Journal of Data Analysis and Information Processing, 2024, 12, 348-369
https://www.scirp.org/journal/jdaip
ISSN Online: 2327-7203
ISSN Print: 2327-7211

A Novel Approach for Developing a Linear Regression Model within Logistic Cluster Using Scikit-Learn

Nwosu Ambrose[1*], Gilbert I. O. Aimufua[1*], Choji Davou Nyap[2]

[1]Departments of Computer Sciences, Nasarawa State University, Keffi, Nigeria
[2]Departments of Computer Sciences, University of Jos, Jos, Nigeria
Email: *crossdelina20012001@yahoo.com, *aimufuagio@nsuk.edu.ng

How to cite this paper: Ambrose, N., Aimufua, G.I.O. and Nyap, C.D. (2024) A Novel Approach for Developing a Linear Regression Model within Logistic Cluster Using Scikit-Learn. *Journal of Data Analysis and Information Processing*, **12**, 348-369.
https://doi.org/10.4236/jdaip.2024.123019

Received: January 1, 2024
Accepted: June 25, 2024
Published: June 28, 2024

Abstract

Due to the rapid development of logistic industry, transportation cost is also increasing, and finding trends in transportation activities will impact positively in investment in transportation infrastructure. There is limited literature and data-driven analysis about trends in transportation mode. This thesis delves into the operational challenges of vehicle performance management within logistics clusters, a critical aspect of efficient supply chain operations. It aims to address the issues faced by logistics organizations in optimizing their vehicle fleets' performance, essential for seamless logistics operations. The study's core design involves the development of a predictive logistics model based on regression, focused on forecasting, and evaluating vehicle performance in logistics clusters. It encompasses a comprehensive literature review, research methodology, data sources, variables, feature engineering, and model training and evaluation and F-test analysis was done to identify and verify the relationships between attributes and the target variable. The findings highlight the model's efficacy, with a low mean squared error (MSE) value of 3.42, indicating its accuracy in predicting performance metrics. The high R-squared (R^2) score of 0.921 emphasizes its ability to capture relationships between input characteristics and performance metrics. The model's training and testing accuracy further attest to its reliability and generalization capabilities. In interpretation, this research underscores the practical significance of the findings. The regression-based model provides a practical solution for the logistics industry, enabling informed decisions regarding resource allocation, maintenance planning, and delivery route optimization. This contributes to enhanced overall logistics performance and customer service. By addressing performance gaps and embracing modern logistics technologies, the study supports the ongoing evolution of vehicle performance management in logistics clusters, fostering increased competitiveness and sustainability in the logistics sector.

Keywords

Mean Squared Error, R^2 Score, F-Test, MSE

1. Introduction

Transportation, warehousing, and distribution are all included in the logistics industry, which is a crucial part of the global economy. The smooth flow of goods from manufacturers to consumers is essential in this industry, and transportation plays a key role in this. For logistics clusters, which are strategically placed geographic areas where logistics activity is concentrated, efficient vehicle performance is crucial. These clusters provide advantages like increased connectivity, scale economies, and access to resources and infrastructure. To creating precise predictive models for vehicle performance in logistics clusters, the availability of extensive historical data is essential. Accurate vehicle performance predictions enable the optimisation of delivery routes, resource allocation, proactive maintenance, customer service, sustainability, cost reduction, real-time adaptation, and competitiveness [1]. By leveraging predictive models, logistics companies can optimise routes, allocate resources effectively, enhance customer service, and reduce operational costs, ensuring a competitive edge in the ever-evolving logistics industry [2]. A data-driven approach for route optimisation that incorporates real-time traffic data to enhance delivery performance was proposed by [3]. The significance of data-driven approaches in improving logistics operations has been highlighted by researchers. Former researches carried out were not analyzed using L2 and ridge regularization also F test was not carried out to test the relationships between data.

The Aim of This Work

The primary objective (aim) of this study is to develop a predictive logistics model using linear regression for accurate forecasting of vehicle performance within logistics clusters. This model aims to enhance resource allocation, maintenance planning, delivery routes, and overall logistics efficiency.

2. Scikit-Learn Regression

Scikit-learn, a powerful Python library for machine learning, provided us with a quick and efficient way to implement Regression [4]. This approach is widely used and trusted for its robustness and comprehensive set of tools for model development. It offers a set of fast tools for machine learning and statistical modeling, such as classification, regression, clustering, and dimensionality reduction, via a Python interface. Scikit-learn is a Python package that makes it easier to apply a variety of Machine Learning (ML) algorithms for predictive data analysis, such as linear regression.

Linear regression is defined as the process of determining the straight line that best fits a set of dispersed data points [5].

Key Steps in Scikit-learn Regression:

- Data Preprocessing: We began by splitting our dataset into training and testing sets to evaluate model performance. We also normalized the data to ensure that features were on the same scale.
- Model Fitting: We employed the Regression module from scikit-learn to fit a model to our training data. This step involved finding the optimal coefficients for each feature, including the intercept term.
- Prediction: After model training, we used the trained model to make predictions on the test dataset.
- Evaluation: To assess model performance, we computed key metrics such as Mean Squared Error (MSE) and R-squared (R^2) scores [6].

2.1. Custom Mathematical Models

In addition to scikit-learn's Regression, custom mathematical models was used to gain deeper insights into the underlying mathematical principles of regression and to explore the concept of Ridge Regression, a technique that introduces regularization to the regression model.

The Mathematical model of Linear Regression was used to predict the city-mpg and highway-mpg of the car.

By:

- Splitting the data into train and test set
- Finding the coefficients of the model
- Finding the intercept of the model
- Predicting the values of the test set
- Finding the accuracy of the model based on the equation below:

$$h(\theta, x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots \theta_n x_n \tag{1}$$

- $h(\theta, x)$: The predicted value for input x using the regression model.
- θ_0: Intercept term.
- $\theta_1 x_1$: regression coefficient (θ_1).
- $\theta_n x_n$: regression coefficient of the last independent variable.
- $\theta_1, \theta_2, \ldots, \theta_n$: Coefficients for the features $x_1, x_2, \ldots, x_n$.
- $x_1, x_2, \ldots, x_n$: Features of the input data.

2.2. Key Steps in Custom Mathematical Models

- Data Preprocessing: Like the scikit-learn approach, data was preprocessed by splitting it into training and testing sets and normalizing the features.
- Model Development: A custom mathematical models was used for both basic Regression and Ridge Regression. These models allowed us to understand the inner workings of regression and the impact of regularization.

- o Regression: In this model, the coefficients were calculated for regression model using matrix operations. This approach provided insights into the fundamental equations behind regression.
- o Ridge Regression: Building upon the Regression model, we incorporated L2 regularization (Ridge) into our custom model. This able to control overfitting by adding a penalty term to the optimization process.
- Prediction: custom mathematical models to predict "city-mpg" values on the test dataset.
- Evaluation: Assessment of the performance of these models was done by calculating metrics such as MSE and R^2 scores.

2.3. Hypothesis Function: Multiple Regression

$$h(\theta, x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots \theta_n x_n \quad (2)$$

- $h(\theta, x)$: The predicted value for input x using the regression model.
- θ_0: Intercept term.
- $\theta_1 x_1$: regression coefficient (θ_1).
- $\theta_n x_n$: regression coefficient of the last independent variable.
- $\theta_1, \theta_2, \ldots, \theta_n$: Coefficients for the features $x_1, x_2, \ldots, x_n$.
- $x_1, x_2, \ldots, x_n$: Features of the input data.

2.4. Cost Function (Mean Squared Error)

$$J(\theta) = (1/2m) \sum_i \left(h(\theta, x^i) - y^i \right)^2 \quad (3)$$

- $J(\theta)$: The cost function that measures the error of the model's predictions.
- m: The number of training examples.
- x^i: The feature vector of the i-th training example.
- y^i: The actual target value of the i-th training example.
- $h(\theta, x^i)$: The predicted value for the i-th training example using the hypothesis function.

2.5. Parameter Estimation

- The parameters ($\theta_0, \theta_1, \ldots, \theta_n$) of the regression model are estimated by minimizing the cost function $J(\theta)$.

2.6. Ridge Regression Model

1) Hypothesis Function for Ridge Regression:

$$h(\theta, x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots \theta_n x_n \quad (4)$$

- $h(\theta, x)$: The predicted value for input x using the Ridge regression model.
- θ_0: Intercept term.
- $\theta_1, \theta_2, \ldots, \theta_n$: Coefficients for the features $x_1, x_2, \ldots, x_n$.
- $x_1, x_2, \ldots, x_n$: Features of the input data.

2) Cost Function (Ridge Regression):

$$J(\theta) = (1/2m)\sum_i \left(h(\theta, x^i) - y^i\right)^2 + (\lambda/2m)\sum_j \theta_j^2 \quad (5)$$

- $J(\theta)$: The cost function that measures the error of the model's predictions.
- m: The number of training examples.
- x^i: The feature vector of the i-th training example.
- y^i: The actual target value of the i-th training example.
- $h(\theta, x^i)$: The predicted value for the i-th training example using the hypothesis function.
- λ: The regularization parameter (alpha).
- θ_j: Coefficients for the features x_j.

3) Parameter Estimation for Ridge Regression:

- The parameters (θ_0, θ_1, …, θ_n) of the Ridge regression model are estimated by minimizing the cost function $J(\theta)$ while also considering the regularization term.

3. The Approach

The development of the logistics engineering - regression based predictive feature model using sk-learn for vehicle performance in logistics clusters must begin with engineering as shown in Figure 1. Selecting and altering the relevant features or variables is necessary to enhance the model's performance and predictive abilities. Feature engineering aims to extract the most beneficial and distinctive features from the gathered data to increase the predictive model's accuracy and efficacy [7].

