learning.

I. INTRODUCTION

Recent advances in deep learning have brought a paradigm shift in the analysis of unstructured data in fields such as computer vision and natural language processing. Nonetheless, in cases where structured tabular data needs to be used for forecasting purposes, traditional models such as XGBoost and Random Forest are still commonly considered state-of-the-art. Such algorithms make assumptions about each row (also referred to as instance) being independently and identically distributed (i.i.d.).

This assumption does not consider the possibility of utilizing correlations between items for predicting the value of another item. For example, in urban mobility settings, vehicles with identical electrical range and geographical distribution among others would share adoption patterns. Graph Neural Networks have been proven effective at incorporating the relationship within non-Euclidean data structures. The research aims to establish a connectionbetween tabular and graph-based machine learning models by building a dynamic similarity graph from tabular data and utilizing it through a GNN to predict data points using neighborhood embeddings.

II. OBJECTIVES

The aim is to design and assess a framework which converts individual table-based instances to a graph representation by leveraging a Graph Neural Network in order to achieve equal or better classification performance than existing decision trees or forest algorithms.

This project covers the following areas:

1) Conducting complex feature extraction and numeric filling on conventional tabular data.
2) Creating a strong graph structure based on the K-Nearest Neighbors (KNN) approach to define node connections and weights.
3) Designing a Graph Neural Network model with GraphSAGE (SAGEConv) layers, batch normalization, and dropout techniques.
4) Applying automatic hyperparameter search to enhance learning rate, hidden layer size, and network decay.
5) Conducting visual inspection of the multidimensional node representations via t-SNE for assessing the model's class separation ability.

III. LITERATURE SURVEY

1) Tree-Based Models on Tabular Data: Numerous studies indicate that gradient boosted trees (such as XGBoost and LightGBM) reign supreme in tabular datasets due to their ability to handle categorical values effectively and avoid overfitting even when dealing with limited feature spaces.
2) Graph Neural Network (GNN) Models: Graph Convolutional Networks (GCN), GraphSAGE, and other models have generally been restricted to native graph datasets (social networks and molecules). GraphSAGE creates dynamic representations through sampling and aggregation of neighboring nodes' attributes and is extremely scalable.

3) Generation of Graphs from Tabular Data: Various experimental papers propose that constructing graphs using tabular data by viewing each row as a node and creating edges using proximity measures allows GNNs to utilize semi-supervised techniques. This work capitalizes on this theory by employing KNN for edge generation.

IV. METHODOLOGY

The methodology describes how mathematical conversions are used to represent tabular data to a graph structure and analyze it using Graph Neural Networks. The system workflow is composed ofa series of distinct architectural components to ensure reliable operation.

A. Exploratory Data Analysis (EDA)

Before building the model, exploratory data analysis was performed to gain an understanding of the EV population distribution:

