CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Vehicle Design Optimization Using AI/ML Methods

Powertrain parameter optimization using surrogate model

Master's thesis in Computer science and engineering

Puthige Venugopal Chetan Acharya

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2025

Master's thesis 2025

Vehicle Design Optimization Using AI/ML Methods

Powertrain parameter optimization using surrogate model

PUTHIGE VENUGOPAL CHETAN ACHARYA

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2025

Vehicle Design Optimization Using AI/ML Methods
Powertrain parameter optimization using surrogate model
PUTHIGE VENUGOPAL CHETAN ACHARYA

Supervisor: Peter Damaschke, Department of Computer Science and Engineering, Chalmers University of Technology
Advisors: Yu Xu, Prajwal Bharadwaj and Xiaojuan Wang, Energy Dynamics Team, Zeekr Technology Europe
Examiner: Marina Axelson-Fisk, Department of Mathematical Sciences, Chalmers University of Technology

Master's Thesis 2025
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Gothenburg, Sweden 2025

Vehicle Design Optimization Using AI/ML Methods
Powertrain parameter optimization using surrogate model
PUTHIGE VENUGOPAL CHETAN ACHARYA
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Energy efficiency and performance are important attributes when developing and designing vehicles. With the transition of the automotive industry towards energy efficiency and sustainability, it is ever more important to save computational resources. Traditional vehicle design optimization heavily relies on computationally expensive simulations. The simulations carried out in this project particularly focus on the powertrain of vehicles. This work focuses on developing a surrogate-assisted, multi-objective optimization framework that efficiently finds the optimal values for the given variables. Surrogate modeling is an engineering method used when the outcome of an experiment cannot be easily computed, so a mathematical approximation is applied. In this case, we use machine learning models to predict the outcome of expensive simulations. These trained model(s) is then used for optimization instead of running optimization on the simulation model directly. First, we generate 4096 Sobol-sampled configurations spanning different parameters like gear ratios and electric motors. We train and compare different surrogate models like Random Forest, XGBoost, and LightGBM on these data, achieving test R^2 scores up to 0.96 with Random Forest. Next, we employ the Non-Dominated Sorting Genetic Algorithm II (NSGA-II) to explore trade-offs among various conflicting objectives, extracting a Pareto front of optimal designs. A weighted-sum post-processing step or a constrained method later selects a single best-trade-off configuration, which full simulation validates. This framework slashes computational cost and empowers rapid, data-driven vehicle powertrain design.

Keywords: Machine Learning, Surrogate modeling, Multi-objective optimization, Simulation, Sampling techniques.

Acknowledgements

I would like to express my deepest gratitude to Yu Xu, Prajwal Bharadwaj, and Xiaojuan Wang of the Energy Dynamics team at Zeekr for their constant support and invaluable technical assistance throughout this research. Their expertise and thoughtful guidance have been instrumental in shaping this work.

My sincere thanks go to Peter Damaschke for his patient supervision, insightful feedback, and encouragement at every stage of the thesis. I am also grateful to Marina Axelson-Fisk for serving as my examiner and for her constructive critiques that have greatly improved the clarity and rigor of this work.

I would also like to thank my friends and family for their unwavering encouragement, understanding, and motivation throughout the entire process of writing this thesis. Their support has made this journey possible.

Finally, I am thankful for the many friendships I formed at Zeekr Technology Europe, and for the warm and collaborative environment that enriched my thesis experience.

Puthige Venugopal Chetan Acharya, Gothenburg, 2025-09-01

Contents

List of Figures

List of Tables

1

Introduction

The automotive industry is evolving ever so rapidly with increasingly stringent emissions regulations, rising consumer demand for performance and range, and the ever-present imperative to reduce development time and cost. In order to meet these requirements, several innovations have been made in the automotive field, especially in the powertrain design. Vehicle powertrains have grown vastly more complex over the past two decades, incorporating multiple energy sources, high-power electric machines, sophisticated transmission architectures, and advanced control electronics. While high-fidelity physics-based simulation tools (e.g., Hardware in the Loop test benches or detailed multi-body vehicle dynamics models) remain the gold standard for predicting real-world performance, their sheer computational expense severely limits their use in global design optimization, where hundreds of design variants must be evaluated. This project addresses this challenge by combining data-driven surrogate models with multi-objective optimization algorithms, enabling rapid exploration of the full design space without sacrificing accuracy.

1.1 Evolution of Vehicle Powertrains

Historically, vehicles solely relied on an internal combustion engine (ICE) along with a transmission system to drive the wheels. In recent developments, hybrid electric vehicles (HEVs), plug-in electric hybrids (PHEVs), and electric vehicles(BEVs) have introduced one or more electric motors and generators into the powertrain in series, parallel, or power split architectures. In a series hybrid, the ICE drives a generator to charge a battery or directly power an electric traction motor. In a parallel hybrid, the ICE and motor each contribute torque through a transmission to the wheels. Series-parallel (or power-split) hybrids are a combination of both series and parallel architectures. PHEVs extend this concept further by making use of a larger battery that can be charged from the electric grid. This enables an extended electric-only driving range before the ICE kicks in. Finally, BEVs eliminate the ICE entirely. BEVs rely on one or more electric motors and a high-capacity battery. BEVs achieve zero tailpipe emissions but depend on charging infrastructure and larger battery packs to address range concerns. (Ehsani et al., 2010) [1] (Emadi, 2017) [2].

Each architecture carries its own trade-offs between complexity, energy efficiency, peak power capability, and control strategy flexibility. Moreover, within each architecture, the sizing and selection of each physical component, the internal combustion engine

displacement, electric machine power and torque ratings, battery capacity, and transmission gear ratios critically determine the vehicle's fuel economy, acceleration performance, emissions, manufacturability, and cost. Figure 1.1 shows the different types of powertrain architectures.

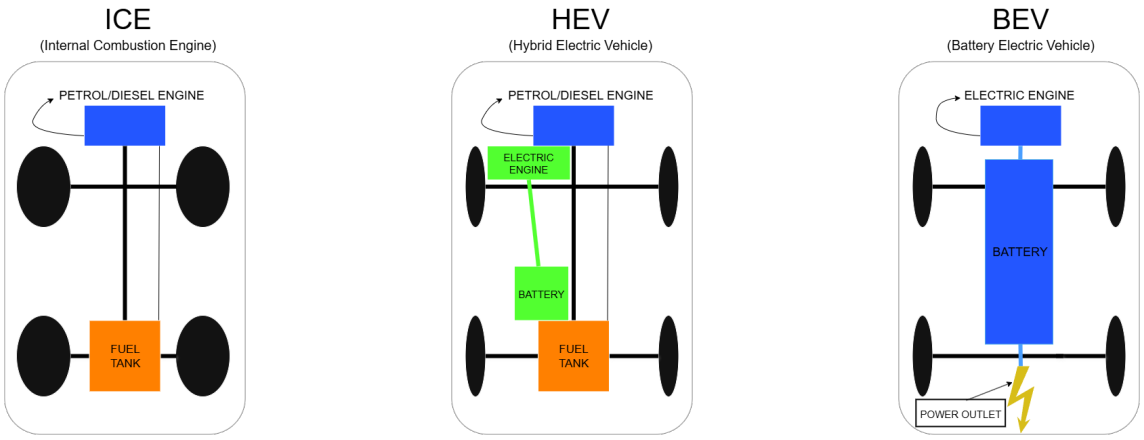


Figure 1.1: Types of powertrain architectures

1.2 The Role of Components

Each component involved in the powertrain design has its own significance. Tuning these components to the right value drastically affects various aspects of the vehicle's performance and efficiency. This section will be the discussion of some of the main components that are varied and optimized using the framework we developed in this work.

1.2.1 Gear ratios

A gear ratio quantifies how rotational speed and torque are transformed between two shafts connected by gears. If the gear on the input (driving) shaft has N_{input} teeth and meshes with a gear on the output (driven) shaft with N_{output} teeth, then the gear ratio g is given by,

$$g = \frac{\omega_{\text{input}}}{\omega_{\text{output}}} = \frac{N_{\text{output}}}{N_{\text{input}}}, \tag{1.1}$$

here ω denotes angular velocity.

In a conventional six speed gearbox, each gear is a fixed pair of its own gear ratio. For example, $g_1 = 3.6{:}1$, $g_2 = 2.1{:}1$,..., $g_6 = 0.8{:}1$, where lower gear ratios like g_1 multiply engine torque at the wheels, enabling rapid acceleration from a standstill but limiting top speed. Whereas, higher gear ratios allow high cruising speeds with low engine Revolutions Per Minute (RPMs), which in turn gives better fuel economy.

In a power-split hybrid, multiple gearsets are used to allocate torque among the ICE, generator (P3), and traction motor (P4). A simple planetary gearset consists of a

sun gear, planet gears (mounted on a carrier), and a ring gear. By fixing or rotating different elements, one can derive relationships among the three shafts. For instance, if the sun gear (connected to the ICE) spins at ω_s, the planet carrier (connected to the vehicle output) spins at ω_c, and the ring gear (connected to the generator) spins at ω_r, the kinematic constraint is

$$\omega_s \; + \; \frac{N_s}{N_r}\,\omega_r \; = \; \left(1 + \frac{N_s}{N_r}\right)\omega_c, \tag{1.2}$$

where N_s and N_r are the tooth counts of sun and ring, respectively (Gleeson, 2012)[5]. By choosing the ratio N_s:N_r, the designer fixes how engine speed is split between mechanical power to the wheels (ω_c) and electrical power to the generator (ω_r). In multi-stage designs (e.g, P3, P4 gearsets), additional planetary ratios govern how the traction motor interacts, creating a high-dimensional decision space. Selecting these gear ratios improperly can force the ICE or electric machines to operate outside their optimal efficiency regions, which might end up degrading fuel economy or diminishing performance and even result in increasing emissions.

1.2.2 Electric motors

Engineers have to decide among dozens of candidate motors differing in peak power (e.g., 160 kW, 260 kW), continuous power rating, torque density, and thermal limits, which determine both dynamic performance and efficiency across the driving cycle. With 24 different front/rear motor combinations under consideration, the combinatorial explosion of possible powertrain architectures becomes immense. Each choice, in combination with discrete gear-ratio selections, multiplies the number of possible architectures. For example, pairing a 160 kW front motor with a 260 kW rear motor yields different performance and cost than the reverse pairing; combining either pairing with slight adjustments to g_{ICE}, g_{P3} and g_{P4} further expands the design space. This combinatorial explosion makes exhaustive evaluation via high-fidelity simulation impossible within a reasonable time or computational budget.

1.3 Challenges of High-Fidelity Simulation and Design Exploration

Typically, powertrain design is explored by using simulation tools like Siemens Amsim, IPG carmaker, and GT-SUITE Simulink. The real-world scenario is simulated using these tools, and all potential options are explored. A single simulation can take minutes to hours or even days, depending on the level of complexity, meaning that a brute-force grid search with tens of thousands of points quickly becomes intractable.

Moreover, powertrain design is inherently multi-objective: minimizing fuel consumption often conflicts with maximizing 0–100 km/h acceleration. Classical gradient-based methods struggle here because many objectives are nonconvex and noisy, the search space is mixed continuous (gear ratios) and discrete (motor selection), and gradients are unavailable or unreliable for black box simulators.

1.4 Surrogate Modeling: Principles and Approaches

In order to overcome these computational bottlenecks, surrogate models are used, which approximate the mappings from design variable parameters to performance metrics. Simulated data is used as training data for these models.

Let $\mathcal{X} \subset \mathbb{R}^d$ denote the design space (gear ratios and motor choice) and $\mathcal{Y} \subset \mathbb{R}^m$ the space of outputs (efficiency and performance metrics). The expensive simulator defines a mapping $f : \mathcal{X} \to \mathcal{Y}$. For any $x \in \mathcal{X}$ we write $y = f(x) \in \mathcal{Y}$. Given data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ with $y_i = f(x_i)$, we train a surrogate $\hat{f} : \mathcal{X} \to \mathcal{Y}$ so that $\hat{f}(x) \approx f(x)$ for unseen x, and use $\hat{f}(x)$ in place of $f(x)$ during optimization.

Different machine learning models are compared, and the best-performing model for each task is chosen to be used in the optimization task. Random forest, Ensemble models like XGBoost, LightGBM, Feed forward Neural Network are some of the models which has been explored within this work. Validation of the surrogate model is done via cross-validation (R^2, MSE), which ensures low bias/variance before trusting it in optimization loops. Once a surrogate achieves sufficiently high fidelity (here, test $R^2 > 0.80$ on key metrics), it can replace the full simulator inside the NSGA-II optimizer, which reduces the computation time immensely.

Introduced by Breiman (2001) [3], a Random Forest (RF) is an ensemble of T decision trees. Each tree is trained on a bootstrap sample of the data, and at each split node, a random subset of features is considered to determine the best split. For regression tasks, the RF prediction for output y_k is

$$y_k^1(x) \;=\; 1/T \sum_{t=1}^{T} h_{t,k}(x), \tag{1.3}$$

where $h_{t,k}$ is the $k-th$ output from the $t-th$ tree. Random Forests naturally handle mixed continuous and categorical variables, are robust to outliers, and require minimal hyperparameter tuning beyond the number of trees and maximum depth of the tree.

XGBoost (Chen & Guestrin, 2016) [4] and LightGBM (Ke et al., 2017) [5] are the two Gradient Boosted Decision Trees (GBDT) that have been implemented. Unlike RF, where trees are trained independently on bootstrapped data, GBDT builds trees sequentially to correct the errors of the previous ensemble. GBDT often achieves state-of-the-art performance on tabular data. Key hyperparameters include the number of trees, maximum tree depth, learning rate, and subsampling fraction for row or feature sampling.

Ridge (Hoerl & Kennard, 1970) [6] is a linear baseline that reduces overfitting by penalizing large coefficients. Intuitively, it prefers models that explain the data well without relying on extreme parameter values, which makes it stable when features are correlated. We standardize numeric inputs and one-hot encode the motor ID, then fit one linear model per target via a multi-output wrapper. The single key hyperparameter is the L2 regularization strength α, which we tune by cross-validation.

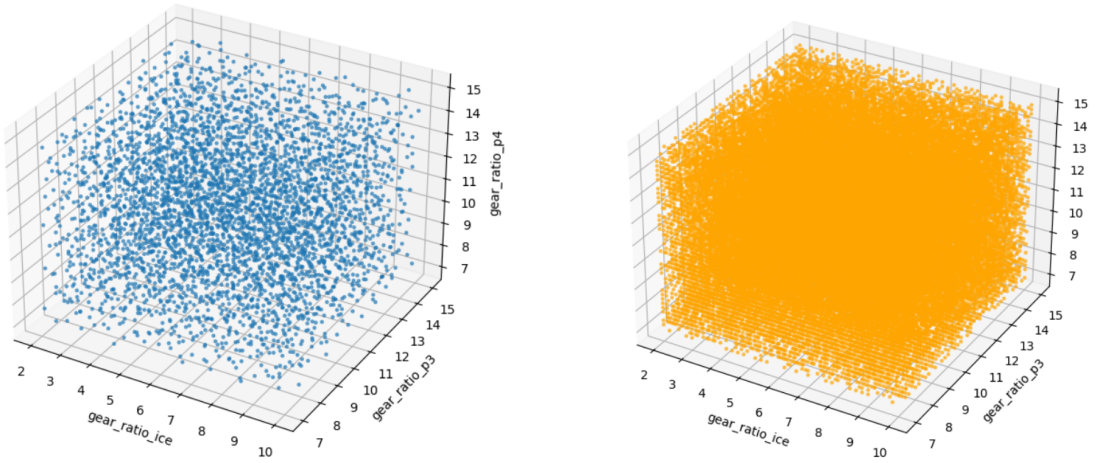
Ridge serves as a fast, interpretable, low-variance reference that quantifies how much nonlinearity the tree/boosting and kernel methods add.

SVR (Vapnik, 1995; Drucker et al., 1997; Smola & Schölkopf, 2004) [7]–[9] seeks a function that stays within a small "tolerance tube" around the data and only penalizes points that lie outside that tube. Using the radial basis function (RBF) kernel lets SVR model smooth nonlinear relationships in tabular data. We standardize numeric features and one-hot encode the motor; the key hyperparameters are the regularization strength C (fit complexity), the tube width ϵ (insensitivity to small errors), and the kernel scale γ (how quickly similarity decays). These are selected by cross-validation. In our results, SVR consistently outperforms the linear Ridge baseline while remaining more conservative than boosted trees, which is useful for sanity-checking surrogate behavior.

For each surrogate type, hyperparameter optimization is performed via k-fold cross-validation on the training partition. The surrogate achieving the highest average R^2 on the test set is designated as the "best performance surrogate" or "best efficiency surrogate" accordingly (Forrester et al., 2008) [10].

1.5 Design of Experiments: Sobol Quasi-Random Sampling

Choosing an efficient sampling method is one of the most crucial aspects of this thesis work, as a good sampling technique greatly reduces the time and resources required to run the simulations in order to gather sufficient data (Davis, 2018) [11]. The straightforward method is to do a full factorial grid search, which covers all the possible combinations of the variables. But, this is rather quite expensive, as for just a few variables, the required amount of data points is close to 44000 just for a few gear ratio components, and would explode exponentially as more components are added into the mix. To overcome this challenge Sobol sampling technique has been implemented. Sobol quasi-random sampling generates low discrepancy sequences that more uniformly fill a multi-dimensional space without the regular alignment artifacts of a grid (Sobol, 1967) [12]. By choosing a power-of-two sample size (e.g., 4096), Sobol ensures good space-filling properties with far fewer total runs. In this application, there are 4 dimensions, three dimensions correspond to continuous gear ratios, and the fourth dimension serves as a continuous surrogate for the discrete motor index. The resulting dataset of 4096 samples is split into training (80%) and test (20%) partitions for surrogate development. Using Sobol sampling rather than random or purely grid-based sampling helps to achieve more uniform "space filling" coverage, reducing the risk of large unsampled regions that could lead to poor surrogate generalization as seen in Figure 1.2 (Forrester et al., 2008) [10].



(a) Sobol-sampled (reduced) set.

(b) Original (dense) set.

Figure 1.2: Reduction of data points using Sobol sampling: (a) reduced set vs. (b) original set.

1.6 Multi-Objective Optimization with NSGA-II

With surrogates in place, Non-Dominated Sorting Genetic Algorithm II (NSGA-II) is deployed to search for Pareto optimal trade-offs among chosen objectives (Bhattacharjee, 2019) [13]. Because no single design simultaneously minimizes all objectives, it is better to have the Pareto-optimal set of solutions. NSGA-II maintains a diverse population via:

- Non-dominated sorting, ranking individuals by Pareto front membership.
- A crowding distance metric to preserve solution spread.
- Standard genetic operators (blend crossover, Gaussian mutation) on continuous and discrete parameters.

After around 50 generations, NSGA-II yields a Pareto front of a few dozen candidate designs. Then, a user-defined weighted sum is applied to collapse the multi-objective front into a single "best-trade-off" solution, which is finally confirmed with one high-fidelity simulation.

Sometimes the user might want to optimize a single primary objective (e.g., minimize fuel consumption) subject to constraints on others (e.g., 0–100 km/h time <= 7 s). To accommodate this, there is an added penalty function. This function adds a huge penalty to the evaluation score if any of the constraints are not matched, and hence, the selected point is dominated by other points that satisfy the constraints.

2

Methods

This section describes how data are gathered and processed to build surrogate models, how those surrogates are validated, how the design of experiments is conducted, and how multi-objective optimization is performed.

2.1 Data Collection and Preprocessing

The starting point for surrogate model training is a collection of archived simulation results produced by a high-fidelity vehicle simulation, in our case, Automaker, a simulation platform built upon a Simulink model. Each simulation input comprises a set of four input variables, three continuous gear ratios and one discrete motor choice and outputs a set of performance and efficiency metrics such as, the time required to accelerate from 0 to 100 km/h, peak acceleration, and top speed, fuel consumption (in L/100 km) and battery energy usage (in kWh) for a specified test cycle, as seen in the Table 2.1.

Since the raw data were stored in spreadsheet form (in our case, Excel files), preprocessing is required before any modeling. First, any textual entries, such as cells with commas instead of periods in numeric values or cells marked "NA" indicating missing data, are converted so that all entries become real numbers. Any row containing an invalid or missing performance output is discarded. Most of the input and output data are in numerical format with occasional appearance of categorical data type (ex, motor type). This categorical data is converted to a one-hot encoded vector. All the continuous values are normalized to zero mean and unit variance over the training set. Normalization is crucial to prevent any one variable from dominating due to scale differences, especially in algorithms such as neural networks that are sensitive to feature magnitudes (Goodfellow et al., 2016) [14].

After cleaning and encoding, the remaining dataset is randomly partitioned into a training set (80% of the data) and a held-out test set (20%). This 80/20 split is chosen to ensure that the surrogate models are validated on data they have never seen, providing an unbiased estimate of predictive accuracy (Kohavi, 1995) [15].

The result of preprocessing is two datasets:

1. Performance data: A set consisting of input as the three scaled gear ratios plus the one-hot motor vector, and output is the eight performance outputs, like 0–100 km/h time, peak acceleration, top speed, etc.

2. Efficiency data: The input remains the same, the output contains six efficiency-related outputs like battery energy, fuel consumption, raw and corrected, etc.

Preprocessing thus ensures that both continuous and categorical design variables are presented in a form suitable for regression modeling, and that validation on a held-out set will accurately reflect generalization performance.

Input	**Outputs**	
	Efficiency	**Performance**
Gear Ratio ICE	Energy from HV battery (kWh)	Time(s) for 0–100 km/h
Gear Ratio P3	Fuel Correction	Acc at 0.3 s
Gear Ratio P4	Fuel consumption corrected with ATCT factor (L/100 km)	Acc at 1 s
Electric Motors	Fuel consumption (raw) from CarMaker (L/100 km)	Peak acceleration (m/s^2)
		Time to peak acc (s)
		Vehicle speed at end (km/h)
		Vehicle acc at end (m/s^2)
		Top speed (km/h)
		Time for 60–120 km/h
		Time for 0–200 km/h

Table 2.1: Input and output groups

2.2 Surrogate Model Development

Because each high-fidelity simulation point can take several minutes or longer to compute, even on a powerful workstation, it is infeasible to search thousands of candidate designs exhaustively. Instead, surrogate modeling is employed, in which a fast statistical or machine learning model is trained to approximate the expensive simulator outputs. Two separate surrogates are built, one to predict performance metrics and one to predict efficiency metrics. Each surrogate is trained on 80% of the available data and validated on the remaining 20%.

Several candidate models are selected to compare and select the best-fit model for each dataset (efficiency and performance). The models compared in this work are Ridge, Random Forest, XGBoost, LightGBM, and Support Vector Regression (SVR). During training, grid search helps to identify the best performing combination of hyperparameters. 5 fold cross-validation on the training set is used to identify the best parameters.

After fitting each candidate model on the training partition with its respective hyperparameters (selected via cross-validation), evaluation is held out test set. Based on the combination of R^2 and Root mean square error values best model for each dataset is selected.

2.3 Design of Experiments: Sobol Quasi-Random Sampling

A Sobol sequence is a deterministic sequence of points $u_i{}_{i=1}^{M}$ in the unit hypercube $[0,1]^d$ (with $d = 4$ in our case, representing three continuous variables plus one continuous surrogate for the discrete motor).

Each Sobol vector $u_i = \Big(u_{i,1},\, u_{i,2},\, u_{i,3},\, u_{i,4}\Big)$ is mapped to actual design variables as follows. The first three components are linearly scaled to the valid ranges of gear ratios:

$$
\begin{aligned}
g_{ICE}^{(i)} &= g_{ICE,\min} + u_{i,1}\Big(g_{ICE,\max} - g_{ICE,\min}\Big),\\
g_{P3}^{(i)} &= g_{P3,\min} + u_{i,2}\Big(g_{P3,\max} - g_{P3,\min}\Big),\\
g_{P4}^{(i)} &= g_{P4,\min} + u_{i,3}\Big(g_{P4,\max} - g_{P4,\min}\Big).
\end{aligned}
$$

where $g_{ICE,min} = 2.0$, $g_{ICE,max} = 10.0$, $g_{P3,min} = 7.0$, $g_{P4,mmax} = 15.0$ and $g_{P3,min} = 7.0$, $g_{P3,max} = 15.0$. The fourth component $u_{i,4} \in [0,1)$ is transformed into an integer motor index by

$$
m_{\text{idx}}^{(i)} \;=\; \lfloor u_{i,4}\, N \rfloor\,,
$$

where N is the total number of motors in the catalog. If $\lfloor u_{i,4}\, N \rfloor$ equals N (which happens only if $u_{i,4} = 1$, but Sobol points lie in [0,1) by definition), it is clamped to $N - 1$. In this way, each Sobol point corresponds to a unique design $z^{(i)} = \Big(g_{ICE}^{(i)},\, g_{P3}^{(i)},\, g_{P4}^{(i)},\, m_{\text{idx}}^{(i)}\Big)$. Because a Sobol sequence of length M= 4096 (a power of two) exhibits especially uniform coverage in four dimensions, M= 4096 such designs are generated. Each design is then simulated with the high-fidelity model to obtain both performance and efficiency outputs. Thus, the training data for efficiency and performance are constructed from these 4096 points. By using Sobol sampling rather than a full factorial grid (which would have millions of combinations if gear ratios were discretized finely), this achieves nearly uniform coverage of the design space with far fewer simulations (Forrester et al., 2008) [10].

Learning curves constructed (Figure 2.1 - 2.2) by subsampling the Sobol DOE ($\approx 500, 1000, 2000, 3300$ points) show rapid accuracy gains up to ~ 2000 samples, followed by diminishing returns. The efficiency surrogate reaches $R^2 \approx 0.95$ at ~ 2000 and $R^2 \approx 0.965$ at ~ 3300; the performance surrogate reaches $R^2 \approx 0.82$ at ~ 2000 and $R^2 \approx 0.84$ at ~ 3300. The concave shape with a knee around 1,000–2,000 justifies our final DOE of 4,096 points as near-optimal in sample efficiency: further doubling the dataset would yield only marginal (< 0.02) improvements in R^2 while doubling simulation cost.

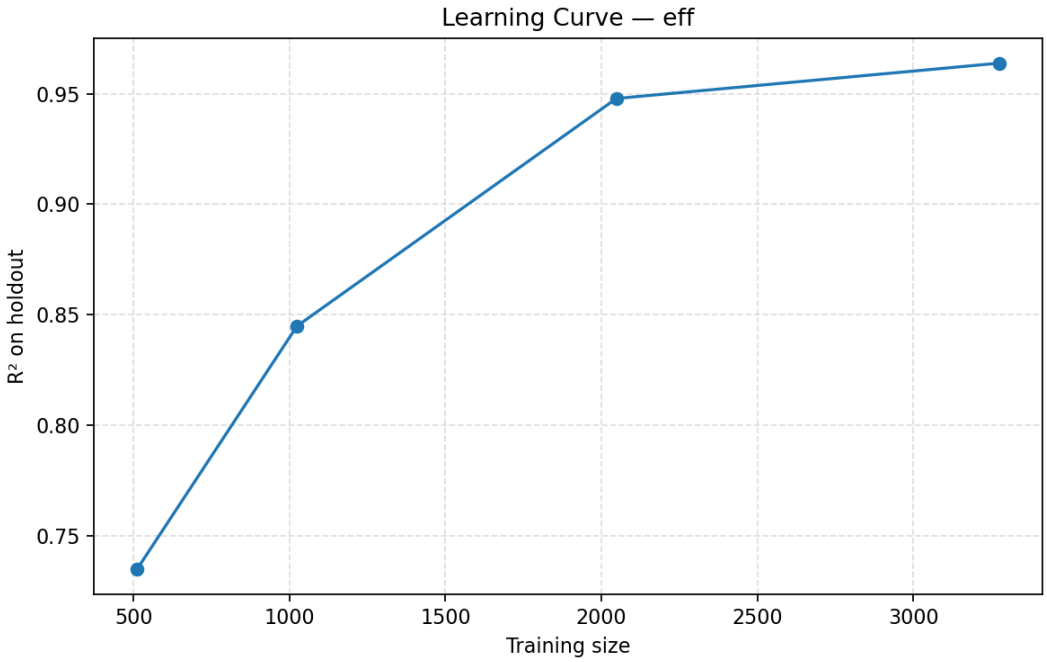


Figure 2.1: Learning curve for efficiency dataset.

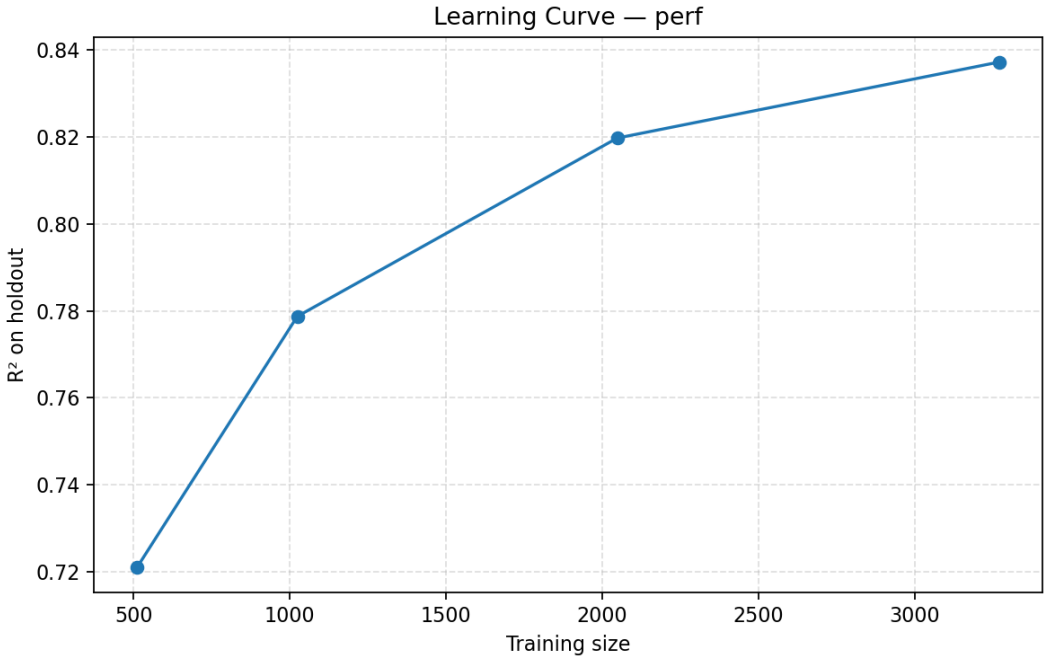


Figure 2.2: Learning curve for performance dataset.

2.4 Multi-Objective Optimization via NSGA-II

Once accurate surrogate models are in place, it is used to search the design space for Pareto optimal trade-offs among multiple objectives. Because each surrogate

evaluation is extremely fast, this helps in affording to explore tens or hundreds of thousands of candidate designs. Non-Dominated Sorting Genetic Algorithm II (NSGA-II) (Deb et al., 2002) [16] is adopted, which is well known for finding diverse approximations of the Pareto front in multi-objective problems.

Let a candidate design be denoted

$$z = \Big(g_{ICE},\, g_{P3},\, g_{P4},\, m_{\text{idx}}\Big),$$

with z belonging to the feasible set

$$Z = [2.0, 10.0] \times [7.0, 15.0] \times [7.0, 15.0] \times \{0, 1, \ldots, N-1\}.$$

Define K objective functions $\{f_k(z)\}_{k=1}^{K}$, where each $f_k(z)$ is predicted by either the performance or efficiency surrogate. A common choice is $f_1(z)$ = predicted fuel consumption (L/100 km, to be minimized), $f_2(z)$ = predicted 0–100 km/h time (to be minimized), $f_3(z)$ = an estimate of total powertrain cost (to be minimized). If $K > 3$, additional objectives might include predicted battery energy usage or top-speed penalty (defined such that minimizing negative top speed is equivalent to maximizing actual top speed).

A design z_a is said to dominate another design z_b if, for every objective k, $f_k(z_a) \leq f_k(z_b)$ and there exists at least one index k for which $f_k(z_a) < f_k(z_b)$. A Pareto-optimal set $P^* \subset Z$ contains designs such that none of them is dominated by any other design in Z. Because it is difficult to enumerate Z exhaustively, NSGA-II approximates P^* by evolving a population of candidate solutions over multiple generations.

The NSGA-II algorithm proceeds as follows:

1. Initialization. A population of size μ (e.g., $\mu = 100$) is initialized by sampling each continuous gene g_{ICE}, g_{P3}, g_{P4} uniformly within its allowed interval and sampling m_{idx} uniformly from $\{0, \ldots, N-1\}$. Each individual's objectives are then evaluated by querying the surrogate models.

2. Non-Dominated Sorting. The population is partitioned into Pareto "fronts" F_1, F_2, ... Front F_1 consists of all non-dominated individuals in the current population. Once F_1 is identified and removed, the next set of non-dominated individuals forms F_2, and so on. Individuals in F_j are assigned rank j.

3. Crowding Distance Assignment. For each front F_j, a crowding distance is computed to measure how isolated each individual is from its neighbors in objective space. Concretely, for each objective k, the members of F_j are sorted by f_k. The two boundary individuals (the smallest and largest values of f_k) receive an infinite crowding distance. Any interior individual i in the sorted list for objective k has its crowding distance incremented by

$$d_i + \frac{f_k(z_{i+1}) - f_k(z_{i-1})}{f_k^{\max} - f_k^{\min}},$$

where $f_k^{\max}$ and $f_k^{\min}$ are the worst and best values of f_k within F_j. Summing this increment across all objectives yields each individual's total crowding distance. A

larger crowding distance indicates that an individual lies in a sparsely populated region of objective space, which NSGA-II favors to maintain diversity (Deb et al., 2002) [16].

4. Selection, Crossover, and Mutation. Parents for the next generation are selected by binary tournament: two individuals are sampled at random, and the one with a lower Pareto rank is chosen; if both have equal rank, the one with a larger crowding distance is chosen. Selected parents undergo genetic operators to produce offspring. For the continuous genes (g_{ICE}, g_{P2}, g_{P3}), a blend crossover (BLX-α) is applied: if two parent values are x_a and x_b, then the child value is drawn from the interval $\Big[\min(x_a, x_b) - \alpha\Delta,\ \max(x_a, x_b) + \alpha\Delta\Big], \quad \Delta = |x_a - x_b|,\ \alpha = 0.5$. After sampling, each child value is clamped to its respective interval. For the discrete gene m_{idx}, uniform crossover (i.e., swapping motor indices with equal probability) is used. Each gene (continuous or discrete) then has a small probability p_{mut} (e.g., 0.2) of mutation: continuous genes are perturbed by adding zero mean Gaussian noise $\mathcal{N}(0, \sigma^2)$ with $\sigma = 0.5$ and then clamped; the discrete gene may be replaced by a randomly chosen motor index. These crossover and mutation operators balance exploration (searching new regions) and exploitation (refining known good regions).

5. Environmental Selection. Once λ offspring are generated (commonly $\lambda = \mu$), their objective values are computed via the surrogate models. The parent and offspring populations—total size $\mu+\lambda$—are combined, and non-dominated sorting is performed again to assign new Pareto fronts. The next generation's population of size μ is formed by taking all of F_1, then as many of F_2 as needed, and so on. If including all of a frontier F_j would exceed μ, individuals in F_j are selected in descending order of crowding distance until μ individuals remain. This elitist $(\mu + \lambda)$ strategy preserves the best solutions discovered so far, while maintaining diversity.

6. Termination. Steps 2–5 are repeated for G generations (for example, $G = 50$). At convergence, the set of all non-dominated individuals in the final population approximates the true Pareto front P^*. Because surrogate evaluations are extremely fast, evolving a population of $\mu = 100$ for $G = 50$ takes only seconds, in contrast to days if each evaluation were run through the full simulator.

2.4.1 Single-Objective Reduction Strategies

Although NSGA-II is designed for true multi-objective search, in many practical scenarios a designer prefers to reduce the problem to a single scalar objective, either by combining all objectives into one weighted sum, or by optimizing a primary objective subject to constraints on the others. Below, both approaches are presented; the user may choose whichever best suits their decision context.

2.4.1.1 Weighted Sum Scalarization

In the weighted sum approach, each Pareto candidate design $z = \big(g_{ICE},\ g_{P3},\ g_{P4},\ m_{\text{idx}}\big)$, is evaluated on K surrogate predicted objectives $f_1(z), \ldots, f_K(z)$. To make these commensurate, first normalize each objective across the current Pareto front of size

P:

$$f_k^{\text{norm}}\Big(z^{(i)}\Big) = \frac{f_k\Big(z^{(i)}\Big) - f_k^{\min}}{f_k^{\max} - f_k^{\min}} \quad \text{for} \quad i = 1, \ldots, P,$$

where

$$f_k^{\min} = \min_i f_k\Big(z^{(i)}\Big) \quad \text{and} \quad f_k^{\max} = \max_i f_k\Big(z^{(i)}\Big).$$

The decision maker then specifies a nonnegative weight vector $w = (w_1, \ldots, w_K)$ with $\sum_k w_k = 1$. Each design $z^{(i)}$ receives a scalar score

$$S_i = \sum_{k=1}^{K} w_k \, f_k^{\text{norm}}\Big(z^{(i)}\Big).$$

Minimizing S_i across $i = 1, \ldots, P$ selects the design that best balances the objectives according to the user's priorities. Because the surrogates evaluate in microseconds, one can rapidly recompute S_i for different w to explore how preferences shift the best-trade-off solution.

2.4.1.2 Constraint-Based Scalarization

Let $\mathcal{X}$ be the design space and let $f = (f_1, \ldots, f_K) : \mathcal{X} \to \mathbb{R}^K$ denote the K objectives (assumed to be minimized; maximization objectives can be negated). Without loss of generality, we order them so that the *primary* objective is written $f_{\text{prim}} : \mathcal{X} \to \mathbb{R}$ and the remaining $K - 1$ objectives are f_j for $j = 1, \ldots, K - 1$. For each constrained objective we specify an upper bound $b_j \in \mathbb{R}$. The design variable is $z \in \mathcal{X}$.

Alternatively, the practitioner may wish to optimize a single primary objective $f_{\text{prim}}(z)$ while enforcing hard limits on the remaining $K - 1$ objectives. Denote these constraints as

$$f_j(z) \le b_j, \quad j = 1, \ldots, K - 1.$$

Define the total constraint violation

$$\Phi(z) = \sum_{j=1}^{K-1} \max\{0,\, f_j(z) - b_j\},$$

which is zero if and only if all constraints are satisfied, then form a penalized objective

$$F_{\text{pen}}(z) = \begin{cases} f_{\text{prim}}(z), & \Phi(z) = 0, \\ f_{\text{prim}}(z) + M\,\Phi(z), & \Phi(z) > 0, \end{cases}$$

where $M \gg 0$ (e.g. 10^6) ensures any infeasible design (with $\Phi > 0$) is dominated by the worst feasible design. One can then run a classical single-objective genetic algorithm (or NSGA-II configured with $K = 1$) to minimize $F_{\text{pen}}(z)$. Feasible designs naturally prevail, and among them, the search focuses on improving the primary objective.

3

Results

In this section, the focus is on presenting the validation of the surrogate models against held-out test data and the outcomes of the multiobjective optimization, including Pareto front characteristics and the selected best trade-off design.

3.1 Surrogate Model Validation

The Random Forest model performs best on the efficiency dataset(R^2=0.9639, RMSE=0.0765),XGB is a close second ($R^2 = 0.9566$, RMSE=0.0790), suggesting the underlying response surface is highly non-linear but with relatively modest extrapolation demands, which bagging and boosted trees can both model well. Whereas XGBoost gives the best output in case of the Performance dataset, as seen in the Table 3.1.

Table 3.1: Surrogate model performance for efficiency and performance prediction tasks.

Model	**Efficiency**		**Performance**	
	MSE	R^2	MSE	R^2
Random Forest	0.0059	0.9639	18.3489	0.7007
Ridge	0.0593	0.6569	102.6265	0.6274
SVR	0.0119	0.9273	28.8506	0.7971
XGBoost	0.0062	0.9566	14.8141	0.8439
LightGBM	0.0104	0.9321	17.7945	0.8400

Best efficiency surrogate: Random Forest ($R^2 = 0.9639$). *Best performance surrogate:* XGBoost ($R^2 = 0.8439$).

The R^2 value for the performance dataset is relatively lower than the efficiency dataset due to the sparse dataset. For very high values of input configurations (unrealistic), some of the columns in the simulations remain empty due to the vehicle not achieving the threshold. For example, when the gear ratios are kept very high (gear ratio ICE=7, P3=14, P4=14), the vehicle never reached a top speed of 250 Km/h. This leaves that column empty, which results in the removal of the entire data point. Even though the R^2 value is comparatively less, it does not affect the

optimizer drastically and still achieves the best results. The output given by the optimizer is compared against the high-fidelity simulations.

Parity plots and residual distributions (Figure 3.1–3.2) indicate well-calibrated surrogates without heavy bias; residuals are unimodal and centered near zero. The plot in the Figure 3.1 shows parity for the 0–100 km/h time target (perf group). Predictions align tightly along the 45° line with slight under-prediction in the slow regime (> 12 s), where data are sparser. This behaviour is typical for boosted trees trained with squared error: the model trades small bias in the tails for lower overall variance. The residual histogram (Figure 3.2) is centred near zero with a light positive skew and truncated tails. There is no evidence of patterning by magnitude (i.e., no strong heteroscedasticity), indicating that the preprocessor (imputation, scaling, One Hot Encoding (OHE)) plus the chosen capacity control (depth, leaves, shrinkage) offers good calibration.

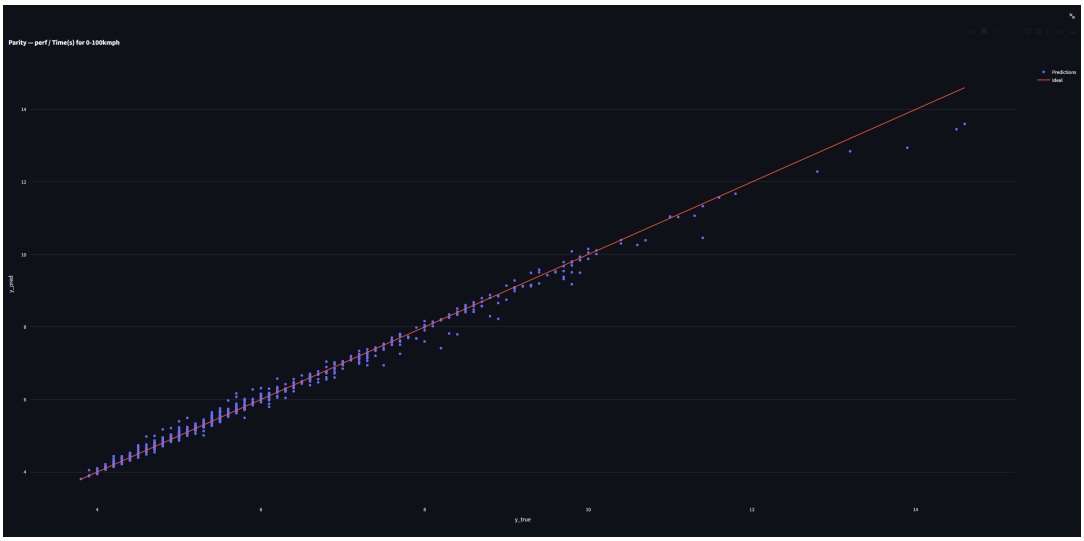


Figure 3.1: Parity plot for a key performance target (test set).

Further, to understand the impact of each feature, the feature importance is analyzed. This reports permutation/tree importances aggregated by original feature (i.e., all one-hot columns for Electric Motor are summed into one). Figure 3.3 and Figure 3.4 show the top drivers of efficiency and performance datasets, respectively.

3.2 Pareto Front Exploration

Using NSGA-II with a population of 100 and 50 generations, the model evaluated approximately 20000 surrogate predicted designs. Figures and tables would typically illustrate the trade-off between different objectives that are to be optimized. As seen in Figure 3.5 and Figure 3.6, when optimized only for Battery usage and acceleration a clear knee plot is seen and the value selected (weighted mode with equal weights), whereas, when optimized on multiple objectives like Acceleration, top speed, battery usage and Fuel efficiency the chosen value on a 2-D graph is not so interpretable.

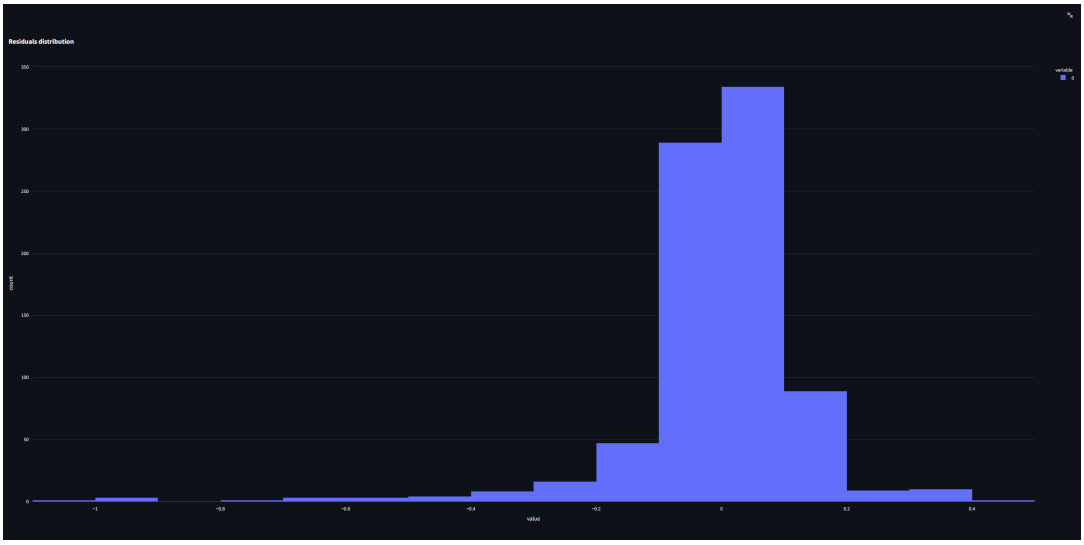


Figure 3.2: Residual distribution for the same target.

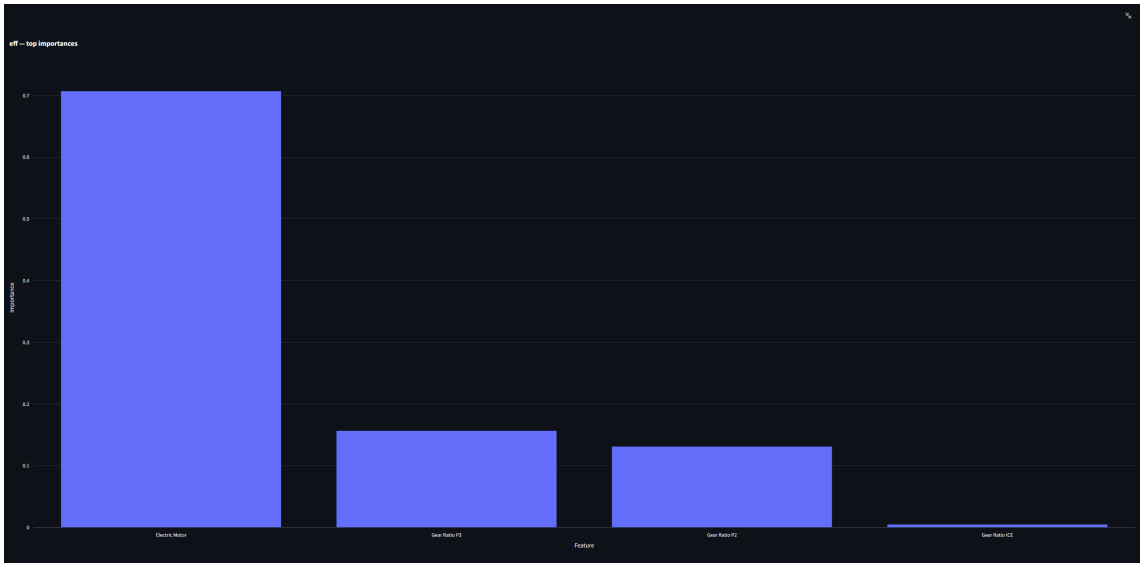


Figure 3.3: Grouped feature importances (one-hot levels collapsed to their base feature) for efficiency dataset.

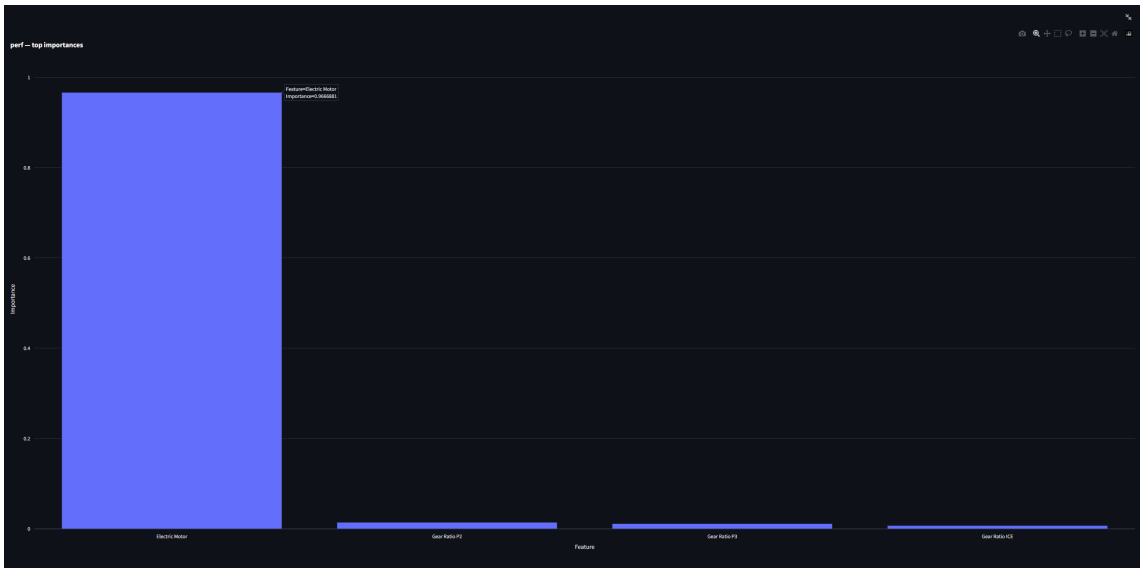


Figure 3.4: Grouped feature importances (one-hot levels collapsed to their base feature) for performance dataset.

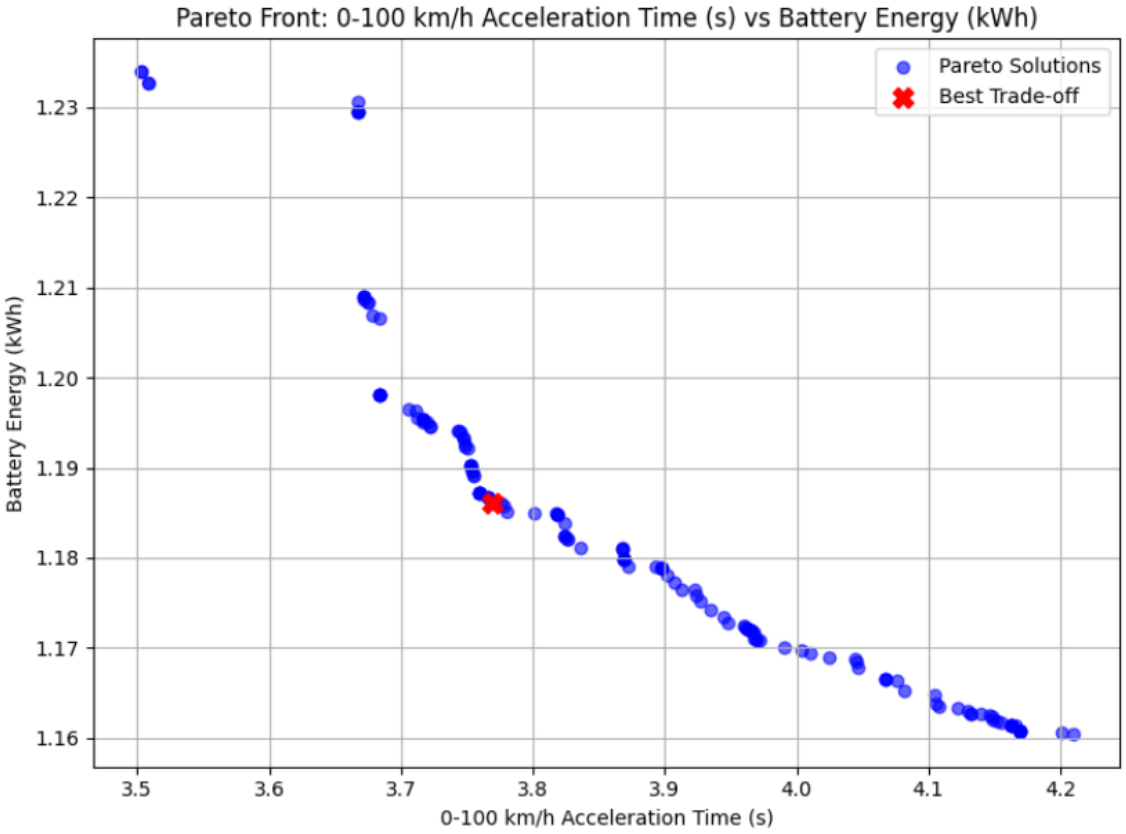


Figure 3.5: 2-D graph showing the Pareto-optimal solutions. Objectives are only Battery Energy(kWh)(Minimized) and 0-100 km/h acceleration time(minimized).

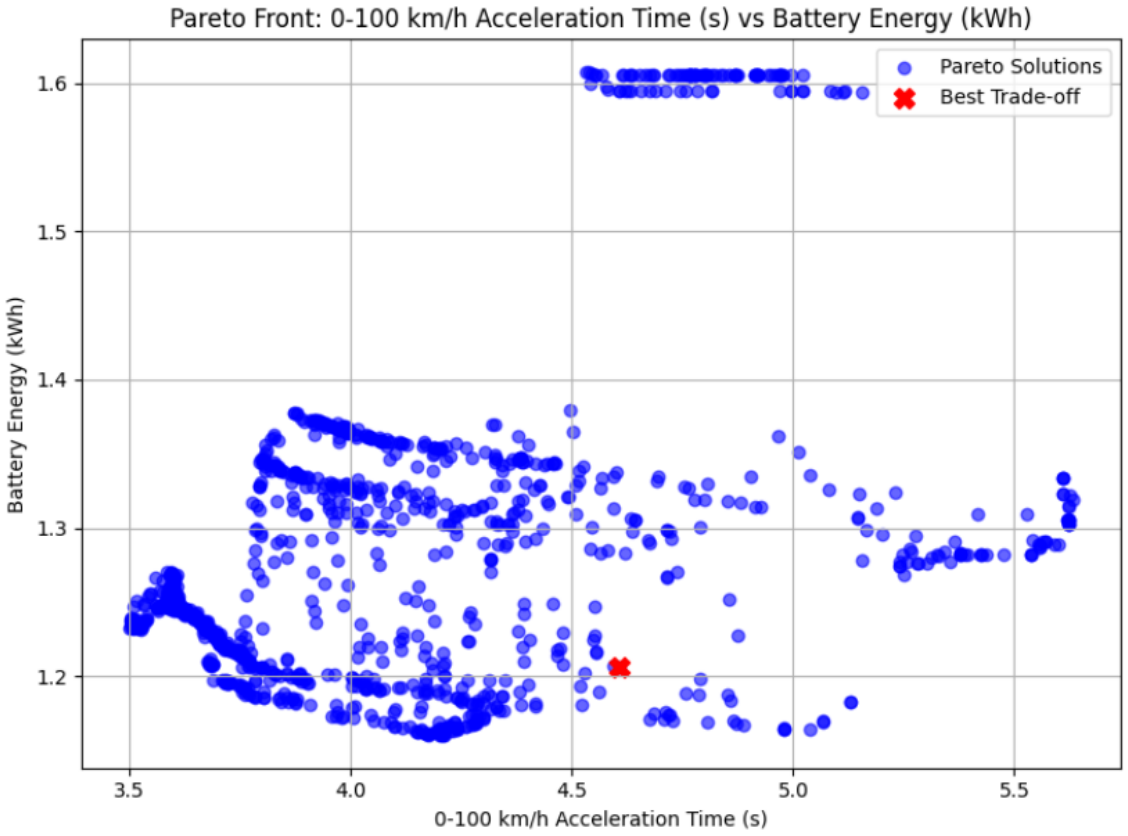


Figure 3.6: 2-D graph showing the Pareto-optimal solutions. Multi-objective optimization.

3.3 Best Trade Off Design Selection

Using the front end developed the user can choose between the weighted sum method or the constrained optimization method if they wish to convert this multi-objective optimization problem to a single solution, single objective optimization task. The selected best values are cross-examined with the high-fidelity simulations.

4

Conclusion

4.1 Discussion

This work demonstrates a scalable, data-driven workflow for hybrid electric powertrain design that sidesteps the prohibitive cost of exhaustive high-fidelity simulations. Using a space-filling Sobol design (4096 candidate configurations), the dataset is generated for two surrogate modelling tasks: an efficiency group (e.g., fuel consumption, battery energy) and a performance group (e.g., acceleration times, top speed).

A carefully engineered preprocessing pipeline (median imputation for numerics; one-hot encoding for the motor category; standardisation; all encapsulated inside sklearn pipelines) ensured leakage-free cross-validation and robust generalisation. This work evaluated five model families spanning the bias–variance spectrum: Ridge (linear baseline), SVR (RBF kernel), Random Forest (bagging), XGBoost, and LightGBM (additive boosting). This portfolio was chosen deliberately: Ridge provides a transparent baseline under multicollinearity; SVR offers a smooth non-parametric comparator; tree ensembles capture high-order non-linear interactions and handle mixed numeric/categorical inputs gracefully. Selecting the best model *per group* via CV and verifying on a held-out split yielded the following headline results:

- **Efficiency surrogate: Random Forest** achieved the best held-out accuracy ($R^2 \approx 0.964$, RMSE ≈ 0.0765), with XGBoost a close second ($R^2 \approx 0.957$).
- **Performance surrogate: XGBoost** led ($R^2 \approx 0.844$, RMSE ≈ 3.84), narrowly ahead of LightGBM ($R^2 \approx 0.84$).

These outcomes are consistent with the nature of each task. The efficiency response surface is highly non-linear but relatively "piecewise-flat", favoring RF's variance-reduction through averaging. Performance metrics exhibit smoother monotone trends with mild tail bias, gradient boosting's shrinkage and depth regularisation deliver lower error there.

With these surrogates in hand, we employed the NSGA-II algorithm to approximate the Pareto front of multiobjective trade-offs among fuel consumption, acceleration, and cost. In a matter of seconds, evaluating some 20000 designs via the surrogates, we identified a diverse set of non-dominated configurations. We then illustrated two common decision-making paradigms: weighted sum scalarization to select a

balanced trade-off point, and constraint-based reduction to enforce hard performance limits. Both approaches produced candidate designs whose surrogate predictions were confirmed by high-fidelity simulation.

To facilitate stakeholder exploration, we built an interactive Streamlit application featuring dynamic Pareto front plotting and "click to inspect" capabilities. Users can select any two objectives for visualization and immediately view all 14 performance and efficiency metrics of any plotted point. The chosen "best" design is highlighted on the chart for clear reference, along with the best solution provided based on either the weighted sum or the constrained method.

By combining principled model selection, rigorous diagnostics, and evolutionary optimization inside an interactive app, this work delivers a practical, end-to-end surrogate-based design tool. The surrogates provide orders of magnitude faster evaluations while preserving accuracy sufficient for decision making. The resulting methodology enables early trade-off exploration, reduces simulation burden, and focuses design effort on the levers that matter most, chiefly the motor choice and, secondarily, the P3/P4 gearing. With the proposed extensions in uncertainty handling, transfer learning, and robust inverse design, the framework is well-positioned to support future hybrid powertrain programmes at an industrial scale.

4.2 Future Work

Looking ahead, several extensions could make this framework even more powerful. First, applying transfer learning, where we use the trained weights and use fewer points to train the models on new data. This would enable the stakeholders to extend this framework to other fields where they just need to input a few data samples. Second, a systematic feature importance analysis (for example, via SHAP or permutation importance) could reveal which other parameters drive each performance and efficiency metric, enabling targeted dimensionality reduction. Building on that, retraining surrogates using only the most influential features may improve generalization and evaluation speed. Third, adding an inverse design capability would let users specify desired outputs (e.g., "0–100 km/h in under 5 s and fuel consumption below 4 L/100 km") and receive a recommended set of configurations via constrained optimization over the surrogate. Finally, incorporating uncertainty quantification using a Gaussian process or Bayesian surrogates would provide confidence bounds on predictions, supporting risk-aware decision-making in powertrain design.

Bibliography

[1] M. Ehsani, Y. Gao, and A. Emadi, *Modern Electric, Hybrid Electric, and Fuel Cell Vehicles: Fundamentals, Theory, and Design*, 2nd ed. CRC Press, 2010.

[2] A. Emadi, *Handbook of Automotive Power Electronics and Motor Drives*. CRC Press, 2017.

[3] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[4] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, 2016, pp. 785–794.

[5] G. Ke, Q. Meng, T. Finley, *et al.*, "Lightgbm: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems*, vol. 30, 2017, pp. 3146–3154.

[6] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[7] V. N. Vapnik, *The Nature of Statistical Learning Theory*. Springer, 1995.

[8] H. Drucker, C. J. Burges, L. Kaufman, A. Smola, and V. Vapnik, "Support vector regression machines," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 9, 1997.

[9] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, 2004.

[10] A. I. J. Forrester, A. Sobester, and A. J. Keane, *Engineering Design via Surrogate Modelling: A Practical Guide*. John Wiley & Sons, 2008.

[11] S. Davis and S. Cremaschi, "Efficient surrogate model development: Impact of sample size and underlying model dimensions," in Jan. 2018, pp. 979–984, ISBN: 9780444642417. DOI: 10.1016/B978-0-444-64241-7.50158-0.

[12] I. M. Sobol, "On the distribution of points in a cube and the approximate evaluation of integrals," *USSR Computational Mathematics and Mathematical Physics*, vol. 7, no. 4, pp. 86–112, 1967.

[13] D. Bhattacharjee, T. Ghosh, P. Bhola, K. Martinsen, and P. Dan, "Data-driven surrogate assisted evolutionary optimization of hybrid powertrain for improved fuel economy and performance," *Energy*, vol. 183, pp. 235–248, Jun. 2019. DOI: 10.1016/j.energy.2019.06.115.

[14] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA: MIT Press, 2016.

[15] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *International Joint Conference on Artificial Intelligence (IJCAI)*, vol. 14, 1995, pp. 1137–1145.

[16] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

A

Appendix 1

A.1 Hyperparameters

Table A.1: Hyperparameter search grids. Parameters on the pipeline's `reg` step are prefixed `reg__`; parameters of an estimator wrapped inside `MultiOutputRegressor` use `reg__estimator__`.

Algorithm	**Parameter**	**Values**
Random Forest	`reg__n_estimators`	[200, 400, 800]
	`reg__max_depth`	[`None`, 10, 20]
	`reg__min_samples_leaf`	[1, 2, 5]
Ridge (MultiOutput)	`reg__estimator__alpha`	[0.1, 1.0, 10.0, 50.0]
SVR (RBF, MultiOutput)	`reg__estimator__C`	[1.0, 5.0, 10.0, 20.0]
	`reg__estimator__epsilon`	[0.05, 0.1, 0.2]
	`reg__estimator__gamma`	[`scale`, `auto`]
XGBoost (MultiOutput)	`reg__estimator__n_estimators`	[200, 400, 800]
	`reg__estimator__max_depth`	[3, 6, 9]
	`reg__estimator__learning_rate`	[0.03, 0.1]
	`reg__estimator__subsample`	[0.7, 1.0]
	`reg__estimator__colsample_bytree`	[0.7, 1.0]
	`reg__estimator__reg_lambda`	[1.0, 3.0]
LightGBM (MultiOutput)	`reg__estimator__n_estimators`	[400, 800]
	`reg__estimator__num_leaves`	[31, 63, 127]
	`reg__estimator__learning_rate`	[0.05, 0.1]
	`reg__estimator__subsample`	[0.7, 1.0]
	`reg__estimator__colsample_bytree`	[0.7, 1.0]
	`reg__estimator__reg_lambda`	[0.0, 1.0]

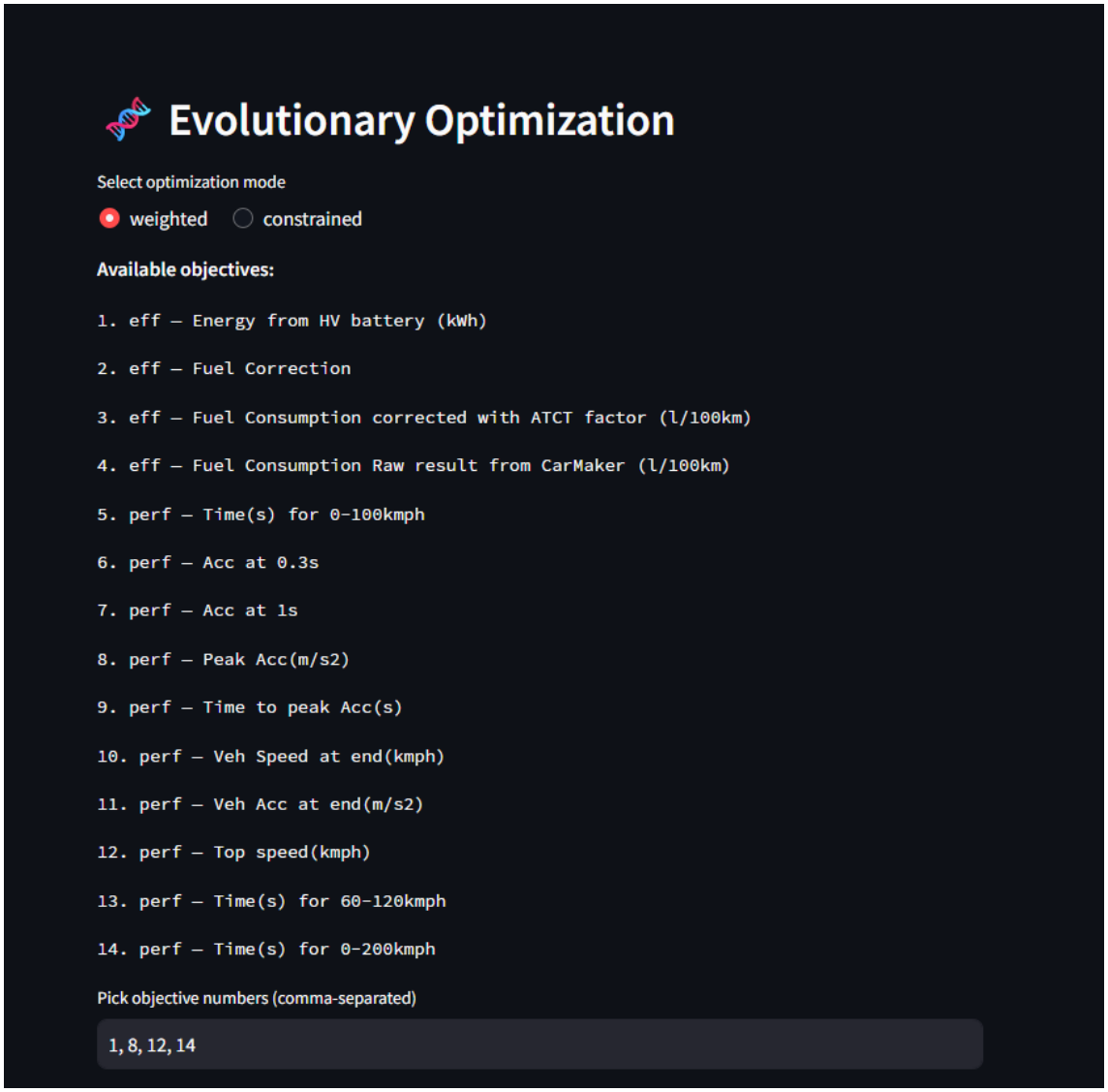
Table A.2: Efficiency group: cross-validation (CV) best scores, held-out metrics, and best parameters selected.

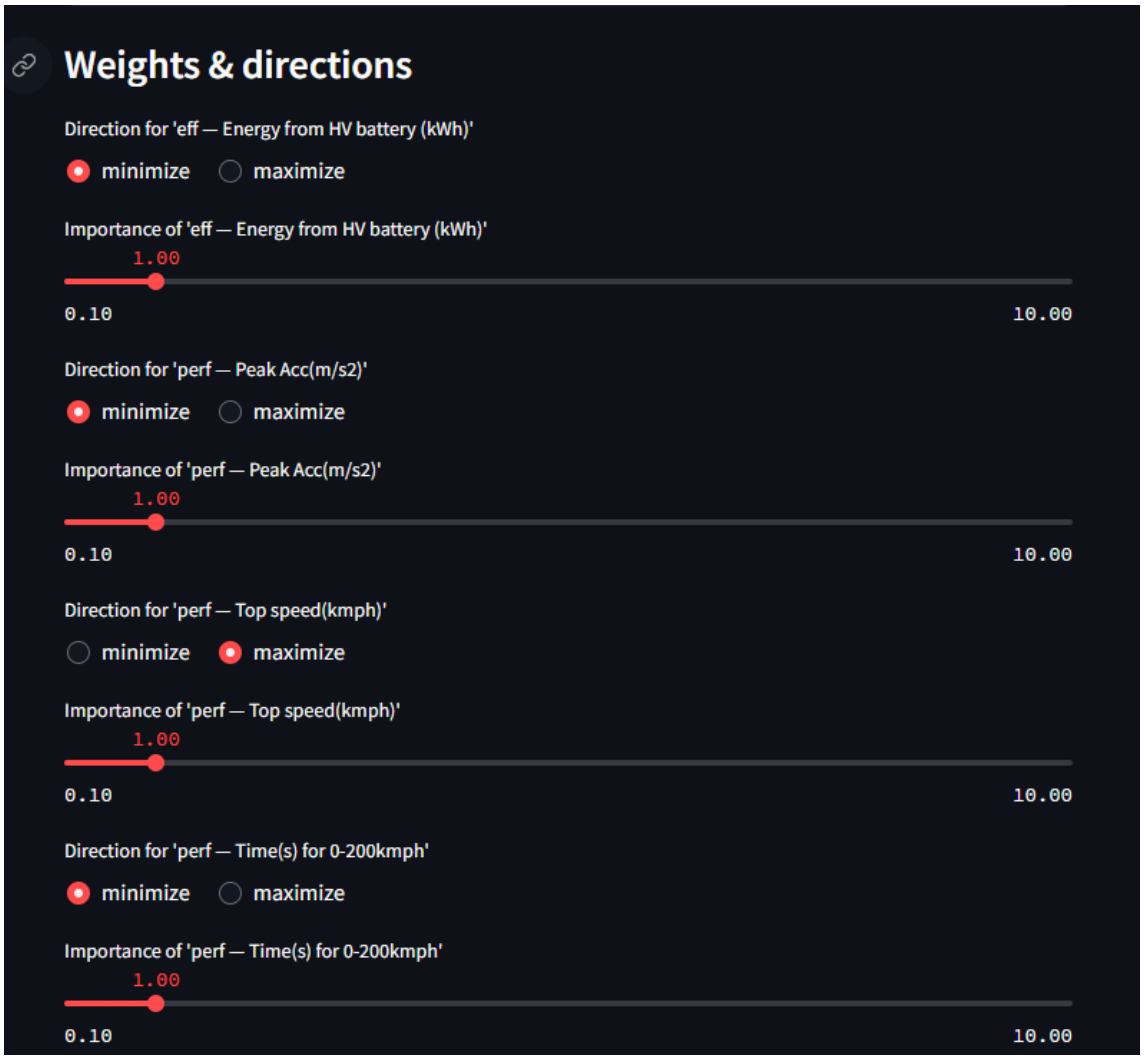
Algorithm	**CV best R^2**	**Holdout R^2**	**Holdout RMSE**	**Best parameters**
Random Forest	0.9453	**0.9632**	**0.0752**	`reg__n_estimators`=800 `reg__min_samples_leaf`=1 `reg__max_depth`=None
Ridge (MultiOutput)	0.6309	0.6657	0.2417	`reg__estimator__alpha`=1.0
SVR (RBF, MultiOutput)	0.9085	0.9263	0.1051	`reg__estimator__gamma`=scale `reg__estimator__epsilon`=0.05 `reg__estimator__C`=5.0
XGBoost (MultiOutput)	0.9421	0.9582	0.0780	`reg__estimator__subsample`=0.7 `reg__estimator__reg_lambda`=1.0 `reg__estimator__n_estimators`=400 `reg__estimator__max_depth`=9 `reg__estimator__learning_rate`=0.03 `reg__estimator__colsample_bytree`=1.0
LightGBM (MultiOutput)	0.9104	0.9320	0.1012	`reg__estimator__subsample`=1.0 `reg__estimator__reg_lambda`=0.0 `reg__estimator__num_leaves`=31 `reg__estimator__n_estimators`=400 `reg__estimator__learning_rate`=0.1 `reg__estimator__colsample_bytree`=1.0

Table A.3: Performance group: cross-validation (CV) best scores, held-out metrics, and best parameters selected.

Algorithm	**CV best R^2**	**Holdout R^2**	**Holdout RMSE**	**Best parameters**
Random Forest	0.6758	0.7006	4.2856	`reg__n_estimators`=800 `reg__min_samples_leaf`=1 `reg__max_depth`=None
Ridge (MultiOutput)	0.6240	0.6274	10.1304	`reg__estimator__alpha`=1.0
SVR (RBF, MultiOutput)	0.7748	0.7971	5.3713	`reg__estimator__gamma`=scale `reg__estimator__epsilon`=0.05 `reg__estimator__C`=20.0
XGBoost (MultiOutput)	**0.8267**	**0.8407**	**3.8170**	`reg__estimator__subsample`=0.7 `reg__estimator__reg_lambda`=1.0 `reg__estimator__n_estimators`=400 `reg__estimator__max_depth`=9 `reg__estimator__learning_rate`=0.1 `reg__estimator__colsample_bytree`=0.7
LightGBM (MultiOutput)	0.8103	0.8393	4.2183	`reg__estimator__subsample`=0.7 `reg__estimator__reg_lambda`=0.0 `reg__estimator__num_leaves`=31 `reg__estimator__n_estimators`=400 `reg__estimator__learning_rate`=0.05 `reg__estimator__colsample_bytree`=0.7

A.2 App Design




Weights & directions
Direction for 'eff — Energy from HV battery (kWh)'
minimize
maximize
Importance of 'eff — Energy from HV battery (kWh)'
1.00
0.10
10.00
Direction for 'perf — Peak Acc(m/s2)'
minimize
maximize
Importance of 'perf — Peak Acc(m/s2)'
1.00
0.10
10.00
Direction for 'perf — Top speed(kmph)'
minimize
maximize
Importance of 'perf — Top speed(kmph)'
1.00
0.10
10.00
Direction for 'perf — Time(s) for 0-200kmph'
minimize
maximize
Importance of 'perf — Time(s) for 0-200kmph'
1.00
0.10
10.00


Evolution settings
Population
40
30
500
Generations
10
5
200
μ (parents kept)
20
10
500
λ (offspring per gen)
20
10
500
Crossover prob
0.60
0.10
0.90
Mutation prob
0.30
0.10
0.90
Run Optimization

Pareto Set

	Gear Ratio ICE	Gear Ratio P2	Gear Ratio P3	Electric Motor	eff — Energy from HV battery (kWh)	per
0	5.5473	14.7095	7.3593	Thesis_comb_14	1.639	
1	5.4677	14.0454	7	Thesis_comb_14	1.6377	
2	5.7289	14.021	7.2774	Thesis_comb_14	1.6371	
3	4.8206	13.811	8.0474	Thesis_comb_14	1.6075	
4	4.8206	13.7006	8.0474	Thesis_comb_14	1.6067	
5	6.2365	12.0101	7.4262	Thesis_comb_8	1.6053	
6	5.8208	11.4685	7.582	Thesis_comb_3	1.6051	
7	4.9216	13.8829	7.8281	Thesis_comb_8	1.6049	
8	6.2438	8.3707	9.2448	Thesis_comb_23	1.6041	
9	6.9918	8.8677	10.8083	Thesis_comb_23	1.6032	

Download CSV

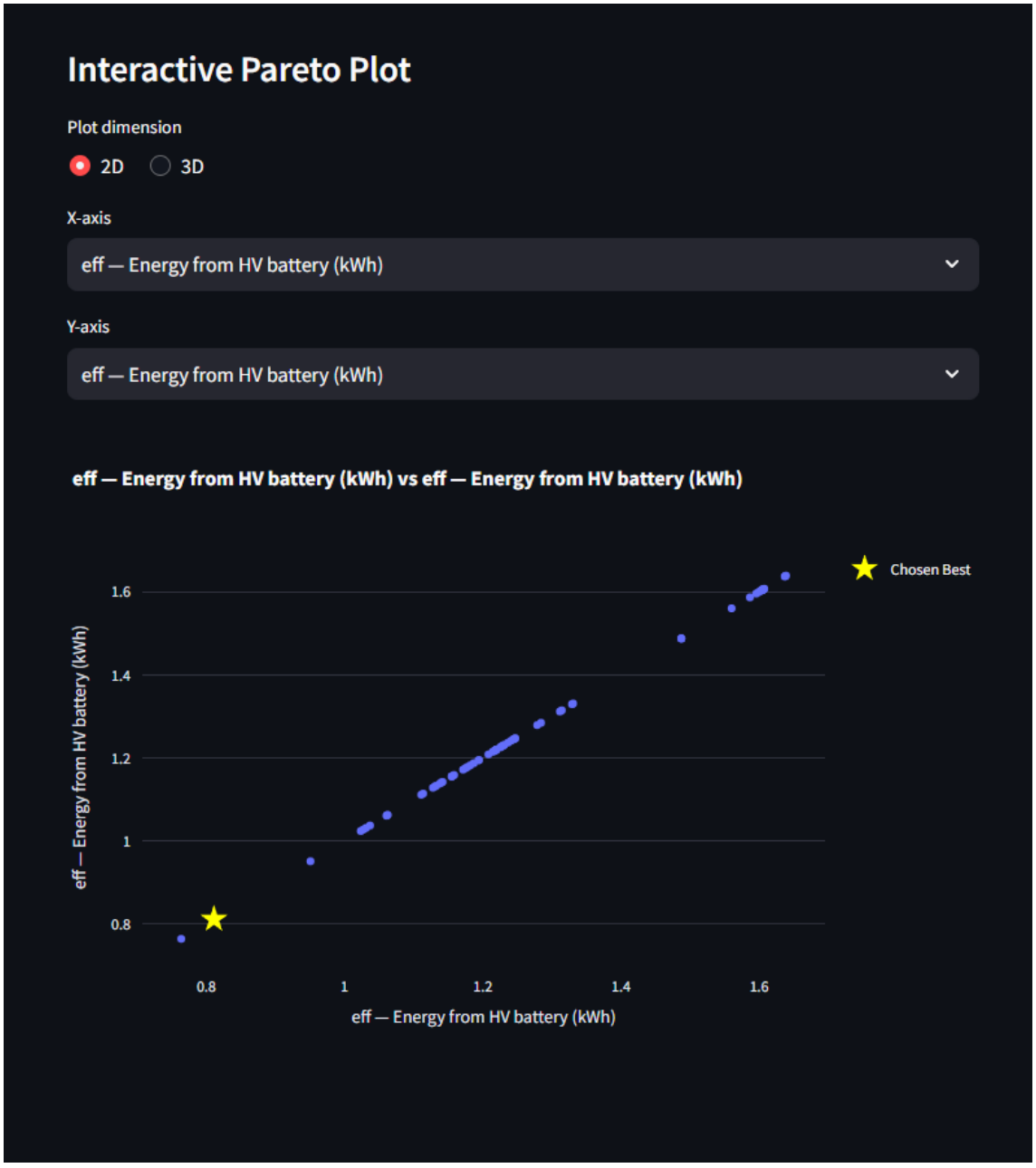
Best Trade-Off — Features & All Outputs

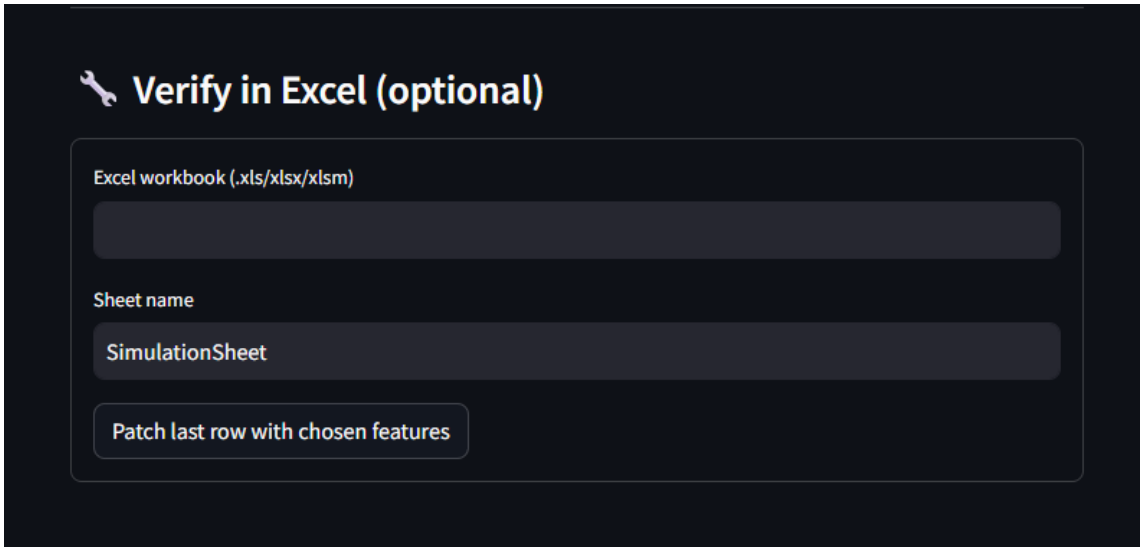
Chosen Features

	Gear Ratio ICE	Gear Ratio P2	Gear Ratio P3	Electric Motor
0	3.6932	8.2708	15	Thesis_comb_10

Predicted Outputs

	eff — Energy from HV battery (kWh)	eff — Fuel Correction	eff — Fuel Consumption corrected with ATCT facto
0	0.8113	1.4427	


Interactive Pareto Plot
Plot dimension
2D
3D
X-axis
eff — Energy from HV battery (kWh)
Y-axis
eff — Energy from HV battery (kWh)
eff — Energy from HV battery (kWh) vs eff — Energy from HV battery (kWh)
Chosen Best
1.6
1.4
1.2
1
0.8
eff — Energy from HV battery (kWh)
0.8
1
1.2
1.4
1.6
eff — Energy from HV battery (kWh)


Verify in Excel (optional)
Excel workbook (.xls/xlsx/xlsm)
Sheet name
SimulationSheet
Patch last row with chosen features