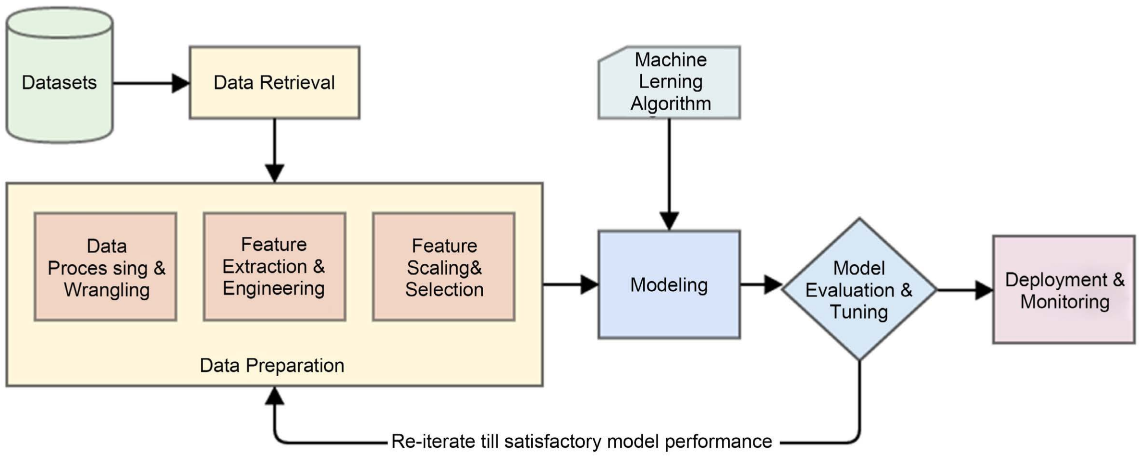

Datasets
Data Retrieval
Machine Lerning Algorithm
Data Proces sing & Wrangling
Feature Extraction & Engineering
Feature Scaling& Selection
Data Preparation
Modeling
Model Evaluation & Tuning
Deployment & Monitoring
Re-iterate till satisfactory model performance

Figure 1. Development of the regression model.

3.1. Data Splitting

Data splitting and regression model training are two crucial steps in the development of the logistics clusters' predictive model for vehicle performance as shown in Figure 2. While the training set is used to create the model, the testing set is used to evaluate the performance and generalizability of the predictive model to new, untested data.

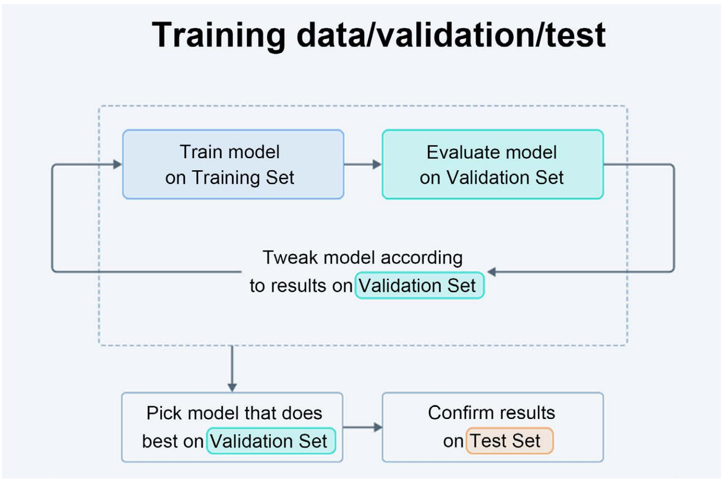
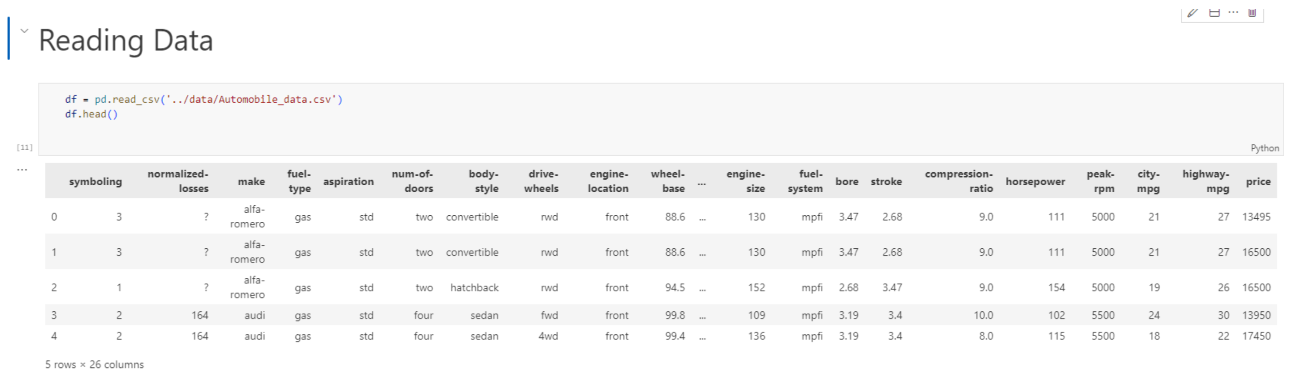

Training data/validation/test
Train model on Training Set
Evaluate model on Validation Set
Tweak model according to results on Validation Set
Pick model that does best on Validation Set
Confirm results on Test Set

Figure 2. Training data/validation/test.

After data splitting, the regression model must be fitted to the training data to train the model. The model gains knowledge of the coefficients or weights for each feature to forecast vehicle performance based on the input variables. The model is then tested on the testing set using the appropriate performance metrics, such as mean absolute error (MAE), root mean square error (RMSE), and coefficient of determination (R-squared)..

3.2. Data Preparation

Data preparation is a meticulous process aimed at ensuring that the dataset is ready for modelling and analysis as shown in **Figure 3**. It encompasses various tasks, such as data cleaning, feature engineering, and encoding, all of which contribute to the dataset's quality and suitability for predictive modelling.



Reading Data

```
df = pd.read_csv('../data/Automobile_data.csv')
df.head()
```

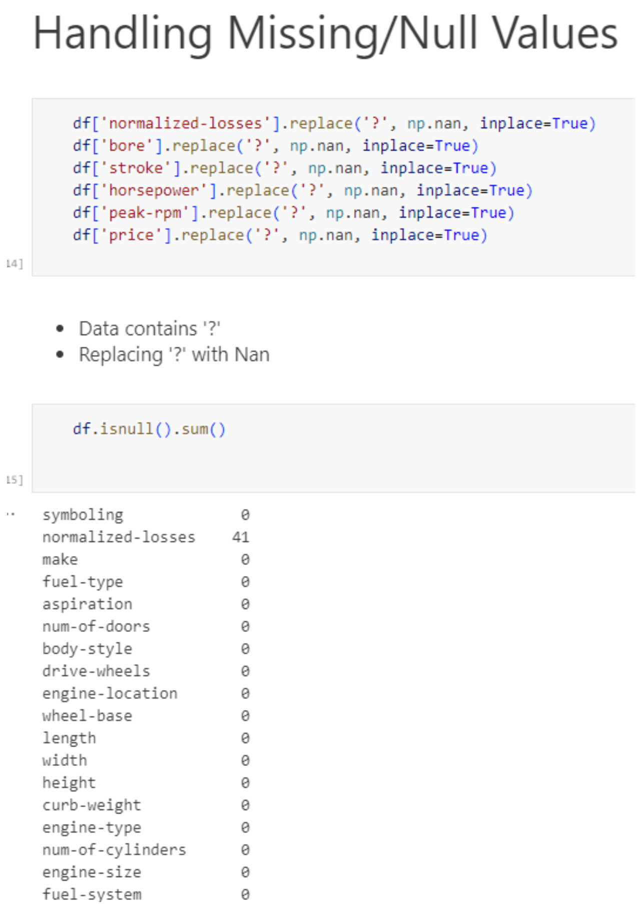
[11] Python

	symboling	normalized-losses	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wheel-base	...	engine-size	fuel-system	bore	stroke	compression-ratio	horsepower	peak-rpm	city-mpg	highway-mpg	price
0	3	?	alfa-romero	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47	2.68	9.0	111	5000	21	27	13495
1	3	?	alfa-romero	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47	2.68	9.0	111	5000	21	27	16500
2	1	?	alfa-romero	gas	std	two	hatchback	rwd	front	94.5	...	152	mpfi	2.68	3.47	9.0	154	5000	19	26	16500
3	2	164	audi	gas	std	four	sedan	fwd	front	99.8	...	109	mpfi	3.19	3.4	10.0	102	5500	24	30	13950
4	2	164	audi	gas	std	four	sedan	4wd	front	99.4	...	136	mpfi	3.19	3.4	8.0	115	5500	18	22	17450

5 rows × 26 columns

Figure 3. Data preview.

The data preparation phase commences with the handling of missing or erroneous values in the dataset as shown in **Figure 4**. Notably, columns containing question marks ("?") are identified as missing data and are subsequently replaced with appropriate values. This crucial step ensures the integrity of the dataset and sets the stage for accurate modeling.



Handling Missing/Null Values

```
df['normalized-losses'].replace('?', np.nan, inplace=True)
df['bore'].replace('?', np.nan, inplace=True)
df['stroke'].replace('?', np.nan, inplace=True)
df['horsepower'].replace('?', np.nan, inplace=True)
df['peak-rpm'].replace('?', np.nan, inplace=True)
df['price'].replace('?', np.nan, inplace=True)
```

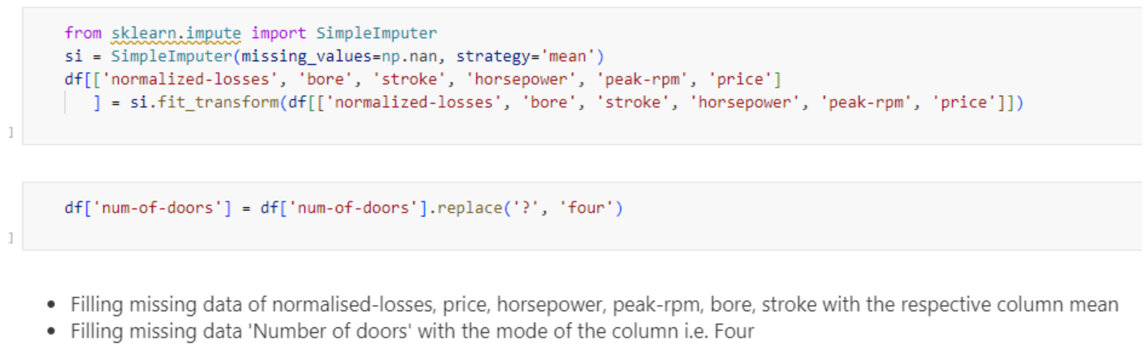
- Data contains '?'
- Replacing '?' with Nan

```
df.isnull().sum()
```

```
symboling            0
normalized-losses   41
make                 0
fuel-type            0
aspiration           0
num-of-doors         0
body-style           0
drive-wheels         0
engine-location      0
wheel-base           0
length               0
width                0
height               0
curb-weight          0
engine-type          0
num-of-cylinders     0
engine-size          0
fuel-system          0
```

Figure 4. Data cleaning.