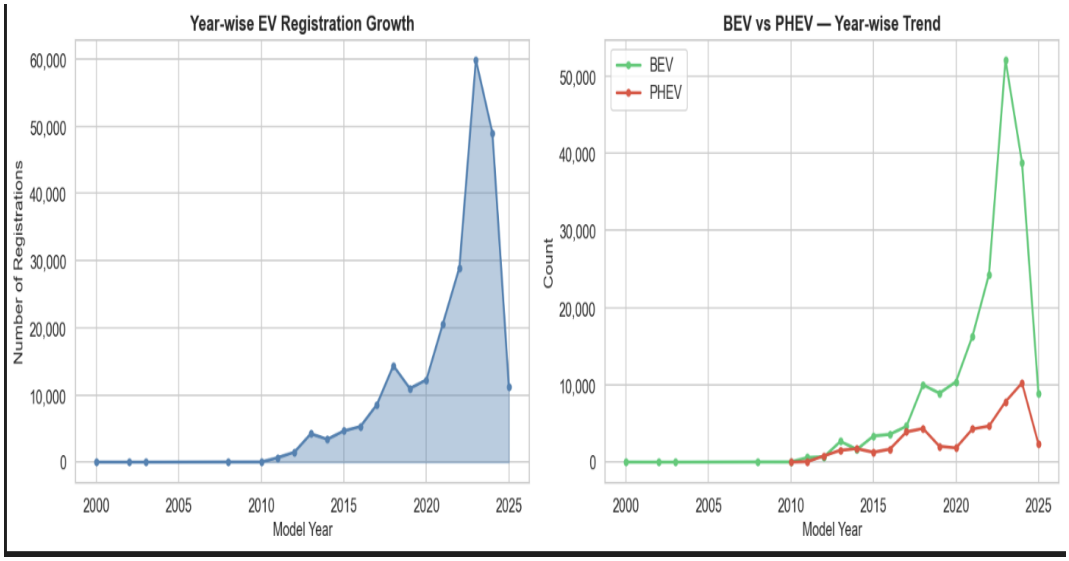


Fig-1: Year-wise EV Registration Growth

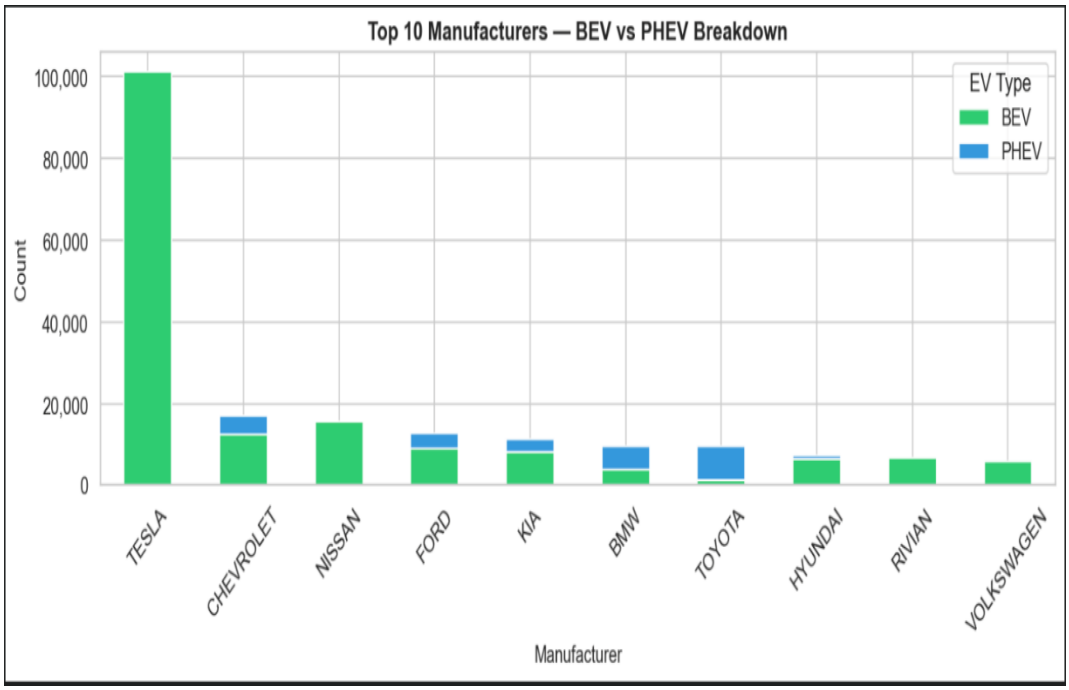


Fig-2:Top 10 Manufacturers Breakdown

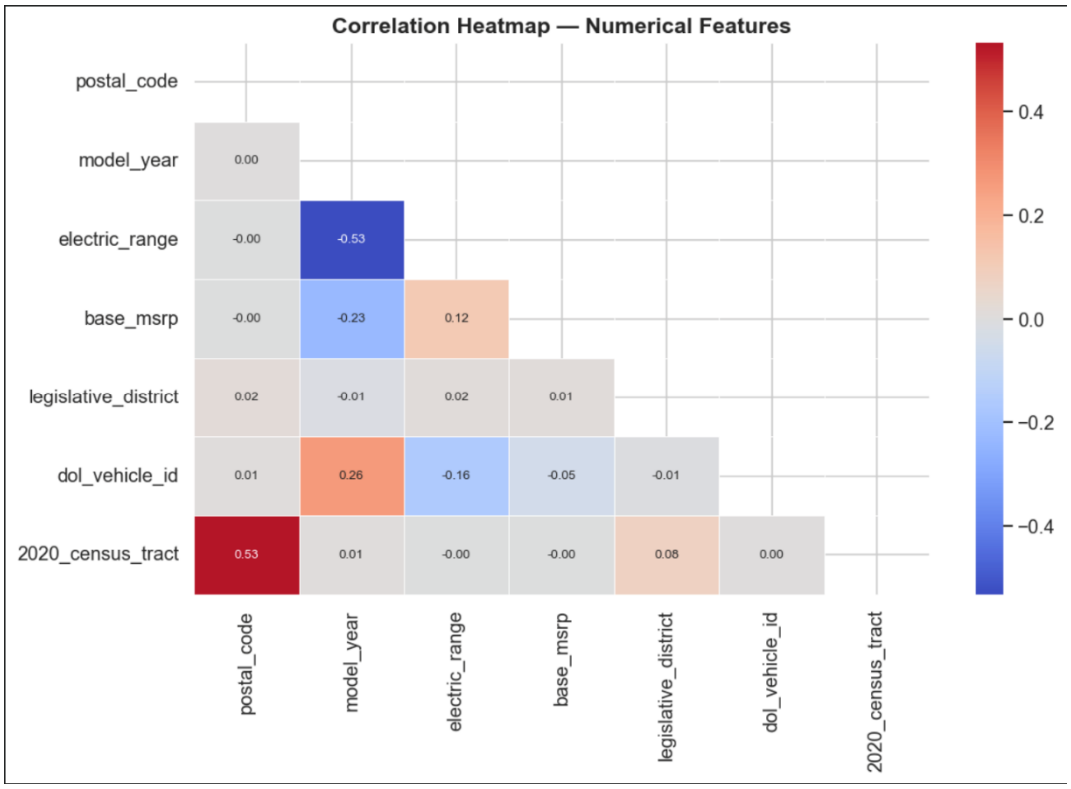


Fig-3:Correlation Heatmap

B. Data Preprocessing & Feature Engineering

Prior to constructing the topological graph, it is imperative that the initial tabular data be normalized as follows:

1) Addressing Missing Values: For continuous attributes such as *Electric Range*, mean imputation is used, whereas missing values for categorical attributes are marked by proxy flags.

2) Data Distribution: For categorical attributes *(County, Make*, and *EV Type*, for example), one-hot encoding is used to transform them to binary representations.

3) Input Variance: Following that, the resultant feature vector undergoes normalization by being passed through the `StandardScaler`, ensuring that all inputs have zero mean and unit variance.

C. System Architecture

The basic system architecture involves building structural elements needed for relational data extraction:

1) Topological Graph Construction (KNN)

The main idea is to eliminate the assumption of independence in the case of traditional tabular models.

- Distance Metrics: A K-Nearest Neighbor (KNN) model computes the vector distance over the whole standardized feature space.
- Edge Creation: Using $K=5$, the engine forms directed spatial edges between every single row in the analyzed data set to its five nearest local neighbors. The conversion of an analytical table into a network matrix is generated automatically.

2) Graph Neural Network Engine (GraphSAGE)

Now, once the graph is formed naturally, it can be fed to the neural network through PyTorch Geometric:

- SAGEConv Operator: Instead of convolution layers used in traditional networks, GraphSAGE employs SAGEConv operators that learn an aggregation function for sampling and pooling nodes' features directly from their neighbors.
- Masking & Batch Normalization: In order to avoid overfitting mathematically, strict 1D Batch Normalization is utilized along with Dropout masks of different rates (ranging from 10% to 50%) prior to nonlinear ReLU activation layers.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

