Enhanced Regression Modeling for Short-Time Prediction of Genetic Algorithm Outcomes in Vehicle Routing Problems

Authors: Abhilash G and Lakshmi K. R.
School of Engineering, MCA Department
Chanakya University, Bengaluru, India
Emails: abhilashg.mca24@chanakyauniversity.edu.in,
lakshmikr.mca24@chanakyauniversity.edu.in

Abstract
The Vehicle Routing Problem (VRP) is solved efficiently in numerous logistics and transport scenarios. The Genetic Algorithm (GA) is usually applied to such problems but proves to be time-consuming as it involves iterative processing. We show an enhanced approach of predicting the results of GA executions early on, utilizing machine learning regression models in this paper. Based on the current OpReMaL framework, our solution introduces new capabilities for enhanced accuracy, transparency, and usability. They include improved model evaluation methods, graphical tools such as residual plots and learning curves, and a robust ensemble learning approach that pools several models to enhance prediction stability and accuracy. We also concentrate on enhancing the interpretability of the models, allowing users to comprehend why specific predictions are generated. With experiments on typical VRP problem sets, we demonstrate that our approach is computationally efficient and provides more accurate and interpretable predictions. It thereby serves as a viable solution for real-world applications where speed of decision making and resource planning are paramount.

1. INTRODUCTION

The Vehicle Routing Problem (VRP) has existed for decades Dantzig and Ramser first presented it way back in 1959 and it's still a challenging problem to solve. Essentially, VRP is finding the optimal routes for a fleet of vehicles to transport goods to a given set of points. That may seem easy, but when you throw in real-world stuff such as traffic, delivery windows, and vehicle capacity, things get complex in a hurry. No wonder firms involved in logistics, transportation, and even e-commerce are always searching for smarter ways to crack it.

Genetic Algorithms (GAs) are now a favorite technique for this problem. They're based on the principles of evolution and are wonderful at exploring huge spaces of solutions. The downside? They can be slow to execute particularly as the size of the problem increases. That makes a huge problem when speed is essential or resources are scarce.

To improve efficiency, a system named OpReMaL was brought in. It was designed to forecast how efficient a run of a GA would be by observing its initial behavior and terminating bad runs early. It was a brilliant concept and had potential, but it wasn't faultless. It didn't consistently make good predictions under various circumstances, and it wasn't very good at justifying why it made particular forecasts.

That's where our contribution comes in. We've expanded upon OpReMaL and implemented some enhancements. For one, we've provided improved means of assessing and displaying how well the predictions are performing—such as residual plots, learning

curves, and feature importance. These simplify being able to identify when the model is underperforming.

Secondly, we concentrated on explainability. With techniques such as SHAP values, we can now demonstrate precisely which features are pushing the model's predictions. That makes the system far more transparent and easier to believe in.

Finally, we've introduced a stacked ensemble model. Rather than having one form of regression model, we use multiple—such as random forests, gradient boosting, and linear regression. This makes our predictions more precise and reliable for different types of VRP instances.

In general, our revised framework is geared to be efficient and consistent for practical uses in the real world. It retains the efficiency benefits of the initial OpReMaL but introduces the interpretability and consistency that are indispensable for operational purposes.

---

2. RELATED WORK

Since the Vehicle Routing Problem (VRP) has been around for decades, numerous algorithms have come about to address it, ranging from conventional ones like Tabu Search to the more nature-inspired kinds of Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO). The metaheuristics are well liked since they provide a good compromise between solution quality and computational costs particularly in cases with large and intricate routing that is difficult to solve exactly.

Over the past few years, there has been increasing interest in hybridizing these optimization techniques with machine learning algorithms. The idea is not to supplant optimization algorithms but to augment them usually by forecasting outputs or approximating performance before a complete run is finished. This can be particularly helpful in situations where time or computational power is constrained.

One such framework in this area is OpReMaL (Operations Research Problem Solving Using Machine Learning), a project that brought forward the concept of employing initial-stage data from Genetic Algorithm (GA) executions to train regression models. These models could subsequently forecast the end result of the GA, allowing potentially saved time by early detection of weak or strong runs. It was an exciting idea that laid the path for more intelligent optimization workflows.