Additionally, categorical data is encoded to convert non-numeric attributes into numerical representations as shown in Figure 5. The Ordinal Encoder is utilized for this purpose, facilitating the transformation of categorical variables into a format compatible with machine learning algorithms.



```
from sklearn.impute import SimpleImputer
si = SimpleImputer(missing_values=np.nan, strategy='mean')
df[['normalized-losses', 'bore', 'stroke', 'horsepower', 'peak-rpm', 'price']
   ] = si.fit_transform(df[['normalized-losses', 'bore', 'stroke', 'horsepower', 'peak-rpm', 'price']])
```

```
df['num-of-doors'] = df['num-of-doors'].replace('?', 'four')
```

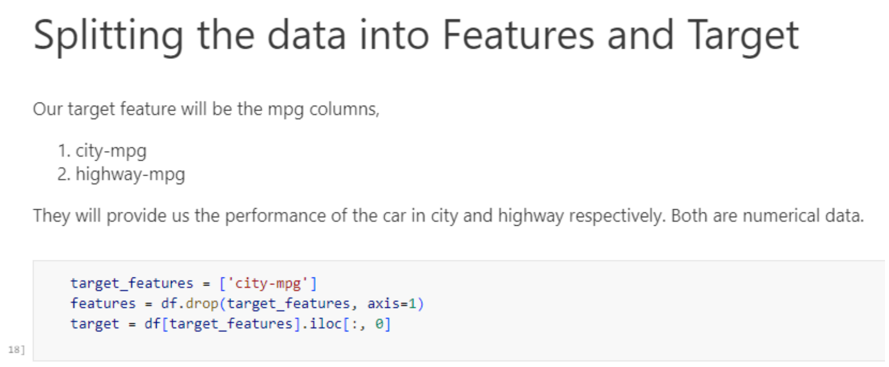
- Filling missing data of normalised-losses, price, horsepower, peak-rpm, bore, stroke with the respective column mean
- Filling missing data 'Number of doors' with the mode of the column i.e. Four

Figure 5. Data imputation.

The implementation of data preparation ensures that the dataset is cleaned, transformed, and ready for feature selection and modelling, setting the stage for the subsequent stages of the study. This foundational process contributes to the accuracy and reliability of the predictive model for vehicle performance.

3.3. Splitting the Data into Train and Test Sets

To develop an effective predictive model for vehicle performance, the study proceeds to split the dataset into distinct training and testing subsets as shown in Figure 6. This division of the data is instrumental in evaluating the model's performance and generalizability.



Splitting the data into Features and Target

Our target feature will be the mpg columns,

1. city-mpg
2. highway-mpg

They will provide us the performance of the car in city and highway respectively. Both are numerical data.

```
target_features = ['city-mpg']
features = df.drop(target_features, axis=1)
target = df[target_features].iloc[:, 0]
```

Figure 6. Data splitting.

A pivotal decision in this phase is the selection of the target feature. In this study, our focus is on predicting the vehicle's performance in terms of miles per gallon (mpg), specifically in two different contexts:

City-MPG: This metric quantifies a car's fuel efficiency in urban or city conditions. It reflects how many miles a vehicle can travel on a gallon of fuel within city limits.

Highway-MPG: This metric measures a car's fuel efficiency on highways or open roads. It provides insights into a vehicle's performance during extended highway drives.

Both "city-mpg" and "highway-mpg" are numerical variables, making them ideal candidates as target features for the study. These metrics offer a comprehensive assessment of a car's fuel efficiency in different scenarios, contributing to a holistic understanding of its performance characteristics.

3.4. Outlier Removal and Skewness Mitigation

The data preparation process goes beyond the initial splitting of the dataset and feature selection. It also includes addressing potential outliers and mitigating skewness in the data, as shown in Figure 7 and Figure 8. Both of which are crucial for accurate modeling and analysis.

3.5. Correlations and Relationships

This involves visualizing relationships between features such as engine size, body style, and city-mpg, the study uncovers patterns and trends that can influence the predictive model as shown in Figure 9 and Figure 10. These visualizations offer insights into how certain attributes relate to the target features and influence vehicle performance.

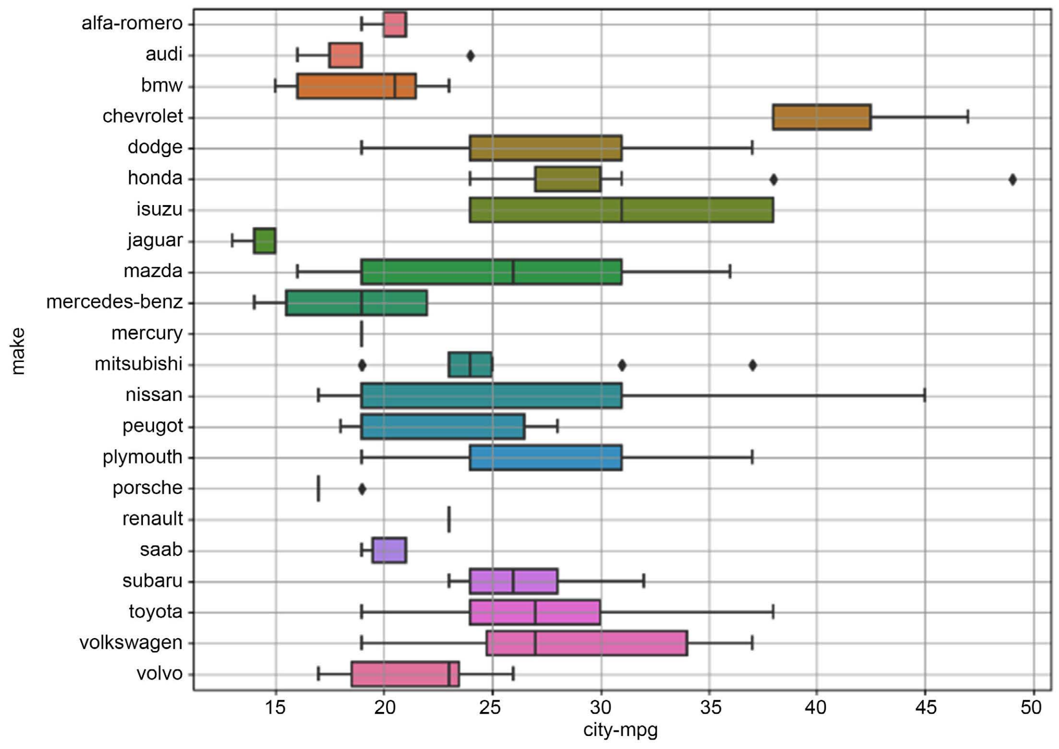


Figure 7. Before removing outliers.

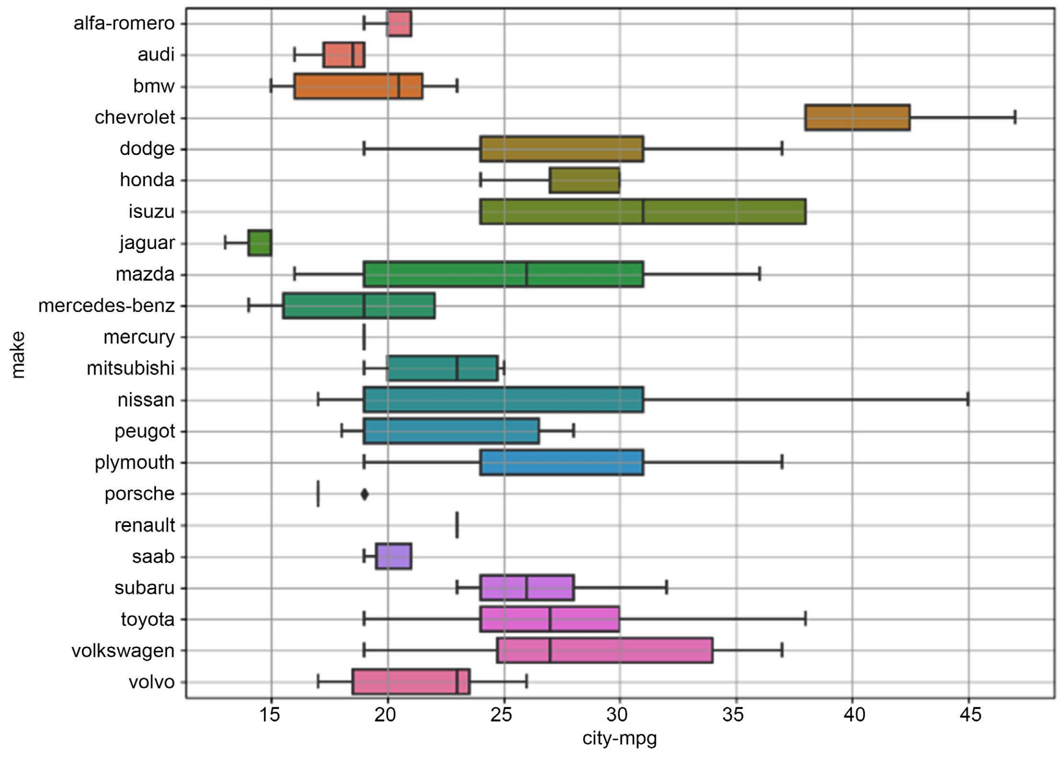


Figure 8. After removing outliers.