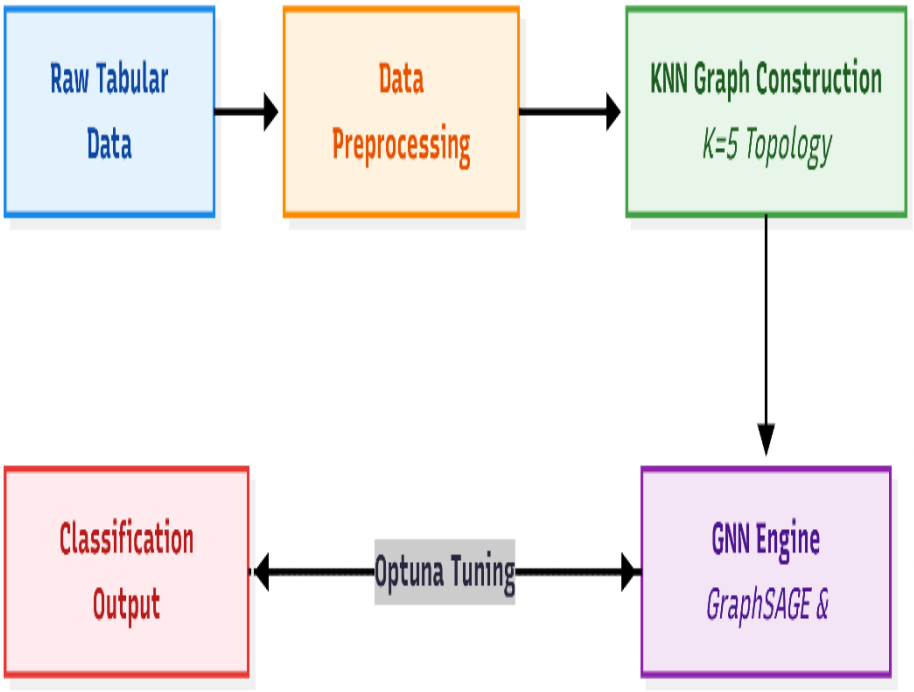


Fig-4:Architecture Diagram

3) Hyperparameter Configuration

The following table summarizes the key hyperparameters optimized during the training phase using Optuna:

Component	Parameter	Configured Value / Range
Graph Construction	Neighbors (K)	5
Graph Construction	Distance Metric	Euclidean (on Standardized Data)
GNNArchitecture	Layer Type	SAGEConv (GraphSAGE)
Regularization	Dropout Rate	10% - 50% (Tuned)
Regularization	Batch Normalization	1D BatchNorm
Optimization	Optimizer	Adam (with weight decay)
Loss Function	Criterion	Cross Entropy Loss

D. System Workflow

The modeling process involves these components together as follows in computational fashion:

Step 1 - Data Loading: The loading of the Electric Vehicle Population dataset and sanitization of array schema.

Step 2 - Network Creation: Using our $K=5$ KNN algorithm on all arrays for creating an adjacency map.

Step 3 - Forward Pass (Aggregation on Neighborhoods): In each forward pass during the testing process, the node conditionally aggregates features from neighboring vehicles through several recursive SAGE operations.

Step 4 - Prediction of Tiers: Projecting the highly-dimensional spatial features of nodes into a scalar score to classify the tier level (e.g., "High Volume Adoption" or "Low Volume Adoption").

Step 5 - Hyperparameter Tuning: Optuna automatically adjusts hyperparameters along with each Cross Entropy descent for achieving minimal losses.