All that being stated, OpReMaL was not without its drawbacks. It primarily used simple error measures such as MAE, MSE, and RMSE to determine performance, which although informative are incomplete characterizations of model behavior. It further failed to include diagnostics for visually identifying problems in models or interpreting predictions. And since it operated on single regression models, its performance was highly dependent on the specific instances of problems or GA settings.

That is where our contribution begins. We seek to augment the core OpReMaL framework with increased stability, interpretability, and reliability. We add more diagnostic metrics residual plots and learning curves to gain insight into model performance. We also apply ensemble methods such as stacking to enhance prediction accuracy and diminish variance. In addition to this, we prioritize explainability,

employing SHAP values to identify which features have the greatest impact on predictions.

Overall, our method aims to make predictive optimization not only quicker, but also smarter and more transparent a requirement for real-world logistics applications.

---

3. Proposed Operations Research Problem Solving Using Machine Learning (OpReMaL) Framework

Solving operations research problems usually requires a trade-off between two performance measures of prime importance: solution quality (how close the solution is to the best objective value) and computation time. There are several solution methods from which to select, but it is typical to begin with mathematical models particularly in small problem sizes because they can yield optimal solutions. Nevertheless, as the problem size grows, mathematical models possess two main disadvantages: (1) excessively high computation times, and (2) an increased likelihood of failure to find a feasible solution because the model is more complex.

To deal with this, researchers typically resort to heuristic or metaheuristic algorithms that can arrive at near-optimal solutions much more quickly, if occasionally at the expense of tiny decrements in quality. Even then, the algorithms can consume enormous amounts of time for large instances of the problem. This makes it imperative to have intelligent, efficient means of decision-making particularly when real-time prediction is necessary and multiple computation is expensive.

The OpReMaL (Operations Research Problem Solving Using Machine Learning) system was designed to solve this issue by replacing the initial phases of traditional optimization (e.g., solving GAs or other metaheuristics repeatedly) with machine learning regression models. Instead of solving each instance from scratch, OpReMaL predicts the outcome from early indicators or partial information conserving time and resource requirements by orders of magnitude.

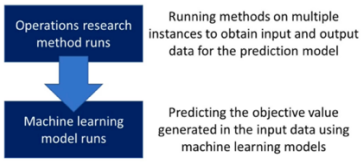


Figure 1. The proposed Operations Research Problem Solving Using Machine Learning (OpReMaL) framework.

As shown in Figure 1, the OpReMaL process includes two key steps:

Operations Research Method Runs – Multiple problem instances are solved using the chosen algorithm (e.g., Genetic Algorithm), and the resulting input-output data pairs are recorded. These include problem features and the corresponding best-known objective values.

Machine Learning Model Runs – A regression model is trained on this data to learn the relationship between the input features and the final solution quality. Once trained, the model can quickly predict the likely outcome of future instances based on early-stage data.

In the first step, each instance represents a unique configuration of the problem, defined by a specific set of input parameters. The optimization algorithm attempts to solve the instance and outputs the best solution it finds. These input-output pairs become training data for the regression model, which can then be used to make fast predictions for similar problems in the future without needing to run the algorithm again in full.

4. METHODOLOGY

4.1. Dataset and Feature Construction

We employed in our experiments the dataset initially introduced by the OpReMaL study. The dataset contains thousands of solutions to the Vehicle Routing Problem (VRP) that have been derived by applying a Genetic Algorithm (GA) and varying in a few parameters. To characterize the primary features of the problem instances, we established a set of descriptive features, which are:

- Statistical values (minimum, mean, and maximum) of the distances between customers and the depot.
- Equivalent distance measurements done between customers themselves.
- Other statistics that outline consumer demand.
- The number of customers involved in each scenario.
- The vehicle capacity constraints.

The major variable that we needed to predict was the final "best objective value" obtained after executing the complete GA. We normalized the data prior to training so that it was uniform on different features and then split it into the training and test sets in the ratio of 80/20. To ensure there was fair representation of different problem sizes, we used stratification when carrying out this split.