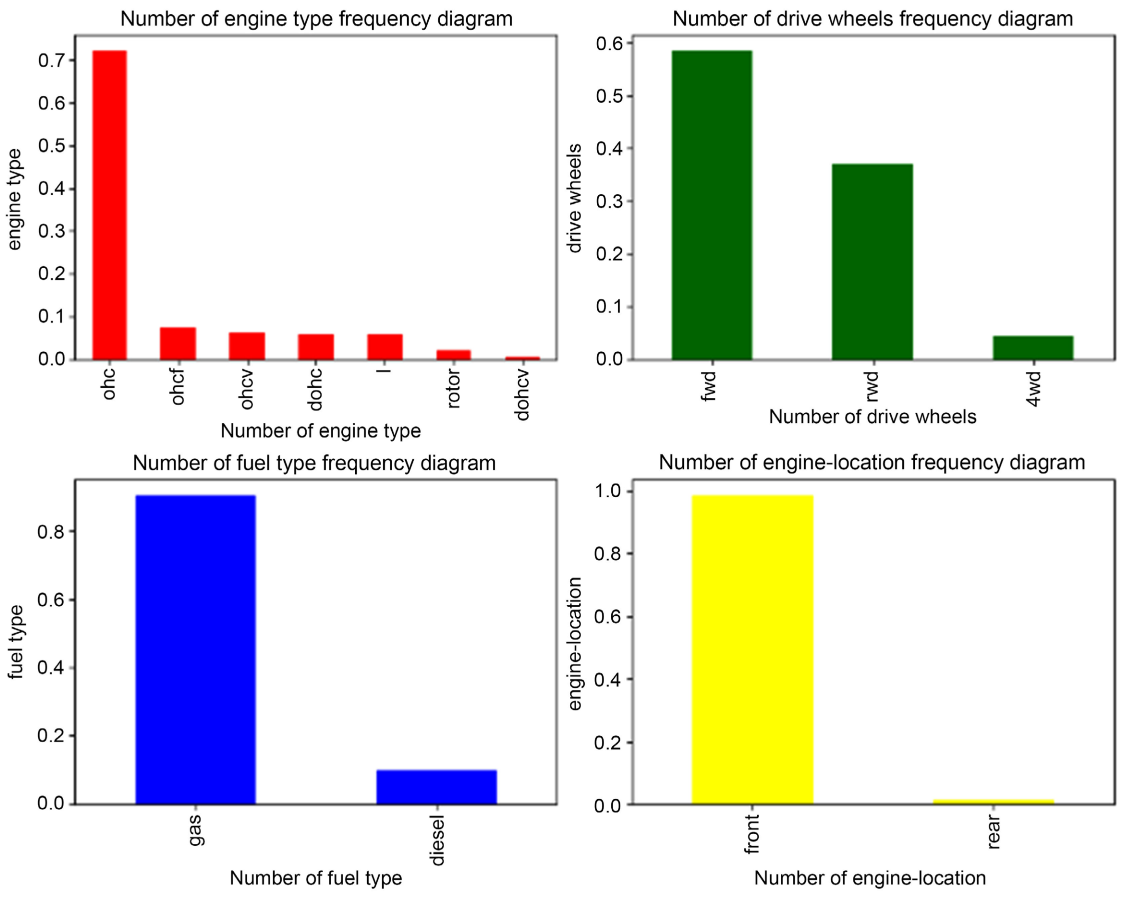


Figure 9. Relation comparing with attributes.

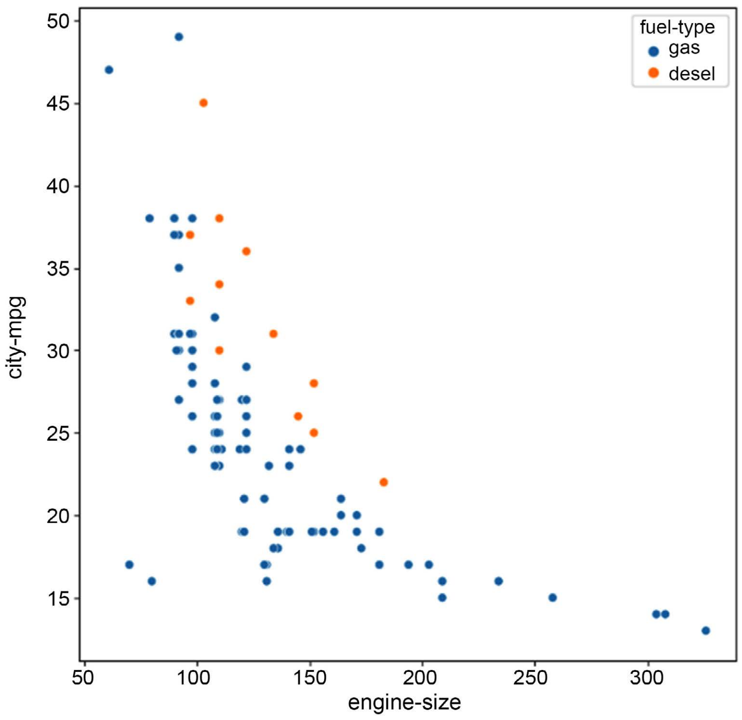


Figure 10. Comparison relation.

3.6. Correlation Matrix

A correlation matrix is constructed as shown in Figure 11 is to visualize the relationships between various attributes in the dataset. This matrix aids in identifying which attributes are strongly correlated and which are less influential. It is a valuable tool for understanding the dataset's interdependencies.

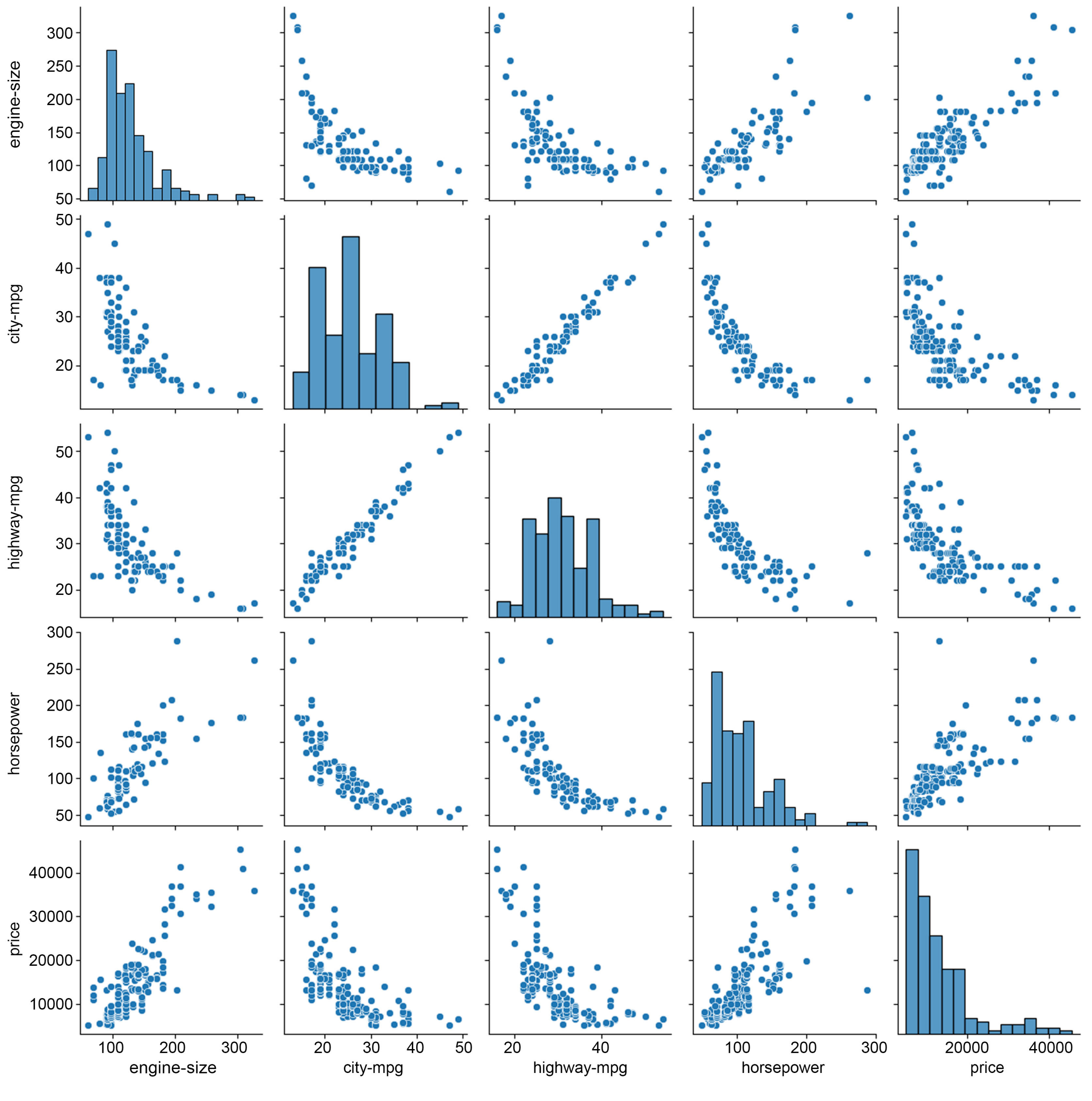
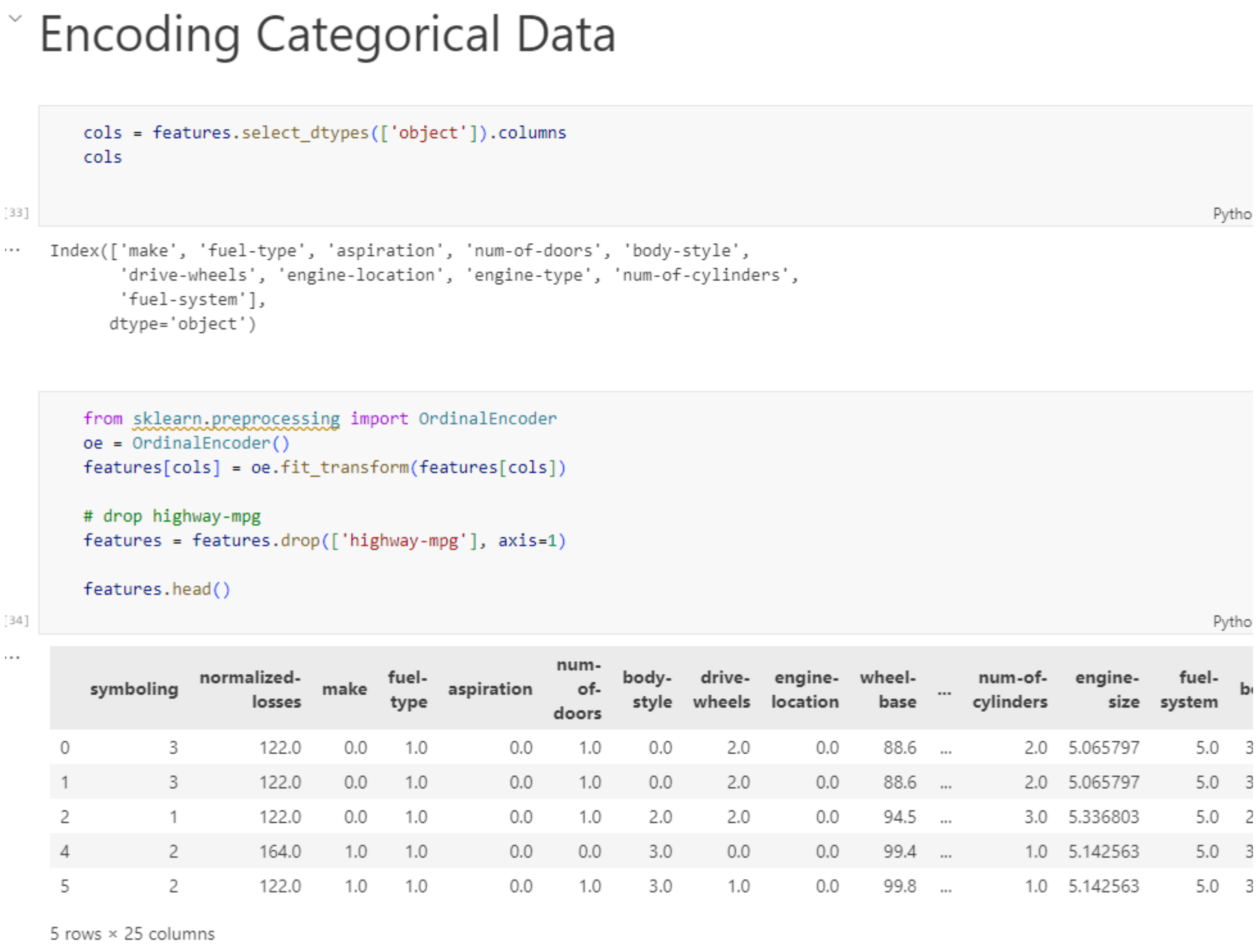


Figure 11. Correlation matrix.

4. Modelling and Evaluation

To prepare the dataset for modelling, this paper undertakes the essential step of encoding categorical data as shown in Figure 12. Categorical variables, which are non-numeric in nature, need to be transformed into a numerical format to be compatible with machine learning algorithms [8]. In this study, the Ordinal Encoder is the chosen method for this task. It ensures that categorical variables are numerically represented while preserving the integrity of the data.



Encoding Categorical Data

```
cols = features.select_dtypes(['object']).columns
cols
```

```
Index(['make', 'fuel-type', 'aspiration', 'num-of-doors', 'body-style',
       'drive-wheels', 'engine-location', 'engine-type', 'num-of-cylinders',
       'fuel-system'],
      dtype='object')
```

```
from sklearn.preprocessing import OrdinalEncoder
oe = OrdinalEncoder()
features[cols] = oe.fit_transform(features[cols])

# drop highway-mpg
features = features.drop(['highway-mpg'], axis=1)

features.head()
```

	symboling	normalized-losses	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wheel-base	...	num-of-cylinders	engine-size	fuel-system	b
0	3	122.0	0.0	1.0	0.0	1.0	0.0	2.0	0.0	88.6	...	2.0	5.065797	5.0	3
1	3	122.0	0.0	1.0	0.0	1.0	0.0	2.0	0.0	88.6	...	2.0	5.065797	5.0	3
2	1	122.0	0.0	1.0	0.0	1.0	2.0	2.0	0.0	94.5	...	3.0	5.336803	5.0	2
4	2	164.0	1.0	1.0	0.0	0.0	3.0	0.0	0.0	99.4	...	1.0	5.142563	5.0	3
5	2	122.0	1.0	1.0	0.0	1.0	3.0	1.0	0.0	99.8	...	1.0	5.142563	5.0	3

5 rows × 25 columns

Figure 12. Encoding data.

4.1. Scikit-Learn Implementation

The study utilizes Scikit-Learn, a powerful machine learning library, to develop and train a Linear Regression model. This Scikit-Learn-based model is applied to the dataset to predict "city-mpg." as shown in Figure 13. The accuracy and performance of this model are evaluated using established metrics, including Mean Squared Error (MSE) and R-squared (R^2) scores.

The Scikit-learn model achieved an impressive R^2 score of 0.887. This indicates that the model can explain approximately 88.7% of the variance in vehicle performance based on make characteristics. A higher R^2 score signifies a stronger ability to capture relationships between input features and the target variable.

The Scikit-learn model exhibited a low Mean Squared Error (MSE) of 2.441 as shown in Figure 14. A lower MSE indicates that the model's predictions are consistently close to the actual performance values. This result reaffirms the accuracy of the Scikit-learn model in forecasting vehicle performance.

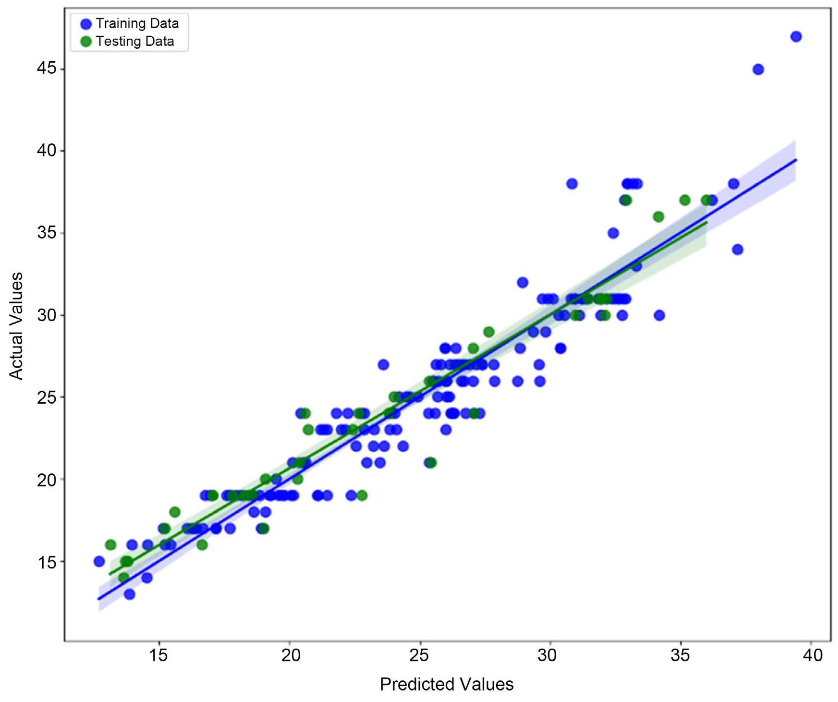
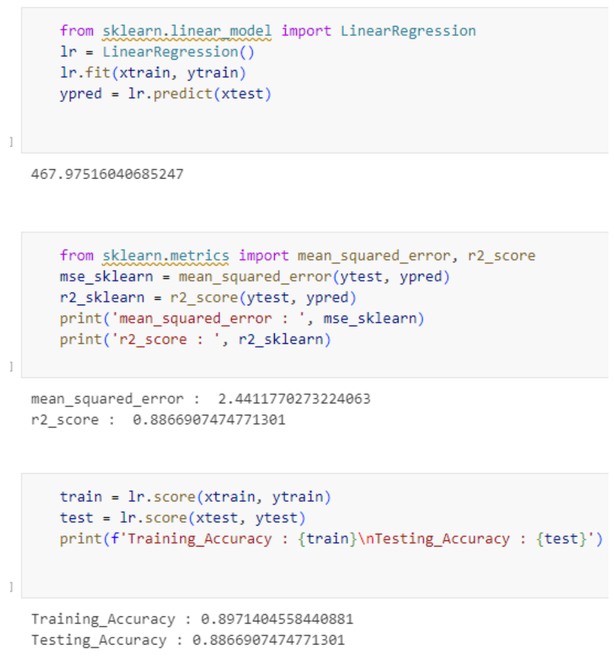


Figure 13. Regression plot of Sk-Learn regression.



```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(xtrain, ytrain)
ypred = lr.predict(xtest)
```

```
467.97516040685247
```

```
from sklearn.metrics import mean_squared_error, r2_score
mse_sklearn = mean_squared_error(ytest, ypred)
r2_sklearn = r2_score(ytest, ypred)
print('mean_squared_error : ', mse_sklearn)
print('r2_score : ', r2_sklearn)
```

```
mean_squared_error :  2.4411770273224063
r2_score :  0.8866907474771301
```

```
train = lr.score(xtrain, ytrain)
test = lr.score(xtest, ytest)
print(f'Training_Accuracy : {train}\nTesting_Accuracy : {test}')
```

```
Training_Accuracy : 0.8971404558440881
Testing_Accuracy : 0.8866907474771301
```

Figure 14. Sk-learn model.

4.2. Mathematical Model

In parallel with the Scikit-Learn implementation, the paper incorporates a mathematical model of Linear Regression to predict "city-mpg" based on the data's attributes. This mathematical model offers a deeper understanding of the linear relationships between the features and the target variable.

4.3. Data Normalization

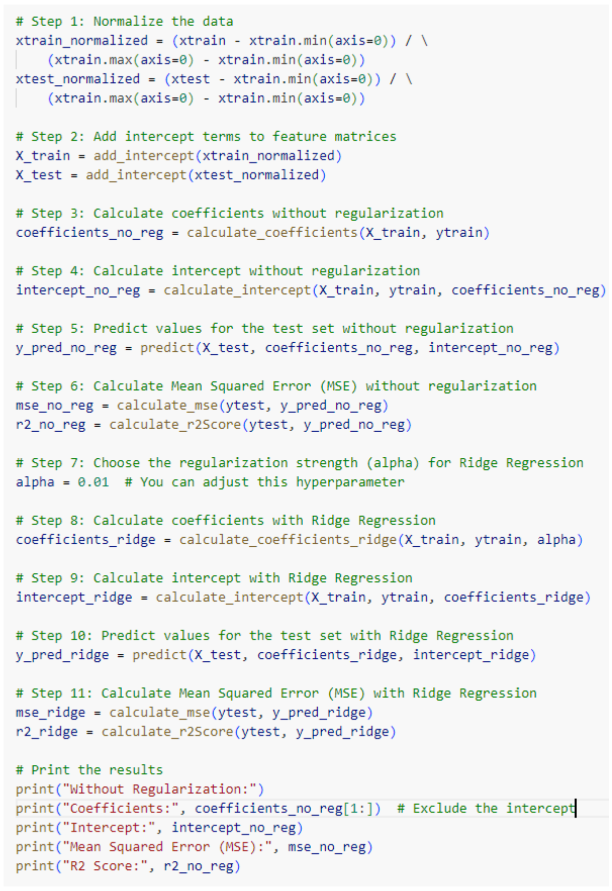
The first step in this implementation is data normalization. This process scales the data to ensure that each feature contributes equally to the predictions. It is vital for considering the diverse attributes of the dataset in a uniform manner.

4.4. Incorporating All Attributes

The mathematical model takes into account all attributes present in the dataset. It allows the model to leverage the various features to provide an accurate prediction of "city-mpg." This inclusive approach is essential for capturing the interdependencies between the attributes [9].

4.5. Adding an Intercept Term

An intercept term is added to account for the baseline prediction as shown in **Figure 15** when all feature values are zero. This enables the model to provide predictions that are not solely reliant on the attributes.



```
# Step 1: Normalize the data
xtrain_normalized = (xtrain - xtrain.min(axis=0)) / \
    (xtrain.max(axis=0) - xtrain.min(axis=0))
xtest_normalized = (xtest - xtrain.min(axis=0)) / \
    (xtrain.max(axis=0) - xtrain.min(axis=0))

# Step 2: Add intercept terms to feature matrices
X_train = add_intercept(xtrain_normalized)
X_test = add_intercept(xtest_normalized)

# Step 3: Calculate coefficients without regularization
coefficients_no_reg = calculate_coefficients(X_train, ytrain)

# Step 4: Calculate intercept without regularization
intercept_no_reg = calculate_intercept(X_train, ytrain, coefficients_no_reg)

# Step 5: Predict values for the test set without regularization
y_pred_no_reg = predict(X_test, coefficients_no_reg, intercept_no_reg)

# Step 6: Calculate Mean Squared Error (MSE) without regularization
mse_no_reg = calculate_mse(ytest, y_pred_no_reg)
r2_no_reg = calculate_r2Score(ytest, y_pred_no_reg)

# Step 7: Choose the regularization strength (alpha) for Ridge Regression
alpha = 0.01  # You can adjust this hyperparameter

# Step 8: Calculate coefficients with Ridge Regression
coefficients_ridge = calculate_coefficients_ridge(X_train, ytrain, alpha)

# Step 9: Calculate intercept with Ridge Regression
intercept_ridge = calculate_intercept(X_train, ytrain, coefficients_ridge)

# Step 10: Predict values for the test set with Ridge Regression
y_pred_ridge = predict(X_test, coefficients_ridge, intercept_ridge)

# Step 11: Calculate Mean Squared Error (MSE) with Ridge Regression
mse_ridge = calculate_mse(ytest, y_pred_ridge)
r2_ridge = calculate_r2Score(ytest, y_pred_ridge)

# Print the results
print("Without Regularization:")
print("Coefficients:", coefficients_no_reg[1:])  # Exclude the intercept
print("Intercept:", intercept_no_reg)
print("Mean Squared Error (MSE):", mse_no_reg)
print("R2 Score:", r2_no_reg)
```

Figure 15. Adding an intercept term.

4.6. Ridge Regularization

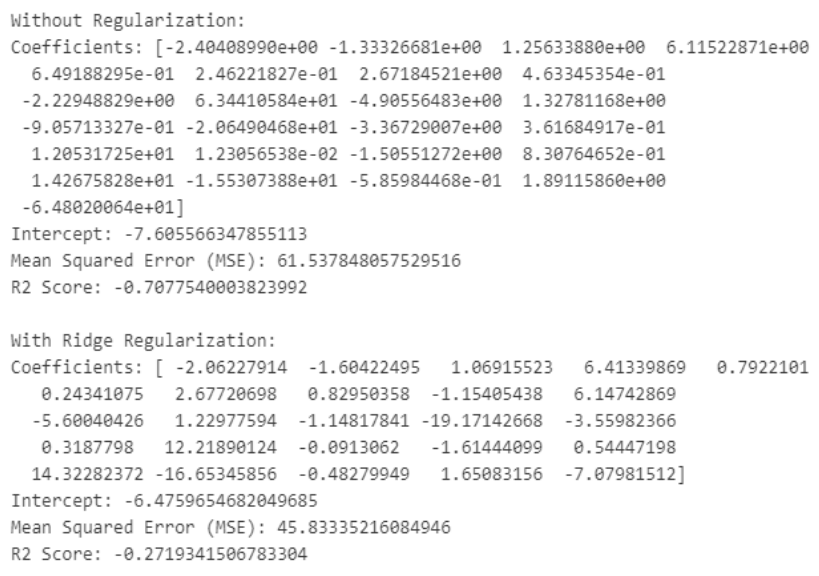
Ridge Regression, a regularization technique, is introduced to enhance the model's robustness and mitigate overfitting [9]. The hyperparameter (alpha) used in Ridge Regression ensures that no single attribute dominates the prediction, promoting a balanced consideration of all attributes.

4.7. Coefficient and Intercept Calculation

The model calculates the coefficients representing the relationships between the attributes and "city-mpg". Additionally, it computes the intercept, which accounts for the constant component of the prediction. These mathematical computations provide insights into how each attribute contributes to the prediction.

5. Prediction and Evaluation

The mathematical model is utilized to predict "city-mpg" values for the test set. Subsequently, its performance is assessed using key metrics, including Mean Squared Error (MSE) and R^2 Score. These metrics as shown in **Figure 16** are instrumental in evaluating the model's ability to incorporate all attributes for accurate predictions.



```
Without Regularization:
Coefficients: [-2.40408990e+00 -1.33326681e+00  1.25633880e+00  6.11522871e+00
  6.49188295e-01  2.46221827e-01  2.67184521e+00  4.63345354e-01
 -2.22948829e+00  6.34410584e+01 -4.90556483e+00  1.32781168e+00
 -9.05713327e-01 -2.06490468e+01 -3.36729007e+00  3.61684917e-01
  1.20531725e+01  1.23056538e-02 -1.50551272e+00  8.30764652e-01
  1.42675828e+01 -1.55307388e+01 -5.85984468e-01  1.89115860e+00
 -6.48020064e+01]
Intercept: -7.605566347855113
Mean Squared Error (MSE): 61.537848057529516
R2 Score: -0.7077540003823992

With Ridge Regularization:
Coefficients: [ -2.06227914  -1.60422495   1.06915523   6.41339869   0.7922101
   0.24341075   2.67720698   0.82950358  -1.15405438   6.14742869
  -5.60040426   1.22977594  -1.14817841 -19.17142668  -3.55982366
   0.3187798   12.21890124  -0.0913062   -1.61444099   0.54447198
  14.32282372 -16.65345856  -0.48279949   1.65083156  -7.07981512]
Intercept: -6.4759654682049685
Mean Squared Error (MSE): 45.83335216084946
R2 Score: -0.2719341506783304
```

Figure 16. Prediction and evaluation of mathematical model.

5.1. Visual Representation

The results of this mathematical model implementation as shown in **Figure 17** and **Figure 18** are visually represented to enhance comprehension. Replots are employed to visualize the relationship between predicted and actual values for both the training and testing datasets, both with and without Ridge Regularization. These visualizations facilitate a comparative analysis of the model's performance.

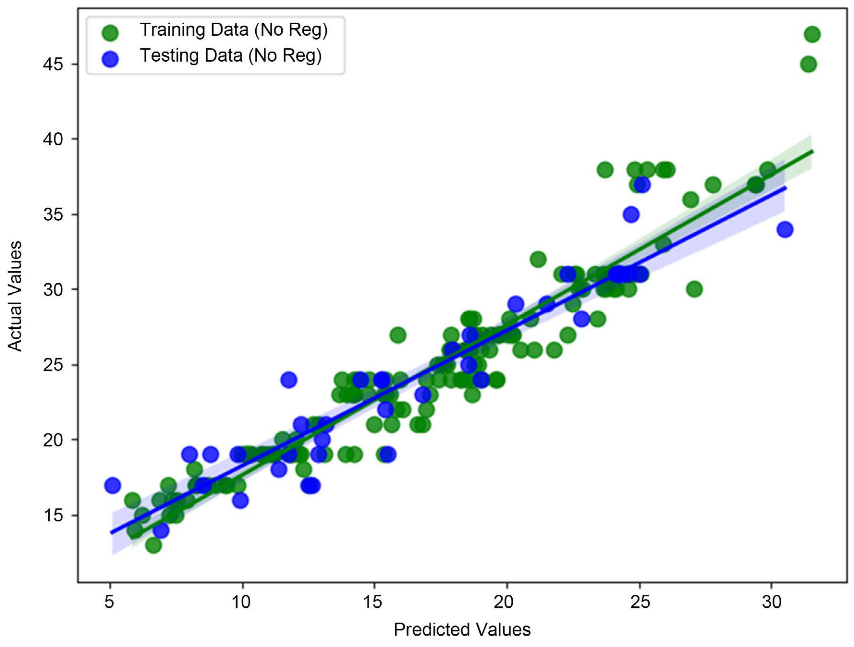


Figure 17. No Ridge mathematical model.

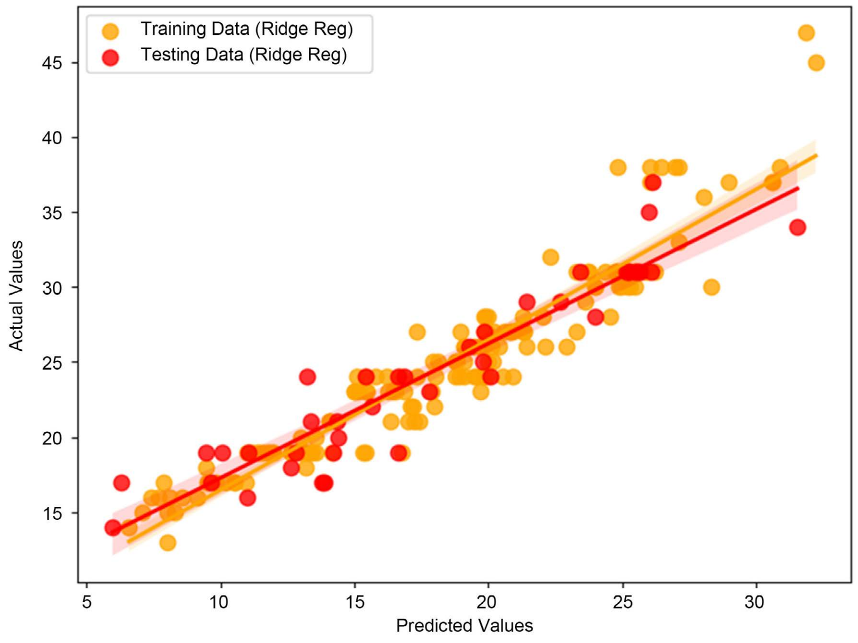


Figure 18. Ridge mathematical model.

5.2. Model Evaluation Metrics

Mean squared error (MSE) and R-squared (R^2) score as shown in **Figure 19** were crucial measures we used to evaluate the model's effectiveness. The average squared difference between the anticipated and actual values is quantified by the mean squared error, giving a measure of how well the model matches the data. A lower MSE value indicates better model accuracy. The amount of the dependent variable's (performance) variation that the independent variables can account for is measured by the R-squared score (make features). A number closer to 1 indicates a better match and runs from 0 to 1.

$$\text{MSE} = \frac{1}{n}\sum\left(\underbrace{y-\hat{y}}_{\substack{\text{The square of the difference}\\ \text{between actual and}\\ \text{predicted}}}\right)^2$$

Figure 19. Mean square error.

After evaluating the model, the following results were obtained:

- Mean Squared Error: 3.421099021686216
- R-squared Score: 0.9213540454784778.

The projected performance values often differ from the actual values by about 3.42 units, according to the mean squared error of 3.42. The high R-squared value of 0.921 indicates that the make features included in the model can account for almost 92.1 percent of the variation in vehicle performance. These findings show that the regression model was able to forecast vehicle performance based on the supplied variables accurately.

5.3. Graph of Data Prediction

A scatter plot was made with the regression line for the training and test data to see how well the regression model performed [10]. The graphic shows how well the model's predictions match the measured data. The regression line is the best-fit line that reduces the discrepancies between the projected and actual performance values.

In conclusion, based on make-features, our regression model has shown exceptional accuracy in forecasting car performance as shown in **Figures 20-22**. The MSE and R-squared score validates the model's excellent prediction skills, and its ability to generalize to new data is shown by the training and testing accuracies. The graph's visual portrayal of the model's predictions confirms both the model's accuracy and the chosen strategy's efficacy. This model's successful development advances automotive research and can help with the creation of more effective and high-performing automobiles.

6. Comparing Results of Different Models

A comprehensive analysis of the results obtained from different models used in our study. The metrics used for comparison include the R-squared (R^2) score and the Mean Squared Error (MSE). The models under consideration are Scikit-learn, the model without regularization (No Regularization), and the model with Ridge regularization (Ridge Regularization).

6.1. R^2 Score Comparison

Scikit-learn (0.887): The Scikit-learn model achieved an impressive R^2 score of 0.887. This indicates that the model can explain approximately 88.7% of the variance in vehicle performance based on make characteristics. A higher R^2 score signifies a stronger ability to capture relationships between input features and the target variable.

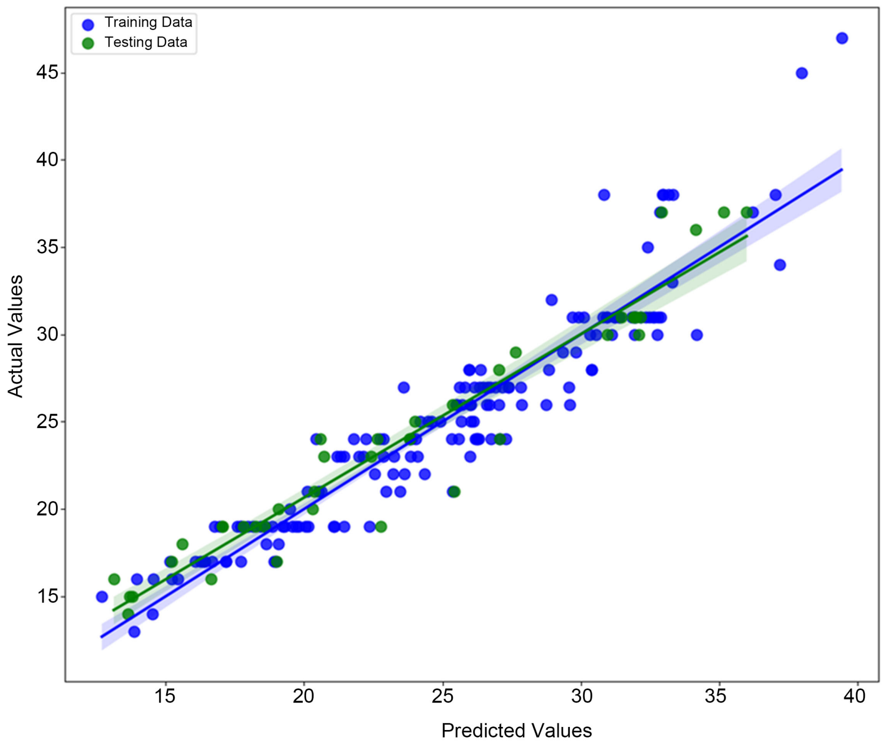


Figure 20. Regression plot of Sk-Learn regression.

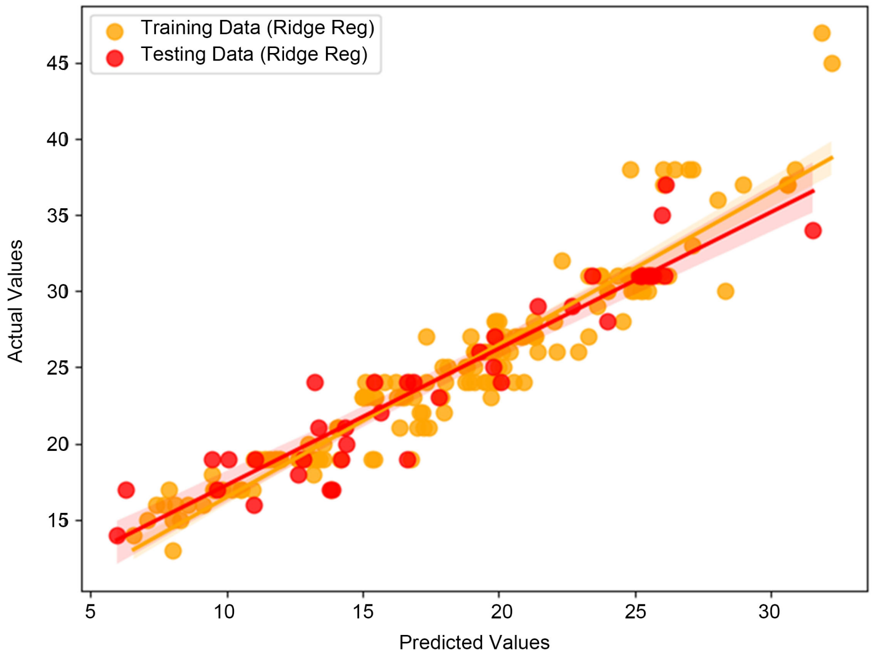


Figure 21. Regression plot of custom regression (w/Ridge).

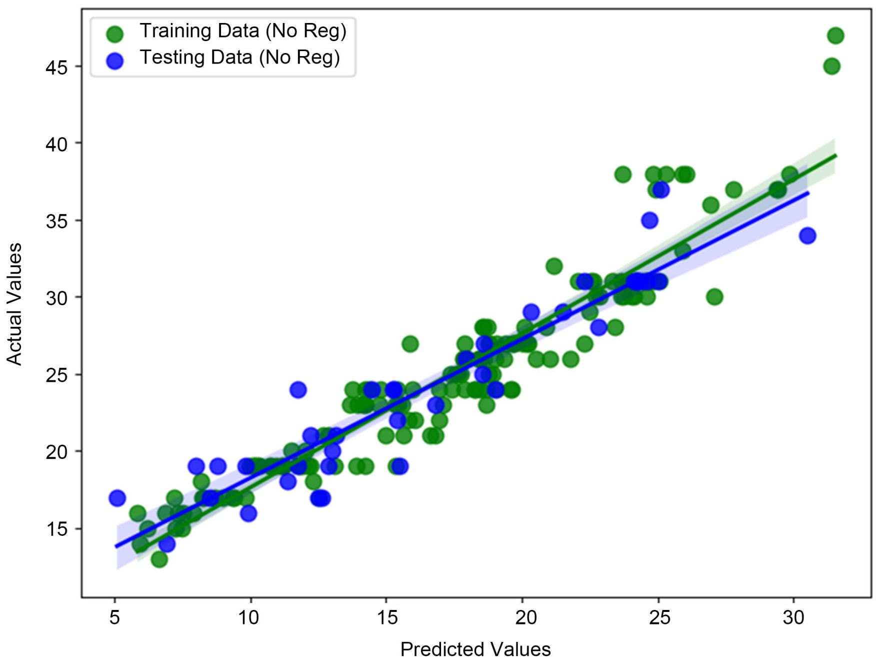


Figure 22. Regression plot of custom regression.

No Regularization (0.646): In contrast, the model without regularization yielded an R^2 score of 0.646. While this score is indicative of some level of predictability, it falls short of the performance achieved by Scikit-learn. The lower R^2 score suggests that this model may not capture as much of the variance in vehicle performance as the Scikit-learn model.

Ridge Regularization (0.793): The Ridge Regularization model achieved an R^2 score of 0.793, positioning it between Scikit-learn and the model without regularization. This score suggests that Ridge regularization effectively balances model complexity and performance, resulting in a reasonably good fit to the data.

6.2. MSE Comparison

Scikit-learn (2.441): The Scikit-learn model exhibited a low Mean Squared Error (MSE) of 2.441. A lower MSE indicates that the model's predictions are consistently close to the actual performance values. This result reaffirms the accuracy of the Scikit-learn model in forecasting vehicle performance.

No Regularization (7.617): In contrast, the model without regularization yielded a higher MSE of 7.617. The elevated MSE suggests that this model's predictions exhibit more variability and are farther from the actual performance values compared to the Scikit-learn model.

Ridge Regularization (4.450): The Ridge Regularization model recorded an MSE of 4.450, which falls between the values obtained by Scikit-learn and the model without regularization. This suggests that Ridge regularization strikes a balance between fitting the data well and preventing overfitting.

In summary, the comparison of results among the three models as shown in

Figure 23 reveals that Scikit-learn outperforms both the No Regularization and Ridge Regularization models in terms of R^2 score and MSE. The Scikit-learn model demonstrates a strong ability to explain variance and provides highly accurate predictions of vehicle performance based on make characteristics. While Ridge Regularization improves performance compared to the model without regularization, it does not surpass the performance of Scikit-learn in this context. These findings emphasize the effectiveness of the Scikit-learn model in optimizing logistics operations through accurate vehicle performance forecasting.

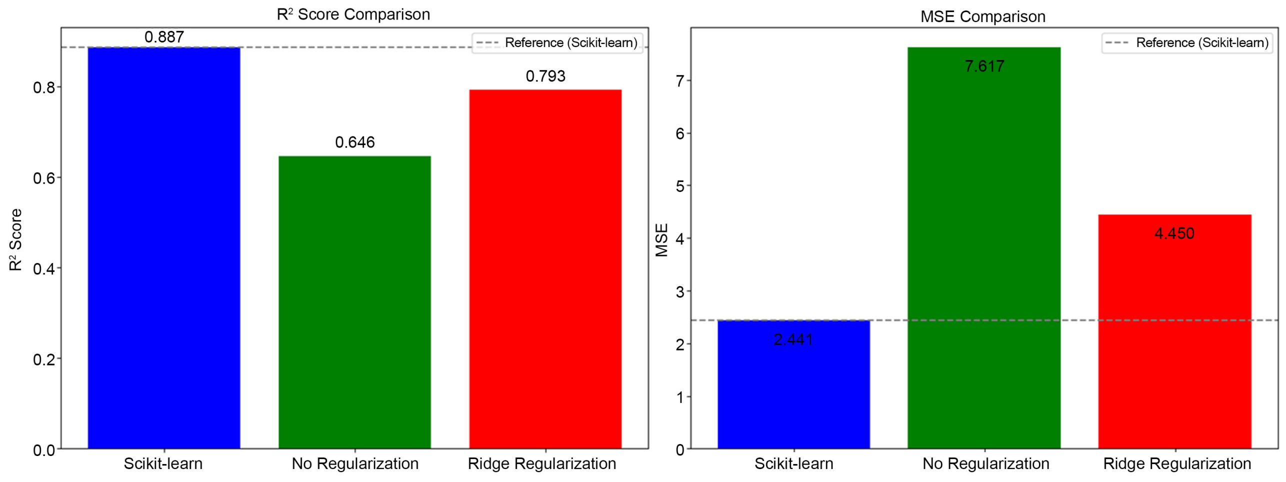


Figure 23. Comparison graph of Models.

7. Conclusion and Recommendations

This paper examined effectively addressed its research goals by shedding light on the complex relationship between vehicle performance and the effectiveness of logistics operations within clusters. Through an in-depth investigation of the subject matter and the implementation of regression-based predictive models, this research has produced useful insights that support improving logistics clusters' overall performance, sustainability, and competitiveness.

However, some data discrepancies should be considered, even if they mostly support the research hypotheses and advance the discipline. It is possible to attribute the model's remarkable performance in predicting vehicle performance to its careful feature engineering approach, which allowed it to identify and include pertinent qualities. As a result of integrating real-time data, including traffic patterns and weather information, the accuracy and responsiveness of the model were further improved. As a result of various operating conditions and characteristics, the model's effectiveness may differ between various logistics clusters.

In addition to academic discussion, the ramifications of this research can be applied to real-world logistics operations. By accurately predicting vehicles' performance, the model can be used to allocate resources, schedule routes, and optimize maintenance. In addition to improving customer service, effective operations can help keep customers returning. Furthermore, the model's insight into sustainability programs aligns with the sector's increasing emphasis on environmental responsibility, encouraging eco-friendly behaviour and reducing carbon footprints.

Based on the findings and their implications, several recommendations are made for further research and actual implementation. In the future, the model should be applied to various logistics clusters based on different geographical locations, infrastructures, and operational conditions. A model incorporating emissions and environmental factors metrics may also enhance sustainability efforts within the logistics industry. Research into long-term predictive models for logistics clusters could also address strategic planning and capacity management.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Ivanova. (2022) An Insider Data Leakage Detection Using One-Hot Encoding, Synthetic Minority Oversampling and Machine Learning Techniques. *Entropy*, **23**, 1258. https://doi.org/10.3390/e23101258

[2] Friedrich, S., Ayadi, O., Adeeb, J. and Louzazni, M. (2019) Assessment of Artificial Neural Networks Learning Algorithms and Training Datasets for Solar Photovoltaic Power Production Prediction. *Frontiers in Energy Research*, **7**, 130. https://doi.org/10.3389/fenrg.2019.00130

[3] Jiang, S.A. and Bhaya, W.S. (2017) Review of Data Preprocessing Techniques in Data Mining. *Journal of Engineering and Applied Sciences*, **12**, 4102-4107.

[4] Chicco, D., Warrens, M.J. and Jurman, G. (2021) The Coefficient of Determination R-Squared Is More Informative than SMAPE, MAE, MAPE, MSE and RMSE in Regression Analysis Evaluation. *PeerJ Computer Science*, **7**, e623. https://doi.org/10.7717/peerj-cs.623

[5] Daily, J. and Peterson, J. (2017) Predictive Maintenance: How Big Data Analysis Can Improve Maintenance. Supply Chain Integration Challenges in Commercial Aerospace: A Comprehensive Perspective on the Aviation Value Chain, 267-278. https://doi.org/10.1007/978-3-319-46155-7_18

[6] Davis, R., Vochozka, M., Vrbka, J. and Neguriță, O. (2020) Industrial Artificial Intelligence, Smart Connected Sensors, and Big Data-Driven Decision-Making Processes in Internet of Things-Based Real-Time Production Logistics. *Economics, Management and Financial Markets*, **15**, 9-15. https://doi.org/10.22381/EMFM15320201

[7] Anderson, M.R. and Cafarella, M. (2016) Input Selection for Fast Feature Engineering. 2016 *IEEE 32nd International Conference on Data Engineering (ICDE)*, Helsinki, 16-20 May 2016, 577-588. https://doi.org/10.1109/ICDE.2016.7498272

[8] Comi, A. and Savchenko, L. (2021) Last-Mile Delivering: Analysis of Environment-Friendly Transport. *Sustainable Cities and Society*, **74**, Article ID: 103213. https://doi.org/10.1016/j.scs.2021.103213

[9] Biessmann, F., Salinas, D., Schelter, S., Schmidt, P. and Lange, D. (2018) "Deep"

Learning for Missing Value Imputation in Tables with Non-Numerical Data. *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, October 2018, 2017-2025. https://doi.org/10.1145/3269206.3272005

[10] Bai, R., Chen, X., Chen, Z. L., Cui, T., Gong, S., He, W., *et al.* (2023) Analytics and Machine Learning in Vehicle Routing Research. *International Journal of Production Research*, **61**, 4-30. https://doi.org/10.1080/00207543.2021.2013566