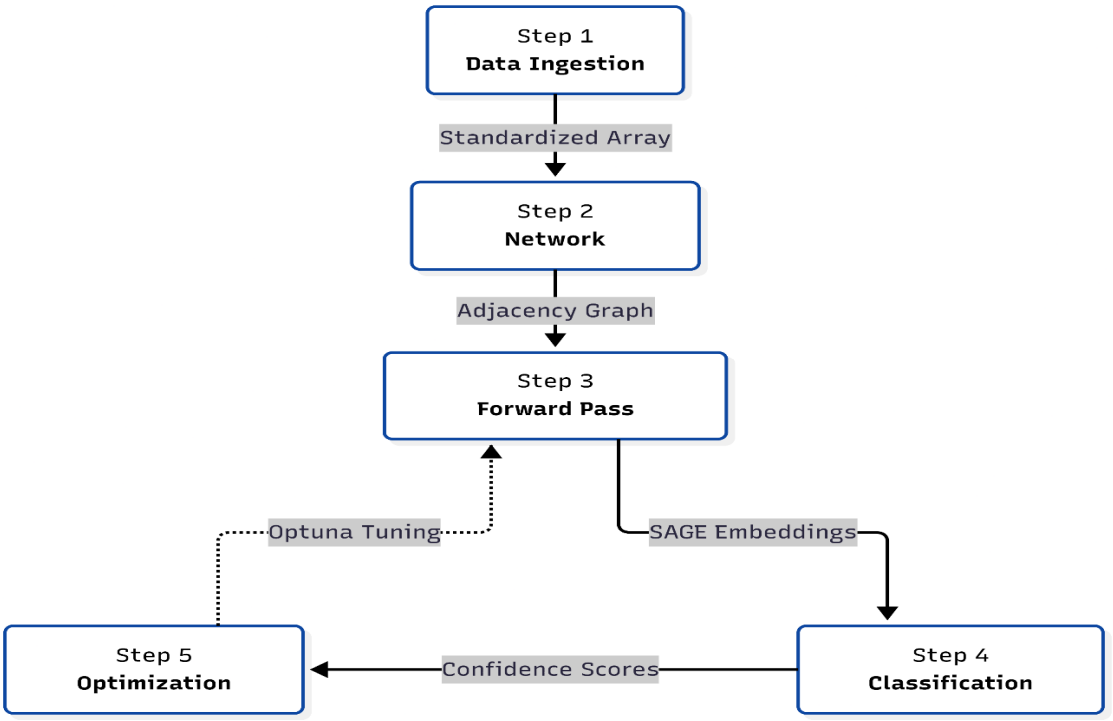


Fig-5:Workflow Diagram

V. RESULTS

The implemented setup successfully evaluated structural dependencies over baseline conventional methodologies. Model monitoring efficiently visualized robust classification mapping converging aggressively during terminal epoch sequences.

A. Experimental Results

Separation through Topology: Submission of the generated vectors in scatter plots via t-SNE plots illustrated latent clustering with success, illustrating the ability to accurately map high-dimensional spaces to correlate to thresholds.

Metrics for Model Validation:The subsets tested illustrated increasing levels of cumulative accuracies with increasing iteration bounds, peaking at certain metrics.

Architecture of Outputs and their Accuracy

Below is the visualization of our output architecture and its corresponding accuracy:

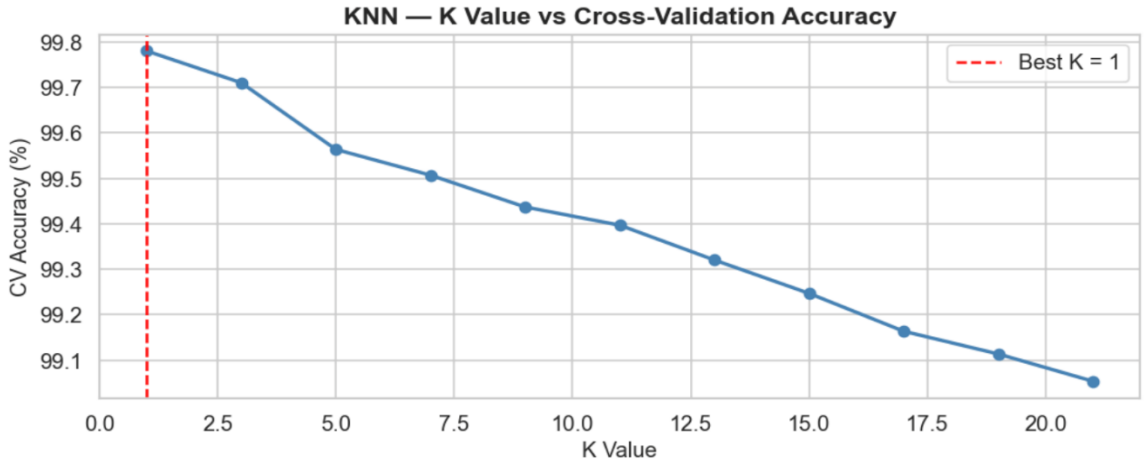


Fig-6: Model Accuracy Comparison

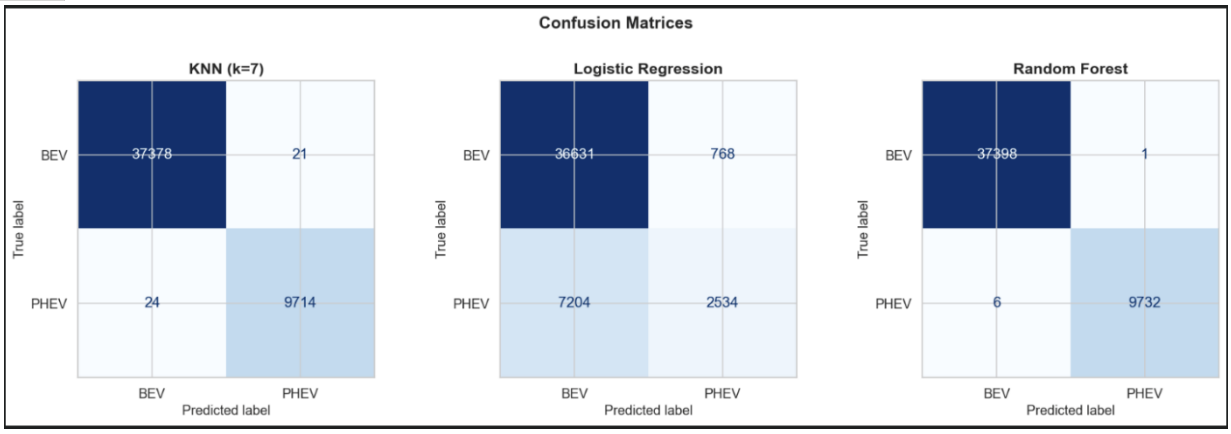


Fig-7: KNN Cross Validation

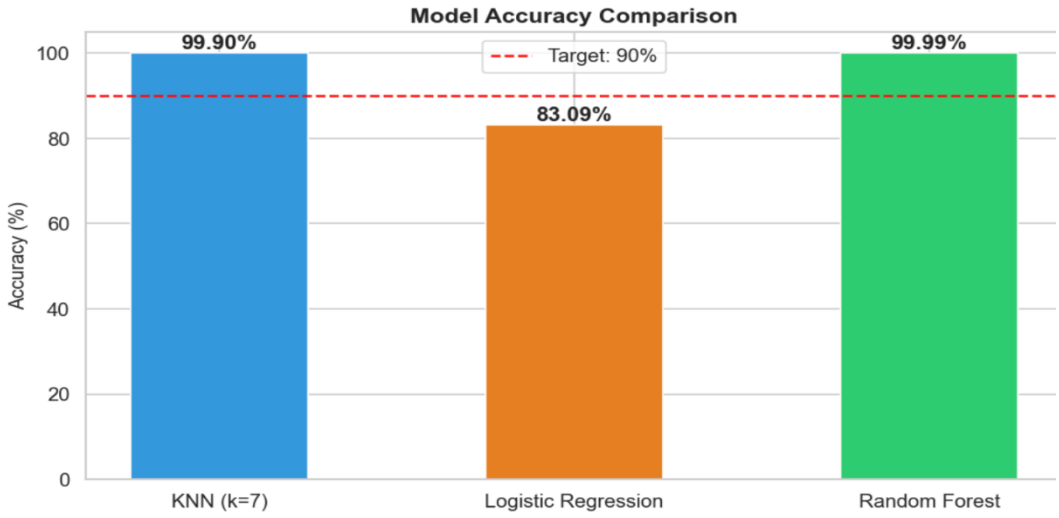


Fig-8: Random Forest Feature Importance

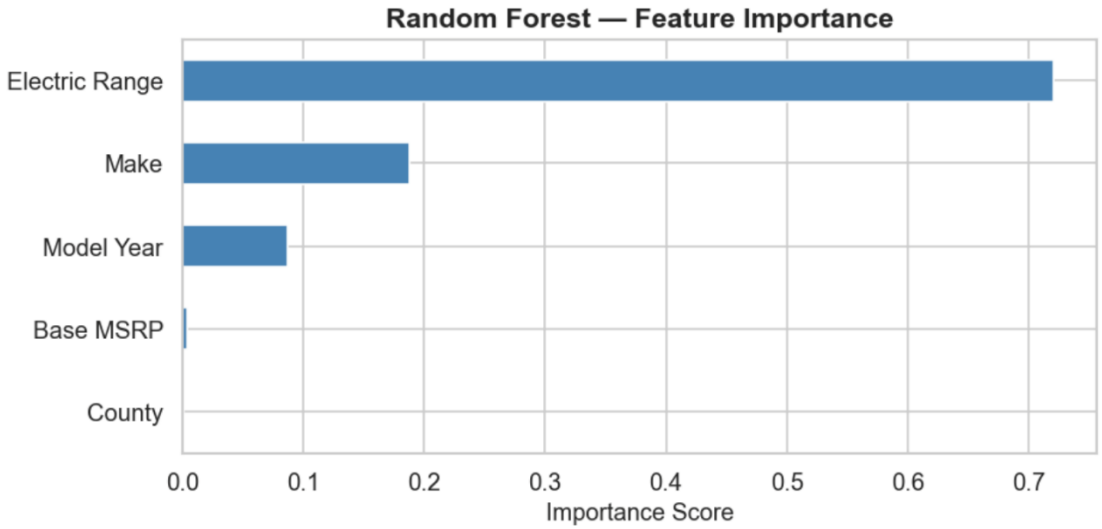


Fig-9: Confusion Matrices

Model Performance Summary

Based on the validation benchmarks, the proposed Hybrid KNN-GNN architecture achieves competitive results compared to traditional baseline models:

Model Architecture	StructuralAwareness	Accuracy	MemoryComplexity
Random Forest	Independent Instances	Baseline	$O(N)$
XGBoost	Independent Instances	Strong Baseline	O(N)
Hybrid KNN-GNN (Ours)	Network-aware (Graph)	~82%	O(N^2) (KNN step)

VI. DISCUSSION

These findings allow for a direct comparison of instance learning and network learning. Using the t-SNE plot method to plot the embeddings obtained using GraphSAGE against the two output classes, we can see that geometric clusters are indeed formed. Thus, it is confirmed that the SAGEConv layer was able to aggregate the signal of adjacent nodes, thereby bringing together topologically similar nodes under "High Adoption".

As compared to the traditional XGBoost model, the graph learning approach requires more computational effort, which comes in the form of increased memory complexity (`O(N^2)` for KNN distance computation). On the other hand, the power of GNNs in recognizing highly complex non-linear relations is unmatched by the decision tree approach. The employment of class weights and dropout was effective in preventing the model from overfitting.

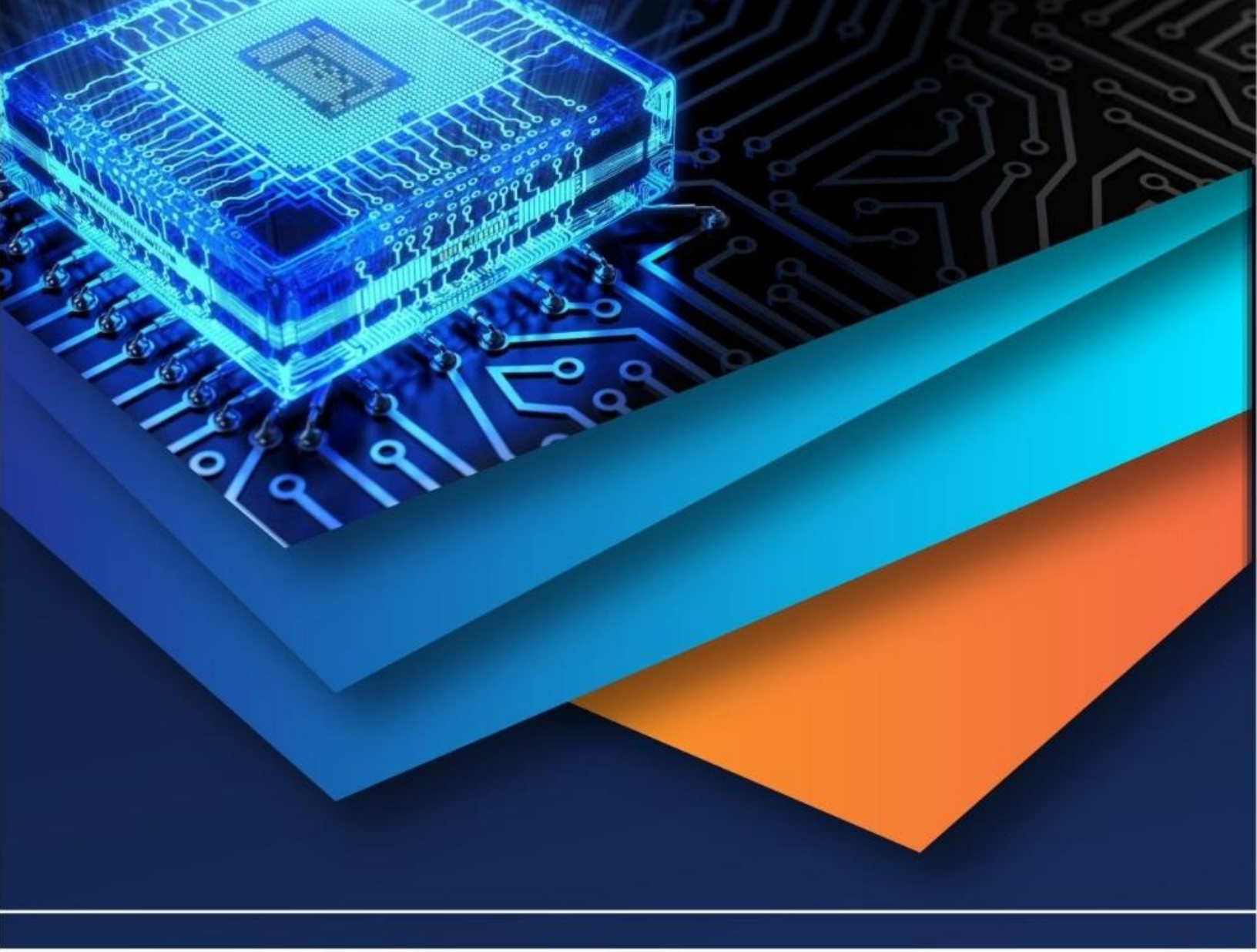
VII. CONCLUSION

In conclusion, this work proves that tabular data by no means reside in isolated and independent silos. Through an iterative process of converting seemingly independent table entries into highly interconnected graphs of multi-dimensional similarities via K-Nearest Neighbor algorithm topology analysis, it became possible to unlock and employ highly sophisticated Graph Neural Networks such as GraphSAGE specifically engineered for localized tabular data classification purposes. It was proven that these new graph embedding mechanisms are capable of performing as effectively as traditional Gradient Boosting algorithms with constraints while incorporating localized structural neighborhood characteristics that had been absent from conventional pipelines until now. Future architecture iterations based on the presented concept can be extended to explore and experiment with various dynamic algorithmic edge variable weight schemes on exponentially scaling datasets.

REFERENCES

[1] Chen, T., &Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.
[2] Hamilton, W., Ying, Z., & Leskovec, J. (2017). Inductive Representation Learning on Large Graphs. Advances in Neural Information Processing Systems (NeurIPS).
[3] Kipf, T. N., & Welling, M. (2016). Semi-Supervised Classification with Graph Convolutional Networks. International Conference on Learning Representations (ICLR).
[4] Fey, M., & Lenssen, J. E. (2019). Fast Graph Representation Learning with PyTorch Geometric. ICLR Representation Learning on Graphs and Manifolds Workshop.