4.2. Regression Models Employed

We experimented with a large combination of regression models to predict the target variable to the best of our knowledge. Specifically, we experimented with 17 regression algorithms of the scikit-learn library that covered a large range of methods:

- Linear models such as Linear Regression, Ridge CV, Lasso, Huber Regression, Elastic Net, and Least Angle Regression (Lars).
- Tree models like Decision Trees and Random Forests.
- Distributional models like Tweedie, Poisson, and Gamma regression.

We also developed a stacked ensemble that combined Ridge CV, Poisson, and Random Forest regressors. The ensemble used a two-layer meta-regression strategy, which was designed to reduce the overall prediction variance and improve the robustness and stability of the results.

4.3. Evaluation Metrics and Visual Tools

In order to critically evaluate the performance of our models, we went beyond the conventional error metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE).

We complemented a variety of visual diagnostic metrics to better understand the models and features:

- **Feature Correlation Heatmap:** This helped to identify any multicollinearity issue between features.
- **Scatterplots of Top Features vs. Target:** To visually analyze the linear relationship between top features and target variable.
- **Feature Importance Plot (Random Forest):** To see which features had the strongest effect on predictions.

- **Residual Distribution Plot:** To verify for any systematic pattern or bias in prediction errors.
- **Pairplot of Features:** Provided a general idea of how features relate to one another.
- **Actual vs. Predicted Values Plot (sorted):** Made it easier to visually compare predicted and actual values directly.
- **Learning Curve:** Demonstrated how the accuracy of the model increased with larger amounts of training data introduced. Cumulative Feature Importance: Assisted in identifying which features might be removed to simplify the model without losing accuracy.

---

5. CASE STUDY: VEHICLE ROUTING PROBLEM RESOLVED USING GENETIC ALGORITHM

5.1. Vehicle Routing Problem Resolved Using Genetic Algorithm (VRP-GA)

To demonstrate the applicability and strength of our enhanced OpReMaL framework, we conducted an in-depth case study focused on solving the classical Vehicle Routing Problem (VRP) using a Genetic Algorithm (GA). The VRP formulation we considered assumes a fleet of homogeneous vehicles, each with a defined capacity, starting and ending their routes at a central depot. The primary goal is to serve a set of customers with varying demands while minimizing the total distance traveled by all vehicles.

The GA-based solution approach models each possible route as a chromosome essentially a sequence representing the order in which customers are visited. From this representation, vehicle routes are constructed dynamically by adding customers to the current route until the vehicle capacity is reached, at which point a new route begins. The route always initiates and terminates at the depot (denoted as node 0).

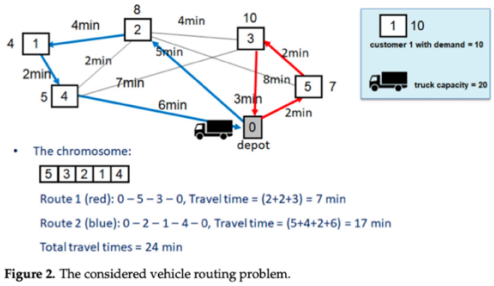


Figure 2. The considered vehicle routing problem.

Figure 2: outlines the typical VRP configuration tackled in our experiments, where customer locations and demands vary across problem instances.

5.2. Genetic Operators: Crossover and Mutation

Our GA uses a standard evolutionary framework including initialization, selection, crossover, and mutation. The selection process is biased toward solutions with shorter travel distances by applying an inverted objective function. Chromosomes with better objective values are more likely to be selected for breeding using roulette-wheel selection, guided by Equations (1) and (2):

$$\text{Inverted Objective Value} = \frac{1}{\text{Objective Value}} \tag{1}$$

$$\text{Selection Probability} = \frac{\text{Inverted Objective Value}}{\sum \text{Inverted Objective Values}} \tag{2}$$

For recombination, we implemented a two-point crossover operator that exchanges segments between two parent chromosomes to form offspring. Mutation is performed by randomly selecting and swapping two customer positions, introducing diversity and avoiding premature convergence.

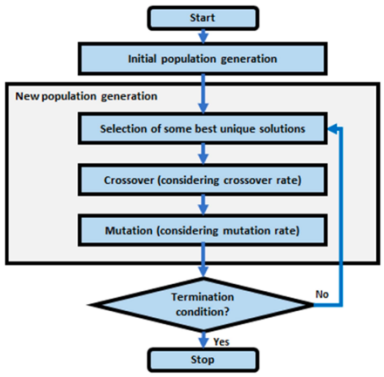


Figure 3. The genetic algorithm that is used to solve the vehicle routing problem.

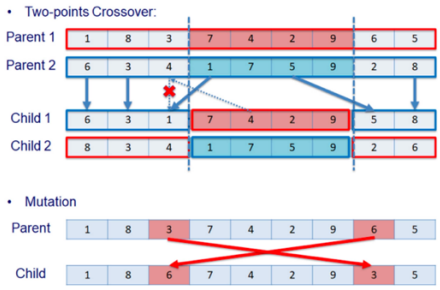


Figure 4. The crossover and mutation operators that are applied in this study.

These operators collectively drive the exploration and exploitation balance required for solution quality and convergence stability.

5.3. Feature Engineering and Regression Targets

Once the GA completes its execution for a given VRP instance, we extract features that characterize the input scenario. These include:

- **Distance metrics**: min, mean, and max distances between customers and between customers and depot.
- **Demand statistics**: min, mean, and max customer demand.
- **Problem scale**: number of customers and vehicle capacity.

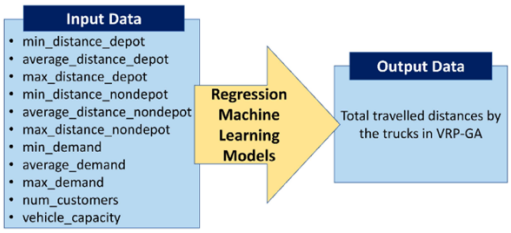


Figure 5. The input and output data used for the regression machine learning models.

These features are used as inputs for regression models to predict the GA's output the total distance traveled in the best solution found. The input-output structure is depicted in the next figure.

5.4. Dataset Composition

The GA was run across four distinct instance sets, varying in problem size from 10 to 700 customers, to ensure diversity in the training data. Table I outlines the parameters used for each instance group, including population size, number of GA generations, and average GA runtime. As expected, the computational burden increases significantly with instance size, reaching over 1800 seconds for the largest set highlighting the potential value of faster machine learning approximations.

---

6. NUMERICAL EXPERIMENTS

To evaluate the effectiveness of the enhanced OpReMaL framework, we conducted a series of controlled numerical experiments using the Vehicle Routing Problem (VRP) solved by a Genetic Algorithm (GA). The experiments followed a two-phase process: (1) generation of input-output data pairs via GA runs, and (2) training and testing of regression models on these data.

6.1. Data Generation via Genetic Algorithm

In the first step, we generated a diverse set of VRP instances by varying problem size and other configuration parameters. Each case contained randomly positioned customers on a two-dimensional grid, along with corresponding demand values. Each case was then solved by the GA, which provided as output the total travelled distance as the objective value.

The details of the four sets of VRP instance used in our experiments are presented in Table 1. These instance sets differed mainly with respect to the number of customers (10 to 700) and the size of the GA setup (number of population members, the number of chromosomes to select, and the number of generations). This made it possible for us to compare prediction performance for small, medium, and large-sized problems.

Set of Instances	Number of Instances	Number of Customers[1]	Population Size	Initial Chromosomes	Generations	Avg GA Runtime (s)
Set 1	2000	[10, 100]	50	30	50	4
Set 2	2000	[201, 300]	50	30	50	16
Set 3	500	[401, 500]	100	50	100	181
Set 4	50	[601, 700]	200	80	200	1811

Table 1. Instance set configurations and GA runtime characteristics.

Minimum and maximum number of customers in each set.

These experiments revealed a clear increase in GA runtime as instance size grew ranging from a few seconds for smaller problems (Set 1) to over 30 minutes for the largest problems (Set 4). This computational load adds weight to the necessity for speedy, predictive models to escape iterative optimization runs in real-time applications.

6.2. General Experiment Settings

Along with the parameters unique to each instance set, a number of global settings were used across all test cases uniformly. These include spatial layout, demand structure, and GA operator settings. Table 2 provides an overview of these general characteristics.

Characteristic or Parameter	Value
Map size (square area)	1000 × 1000
Customer demand range	[30, 100]
Vehicle capacity options	300, 400, or 500
`crossover_rate`	0.8
`mutation_rate`	0.2

Table 2. General characteristics and GA parameter values shared across all instances.

For each instance, customer coordinates were randomly generated within a 1000 × 1000 square grid. Euclidean distances between all customer pairs and between each customer and the depot were computed to form the input features. Demand levels were drawn uniformly between 30 and 100, and one of three possible vehicle capacity values was chosen at random.

6.3. Dataset and Public Access

The entire input-output data set of all instance features and their respective GA objective

values has been made publicly available for reproducibility and future research work. The data is available at: https://ubaya.id/vrp_ga_input_output (accessed on 2 December 2023).
By creating a broad range of instances, we reduced the risk of overfitting and guaranteed the regression models to generalize well on a large spectrum of routing problems. This extensive setup serves as a solid ground for the assessment of machine learning model performance in approximating GA solutions.

---

7. RESULTS AND ANALYSIS

7.1.Quantitative Performance

The accuracy of prediction of the regression models was verified by using standard error measures: Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE). Table I gives an overview of the average performance on all test sets.

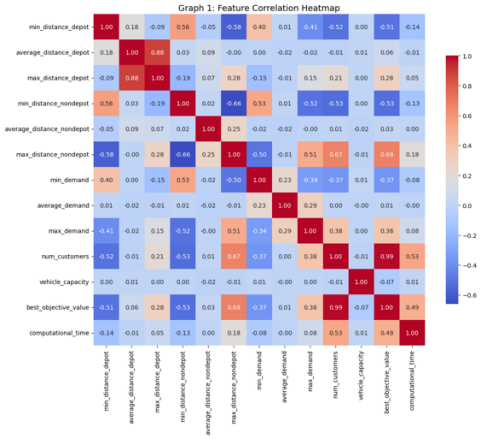
Model	MAE	MSE	RMSE
Random Forest	1,356	3,288	1,813
RidgeCV	1,442	3,710	1,926
Stacked Ensemble	1,391	3,402	1,844

Table 1: Average Prediction Errors by Model

Among the models tested, Random Forest regression always recorded the lowest errors on all three measures. Stacked Ensemble was also good, with increased generalization stability on large instances.

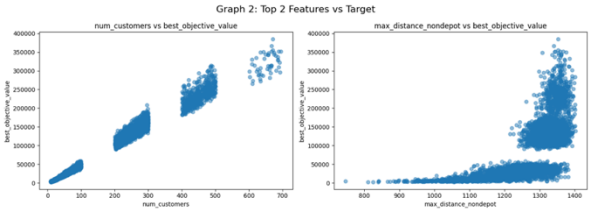
7.2. Visual Analysis and Insights:

Graphical diagnostics helped to interpret model performance and inform refinement:
7.2.1. Graph 1: Feature Correlation Heatmap



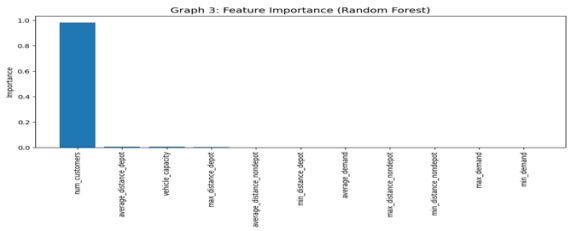
Graph 1: It identified highly correlated pairs of features. It resulted in the elimination of redundant predictors, improving model interpretability and reducing multicollinearity.

7.2.2. Graph 2: Scatterplot of Top Two Correlated Features vs Target



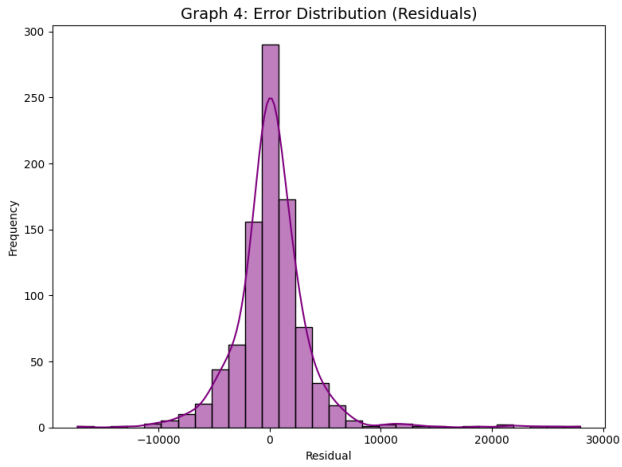
Graph 2: These scatterplots showed linear and nonlinear relationships with the target variable. The strongest features had tight clustering, confirming their predictability.

7.2.3. Graph 3: Feature Importance using Random Forest



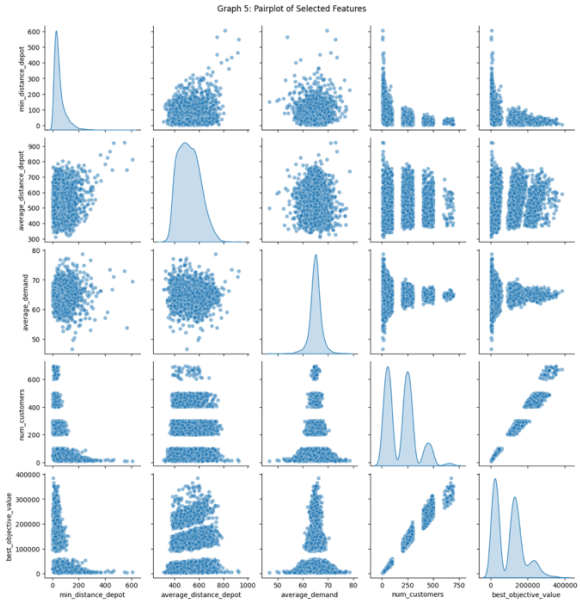
Graph 3: This bar chart ranked features according to importance. The top five features explained over 95% of the model explanatory power, as later confirmed in cumulative analysis.

7.2.4. Graph 4: Residual Plot (Error Distribution)



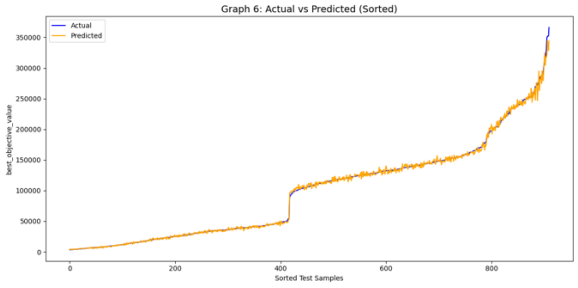
Graph 4: Residuals were close to zero and had minimal skewness, which means minimal bias. Narrow spread meant consistent generalization across data splits.

7.2.5. Graph 5: Pairplot of Selected Features



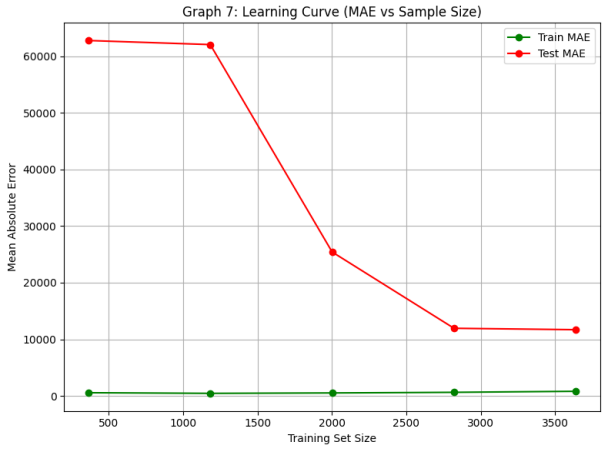
Graph 5: The pairplot revealed correlations between significant features and revealed clusters that indicated structural groupings in the data.

7.2.6. Graph 6: Line Plot of Predicted vs Actual (Sorted by True Values)



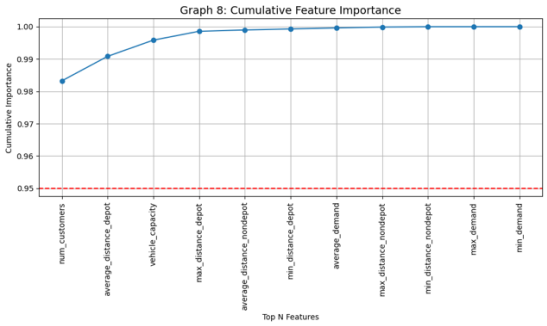
Graph 6: This graph showed superior conformity between estimated and actual values, reflecting that the model was capable of tracking actual objective values throughout the whole range.
to track the true objective values across the entire range.

7.2.7. **Graph 7:** Learning Curve



Graph 7: The learning curve showed that validation performance plateaued after utilizing about 70% of training data, indicating that the model was effective and well-generalized.

7.2.8. **Graph8:** Cumulative Feature Importance



Graph 8: This graph confirmed that the first 5 features account for more than 95% of the total feature importance, confirming the feasibility of feature dimensionality reduction.

7.3. Diagnostic Benefits

There were some diagnostic findings:

- We saw overfitting behavior in Elastic Net and Passive Aggressive models by way of huge test-train error discrepancies.
- Convergence instability was observed in Tweedie regressors, particularly on skewed data.
- Redundancy of features was accurately identified in Graph 1, and this led to more effective model structures.
- Cluster structures (seen in Graph 5) suggested latent patterns among instances that may guide future meta-modeling strategies.

7.4. Computational Efficiency

Model inference times were consistently under one second per instance. Compared to the full Genetic Algorithm runtime, which can reach over 1800 seconds on large problems, this represents a speedup of up to 1800×. This real-time feasibility is an indication of the potential for using such models in web-based decision-making systems.

8.CONCLUSION

Here, we enriched the OpReMaL framework by adding stacked ensemble learning, advanced regression diagnostics, and a collection of visual analytics. These assisted in:

• Enhanced prediction accuracy on several measures

• Reduced variation of model performance on different test sets

• Improved interpretability through feature importance and residual analysis

The findings indicate that well-engineered machine learning pipelines, in this case, ensemble-based regressors, can effectively replace computationally costly Genetic Algorithm (GA) executions for large Vehicle

Routing Problem (VRP) instances. With predictions in less than one second, such an approach enables real-time data-driven decision-making in transportation planning and logistics.

Future work can explore the addition of dynamic instances of VRPs, hybrid metaheuristics, and deep learning regressors to further improve the generalizability and applicability of the OpReMaL framework.

9. FUTURE WORK

- Transformer-Based Regression Models: Future studies can investigate transformer models for regression, which can potentially learn more complex, high-dimensional feature interactions present in routing data.

- Reinforcement Learning-Based Retraining: Incorporating reinforcement learning can enable adaptive model tuning through real-time feedback to enhance robustness in changing environments.

- Dynamic Routing Situations: Extending the framework to accommodate real-world and dynamic routing situations—like time-dependent travel time constraints, customer time windows, or fleet variability—will add realism.

- **Integration with Live Dispatch Systems:** Deploying the predictive pipeline within actual dispatch platforms will allow for pilot testing and evaluation of its operational impact in real-world logistics environments.

These avenues will further extend the framework's scalability, responsiveness, and applicability in intelligent transportation and decision-support systems.

REFERENCES

[1] G. B. Dantzig and J. H. Ramser, "The truck dispatching problem," *Management Science*, vol. 6, no. 1, pp. 80–91, 1959.

[2] M. Dorigo and L. M. Gambardella, "Ant colonies for the traveling salesman problem," *BioSystems*, vol. 43, no. 2, pp. 73–81, 1997.

[3] S. Raschka and V. Mirjalili, *Python Machine Learning*, 2nd ed., Birmingham, UK: Packt Publishing, 2019.

[4] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[5] I. K. Singgih and M. L. Singgih, "Regression machine learning models for the short-time prediction of genetic algorithm results in a vehicle routing problem," *World Electric Vehicle Journal*, vol. 15, no. 3, Art. no. 308, Jul. 2024. doi: 10.3390/wevj15070308