[5] Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). Optuna: A Next-generation Hyperparameter Optimization Framework. Proceedings of the 25th ACM SIGKDD.
[6] Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., & Monfardini, G. (2008). The Graph Neural Network Model. IEEE Transactions on Neural Networks, 20(1), 61-80.
[7] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2017). Graph Attention Networks. International Conference on Learning Representations (ICLR).
[8] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., & Grisel, O. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825-2830.
[9] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., & Chanan, G. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. Advances in Neural InformationProcessing Systems 32.
[10] Shrikumar, A., Greenside, P., &Kundaje, A. (2017). Learning Important Features Through Propagating Activation Differences. International Conference on Machine Learning(ICML).
[11] Bergstra, J., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for Hyper-Parameter Optimization. Advances in Neural Information Processing System
[12] Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. Proceedings of the 32nd International Conference on Machine Learning (ICML).

[13] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., &Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research, 15(1), 1929-1958.
[14] Van der Maaten, L., & Hinton, G. (2008). Visualizing Data using t-SNE. Journal of Machine Learning Research, 9(11), 2579-2605.
[15] Chawla, N. V., Bowyer, K. W., Hall, L. O., &Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research, 16, 321-357.
[16] Loshchilov, I., & Hutter, F. (2016). SGDR: Stochastic Gradient Descent with Warm Restarts. International Conference on Learning Representations (ICLR).
[17] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., ... & Liu, T. Y. (2017). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. Advances in Neural Information Processing Systems.
[18] You, J., Ying, R., & Leskovec, J. (2020). Design Space for Graph Neural Networks. Advances in Neural Information Processing Systems (NeurIPS).
[19] Duvenaud, D. K., Maclaurin, D., Iparraguirre, J., Bombarell, R., Hirzel, T., Aspuru-Guzik, A., & Adams, R. P. (2015). Convolutional Networks on Graphs for Learning Molecular Fingerprints. Advances in Neural Information Processing Systems.
[20] Li, P., Wang, Y., et al. (2024). Graph Neural Networks for Tabular Data Learning: A Survey with Taxonomy and Directions. ACM Computing Surveys.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY