Applied Machine Learning (CSE – 845)

Submitted By

Abdur Rehman

Ahmed Mujtaba

Ayesha Ali

Muhammad Abdur Rehman

Submitted to

Dr. Muhammad Tariq Saeed

School of Interdisciplinary Engineering and Sciences, National University of Sciences and Technology, Islamabad

October 2025.

Table of Contents

Link to the Colab notebook:

https://github.com/biopype/applied-machine-learning/blob/main/assignment-02/code-assignment-02-AML.ipynb

Introduction:

In today's fast-paced logistics and delivery industry, accurately predicting delivery times has become a critical challenge for companies striving to maintain customer satisfaction and operational efficiency. Delivery services face numerous unpredictable factors, including traffic conditions, weather changes, varying distances, and driver experience levels that directly impact how long a delivery takes. When companies cannot accurately estimate delivery times, they face increased customer complaints, inefficient resource allocation, and poor route planning decisions. Currently, many delivery organizations rely on simple average calculations or rough estimates based solely on distance, which often fail to account for the complex interplay of multiple variables affecting delivery performance. This leads to overpromising delivery windows, frustrated customers, and wasted operational resources. There is a clear need for a data-driven approach that can analyze delivery data and predict delivery times with reasonable accuracy by considering all relevant factors simultaneously.

To address this challenge, we use a dataset of recent delivery operations (in CSV format) to build and evaluate several machine learning models. Specifically, we apply multivariate linear regression to analyze and estimate the accurate delivery time prediction.

Problem Statement:

Evaluate the following hypothesis using a multivariate linear regression model.

i. Including all available features in the linear regression model will provide the most accurate predictions.
ii. A linear regression model built using only features that are significantly correlated with the target variable will outperform the model using all features, as it focuses on the most relevant predictors.
iii. Excluding features that exhibit multicollinearity from the linear regression model will yield better performance compared to both the all-features model and the correlated features model.

Objective:

The goal of this analysis is to identify which approach (all features, correlated features, or multicollinearity-reduced features) results in the most accurate prediction of delivery time and to understand how feature selection impacts model performance.

Methodology

The analysis was conducted in Python using libraries such as **pandas, seaborn, matplotlib, scikit-learn, and statsmodels.** The study involved four main stages:

- Data loading and exploration
- Data preprocessing and visualization
- Model building and training under three hypotheses
- Model evaluation and comparison using performance metrics.

These libraries were chosen due to their reliability and popularity in machine learning and data analytics.

Library	Main Purpose	Key Functions Used
pandas	Data handling	read_csv(), DataFrame, info(), describe()
sklearn.preprocessing	Data transformation	LabelEncoder
sklearn.impute	Fill missing data	SimpleImputer
sklearn.model_selection	Split data	train_test_split
sklearn.linear_model	ML algorithms	LinearRegression
sklearn.metrics	Evaluate models	r2_score
seaborn	Statistical plots	pairplot(), scatterplot()
matplotlib	General plotting	figure(), plot(), xlabel(), title(), show()

Data Analysis and Preprocessing:

4.1 Loading and Inspecting the Dataset

We began by loading the delivery dataset using the pandas library and examining its structure. The dataset contains 100 delivery records with 14 variables, including Miles Travelled, Number of Deliveries, Petrol Consumed, Traffic Conditions, Weather Conditions, and **Time Taken,** which serves as our **target variable** for prediction. Initial inspection using data information and descriptive statistics revealed the overall structure of the dataset. We identified several missing values scattered across different columns and obtained statistical summaries showing the mean, standard deviation, minimum, maximum, and quartile values for each numerical variable. This gave us an understanding of the data distribution and helped identify any potential outliers.

4.2 Encoding Categorical Variables

The Weather Condition column contained text values (Clear, Rainy, Snowy, Windy) that needed to be converted to a numerical format since machine learning models require numerical input. We used LabelEncoder from scikit-learn to automatically convert these weather categories into numbers: Clear became 0, Rainy became 1, Snowy became 2, and Windy became 3. To maintain transparency and ensure we could interpret results later, we created a mapping table that clearly showed which number corresponds to which weather condition. This encoding transformation allowed our regression model to process weather information mathematically while preserving the ability to understand what each encoded value represents.

Category	Encoded Value
Clear	0
Rainy	1
Snowy	2
Windy	3

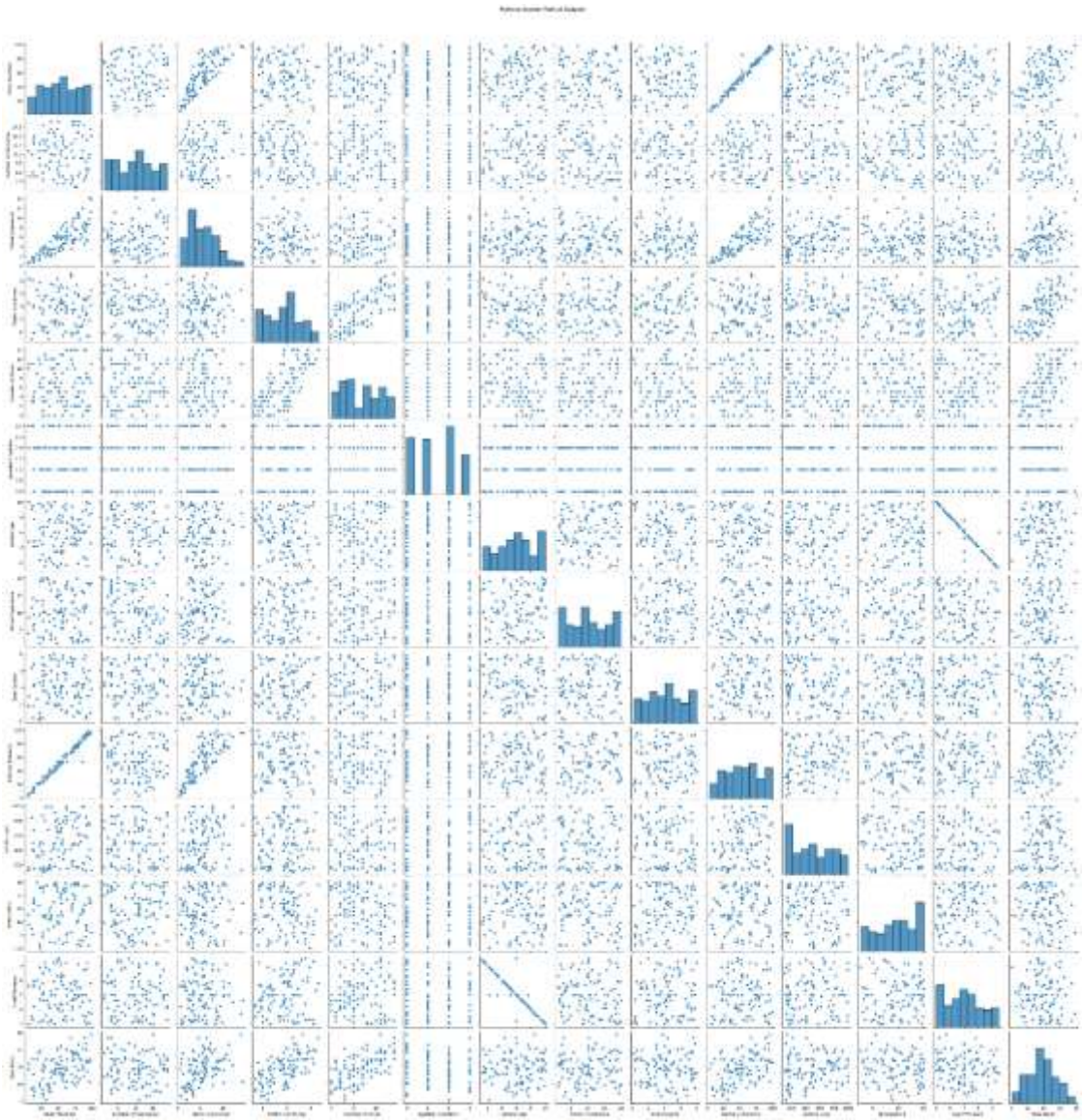
4.3 Handling Missing Values

Addressing missing data was critical for model accuracy. We implemented mean imputation, which replaces each missing value with the average of its respective column. For example, if a delivery record was missing the "Petrol Consumed" value, it was filled with the average petrol

consumption across all other deliveries. This approach is appropriate for our dataset because most variables follow normal distributions, making the mean a reasonable representative value. After imputation, we verified that all columns now had 100 complete entries with no missing values remaining. The dataset was now clean and ready for modeling.

4.4 Exploratory Data Analysis

To understand relationships between variables and identify patterns in our data, we created a comprehensive pairplot visualization using seaborn. This generated a 14×14 grid where each cell showed the relationship between two variables. The diagonal displayed histograms showing individual variable distributions, while other cells contained scatter plots comparing variable pairs. The visualization revealed several important insights. Miles Travelled and Delivery Distance showed an almost perfect linear relationship, indicating they are highly correlated and essentially measure the same thing. Petrol Consumed showed a strong positive relationship with Miles Travelled, which makes logical sense. Most variables displayed approximately normal bell-shaped distributions, which is favorable for linear regression modeling. Many variable pairs showed scattered dots with no clear pattern, indicating they are relatively independent and contribute unique information. However, the pairplot also identified potential multicollinearity issues, particularly between Miles Travelled and Delivery Distance. These findings would be crucial for testing our three hypotheses, especially Hypothesis 3, which examines the impact of removing multicollinear features. With our data now cleaned, encoded, imputed, and thoroughly explored, we were ready to proceed with building and testing our regression models. The preprocessing steps ensured data quality, while the exploratory analysis provided insights that would guide our hypothesis testing approach.



Hypothesis Testing and Model Building:

5.1 Hypothesis 1: All Features Model

5.1.1 Hypothesis Statement

The first hypothesis proposes that including all available features in the linear regression model will provide the most accurate predictions. This approach assumes that more information leads to better predictions, and that every variable in the dataset, regardless of its individual correlation with delivery time, contributes something valuable to the overall prediction accuracy.

The logic behind this hypothesis is straightforward: delivery time is influenced by multiple complex factors that may interact in non-obvious ways. Even features that appear weakly correlated individually might become important when combined with other variables. By including all 13 input features (Miles Travelled, Number of Deliveries, Petrol Consumed, Traffic Conditions, Number of Stops, Weather Condition, Vehicle Age, Driver Experience, Road Quality, Delivery Distance, Vehicle Load, Temperature, and Fuel Efficiency), we give the model access to the complete information available in our dataset.

5.1.2 Model Implementation

For Hypothesis 1, we built a baseline linear regression model using every feature except the target variable. We separated our data into features (X) containing all 13 independent variables, and the target (y) containing Time Taken. The dataset was then split into training and testing sets using an 80-20 split, meaning 80% of the data (80 records) would be used to train the model, while the remaining 20% (20 records) would be held back to evaluate how well the model performs on unseen data.

We initialized a Linear Regression model from scikit-learn and trained it on the training data. The model learned the mathematical relationships between all input features and delivery time by calculating optimal coefficients (weights) for each variable. During training, the algorithm minimized the difference between predicted and actual delivery times across all 80 training examples.

5.1.3 Model Training Process

The linear regression algorithm works by finding the best-fit line (or hyperplane in multi-dimensional space) through the data points. It calculates a coefficient for each feature that represents the degree to which that feature influences delivery time. For example, it might learn that each additional mile traveled adds approximately 0.2 minutes to delivery time, while each additional stop adds 1.5 minutes.

The model essentially creates an equation: Time Taken = ($coefficient_1$ × Miles Travelled) + ($coefficient_2$ × Number of Deliveries) + ($coefficient_3$ × Petrol Consumed) + ... and so on for all 13

features, plus a constant (intercept) term. The training process adjusts these coefficients to minimize prediction errors across the training dataset.

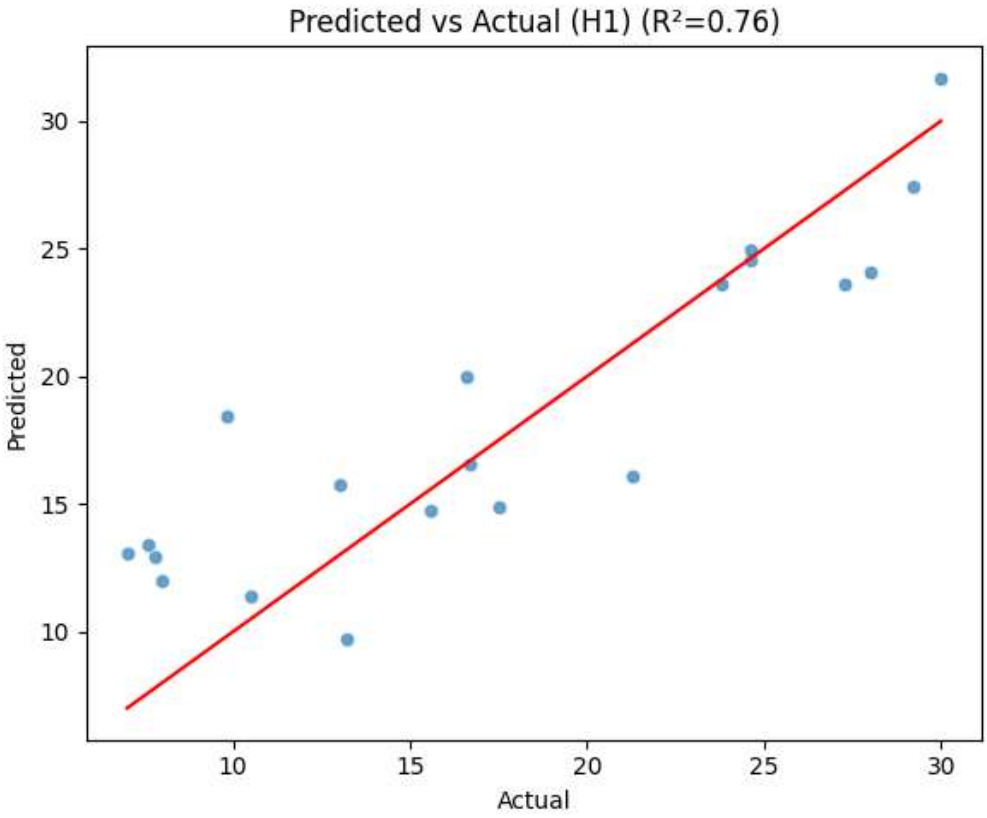
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression

# 'Time Taken' is the target variable and the rest are features
X = data.drop('Time Taken', axis=1)
y = data['Time Taken']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize and train the Linear Regression model
model = LinearRegression()
model.fit(X_train, y_train)

print("Model training for hypothesis 1 complete.")
```

5.1.4 Prediction and Results

Once trained, we used the model to make predictions on the test set - the 20 delivery records it had never seen before. For each test record, the model took the 13 input features and calculated its prediction for delivery time using the equation it learned during training. These predictions were then compared with the actual delivery times to evaluate model performance.

The model achieved an R^2 score of 0.75, meaning it successfully explained 75% of the variation in delivery times. This indicates reasonably good predictive accuracy. The predicted vs actual scatter plot showed most points clustering near the diagonal reference line, confirming that predictions were generally close to reality, though some scatter was visible, indicating the model wasn't perfect.

Hypothesis 1 Performance Metrics	
MAE	3.03
MSE	15.46
RMSE	3.80
R^2	0.75



5.2 Hypothesis 2: Significantly Correlated Features Model

5.2.1 Hypothesis Statement

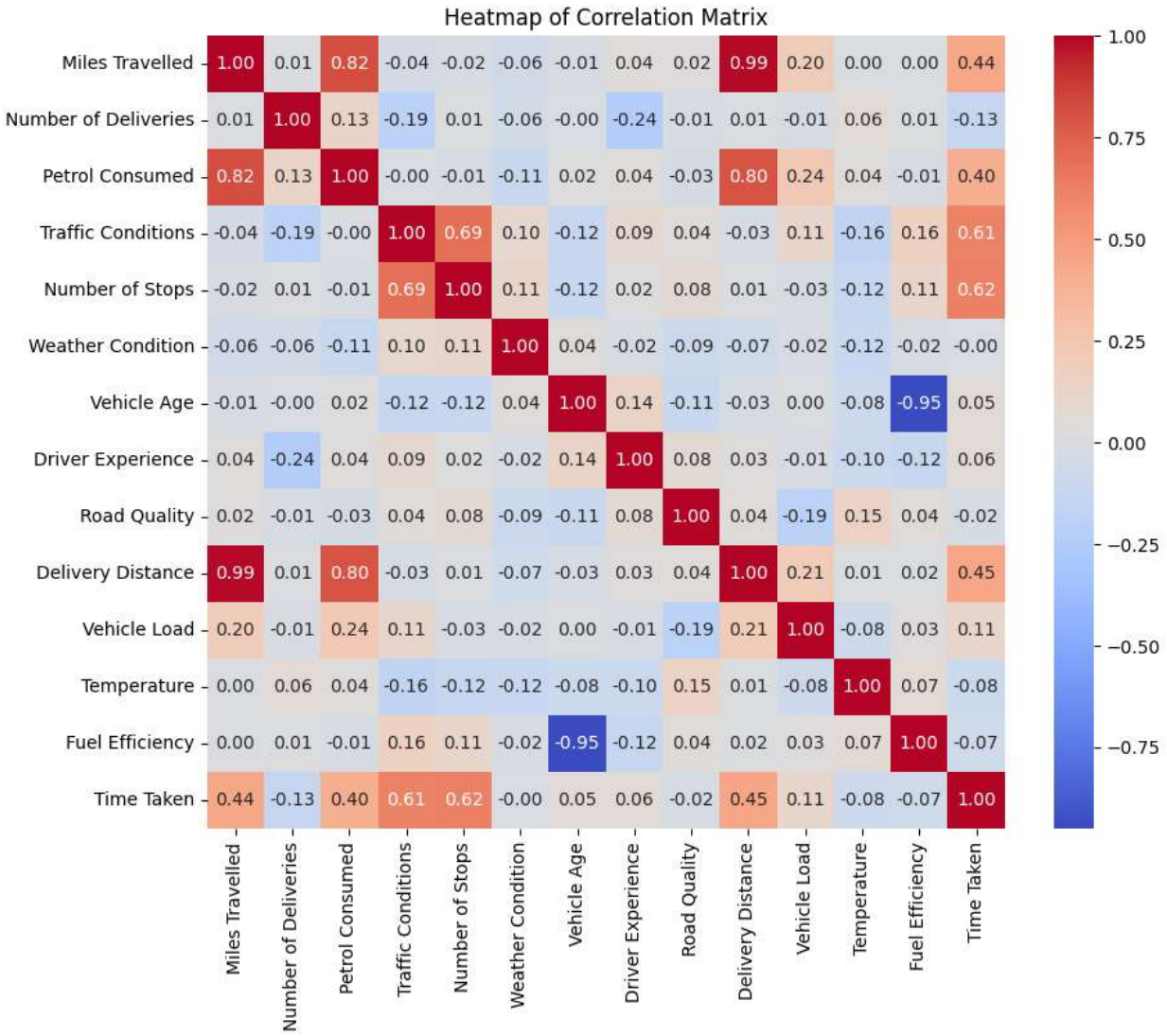
The second hypothesis proposes that a linear regression model built using only features that are significantly correlated with the target variable will outperform the model using all features, as it focuses on the most relevant predictors. This approach suggests that including weakly related or irrelevant features adds noise to the model rather than useful information, and that a more selective feature set will improve prediction accuracy.

The reasoning behind this hypothesis is based on the principle of feature relevance. Not all variables in a dataset contribute equally to predictions. Some features may have weak or negligible relationships with delivery time, and including them could introduce unnecessary complexity and noise. By filtering out weakly correlated features and retaining only those with strong statistical relationships to the target variable, we create a more focused model that concentrates on the truly important predictors.

This approach also addresses the "curse of dimensionality" - when too many features are included, the model may overfit to training data noise rather than learning genuine patterns. A streamlined feature set with only significant predictors should theoretically produce a cleaner, more generalizable model.

5.2.2 Feature Selection Process

To identify significantly correlated features, we calculated the Pearson correlation coefficient between each input variable and Time Taken (our target variable). The correlation coefficient ranges from -1 to +1, where values closer to -1 or +1 indicate strong relationships, while values near 0 indicate weak relationships.



We applied a correlation filter with a **threshold of 0.2** (AlOmari, Yaseen, & Al-Betar, 2023). This means only features with absolute correlation values greater than 0.2 with Time Taken were selected. This threshold helped us identify variables that had meaningful relationships with delivery time while excluding those with negligible influence.

The correlation analysis revealed five variables that met our threshold:

- Number of Stops
- Traffic Conditions
- Delivery Distance
- Miles Travelled
- Petrol Consumed

These five features showed the strongest statistical relationships with delivery time, indicating they are the most relevant predictors. For visualization and better understanding of these relationships, we created a correlation heat map that displayed the correlation coefficients between all variables in a color-coded matrix. We built our Hypothesis 2 model based solely on these five significant variables, as they demonstrated a considerable influence on delivery time. By focusing on these meaningful predictors and excluding the remaining eight weakly correlated features, we aimed to create a more efficient and focused prediction model.

5.2.3 Model Implementation

After identifying the significantly correlated features, we created a new feature set (X) containing only these selected variables, while maintaining the same target variable (y) as Time Taken. The reduced feature set was then split into training and testing sets using the same 80-20 split ratio to ensure fair comparison with Hypothesis 1.

	Feature	Correlation with Target (Time Taken)
4	Number of Stops	0.621255
3	Traffic Conditions	0.613492
9	Delivery Distance	0.447485
0	Miles Travelled	0.441436

2	Petrol Consumed	0.400869
10	Vehicle Load	0.111619
7	Driver Experience	0.060400
6	Vehicle Age	0.054350
5	Weather Condition	-0.002165
8	Road Quality	-0.021218
12	Fuel Efficiency	-0.069463
11	Temperature	-0.082980
1	Number of Deliveries	-0.130603

We initialized a new Linear Regression model specifically for this hypothesis and trained it using only the selected correlated features. This model had fewer coefficients to calculate compared to Hypothesis 1, making it simpler and potentially more interpretable. The training process followed the same approach as before, minimizing the difference between predicted and actual delivery times across the training data.

5.2.4 Model Training Process

With a reduced feature set, the linear regression algorithm created a simpler equation with fewer terms. In our case, only 5 features were selected instead of all 13.

Time Taken = (coefficient$_1$ × Feature$_1$) + (coefficient$_2$ × Feature$_2$) + ... for only 5 significant features, plus the intercept.

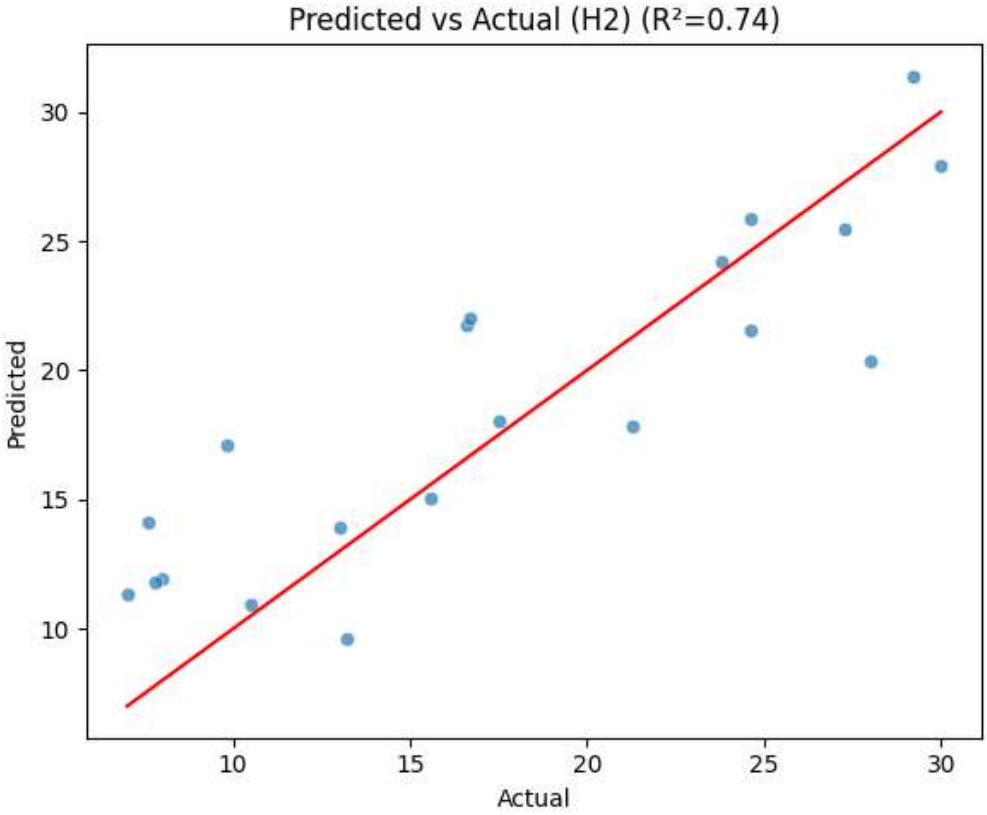
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression

# 'Time Taken' is the target variable and the rest are selected features
X = data2.drop('Time Taken', axis=1)
y = data2['Time Taken']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize and train the Linear Regression model
model = LinearRegression()
model.fit(X_train, y_train)

print("Model training for hypothesis 2 complete.")
```

The advantage of this approach is that each retained feature should have a clear, meaningful relationship with delivery time. The model doesn't waste computational effort trying to extract weak signals from irrelevant features. This focused approach should theoretically lead to more reliable and stable predictions.

5.2.5 Prediction and Results

After training, we applied the model to the same test set used in Hypothesis 1, ensuring a fair comparison. The model generated predictions based only on the selected correlated features, ignoring the variables that were deemed weakly related to delivery time.

Hypothesis 2 Performance Metrics	
MAE	3.22
MSE	15.50
RMSE	3.93
R^2	0.74



The model achieved an R^2 score of 0.74, meaning it explained 74% of the variation in delivery times. While this represents strong predictive performance, it was actually slightly lower than Hypothesis 1's score of 0.76. The predicted vs actual scatter plot showed similar clustering around the diagonal reference line, with comparable levels of scatter indicating similar prediction accuracy.

5.3 Hypothesis 3: Non-Multicollinearity Features Model

5.3.1 Hypothesis Statement

The third hypothesis proposes that excluding features that exhibit multicollinearity from the linear regression model will yield better performance compared to both the all-features model and the correlated features model. This approach addresses a specific statistical issue where independent variables are highly correlated with each other, which can distort model coefficients and reduce prediction reliability.

Multicollinearity occurs when two or more independent variables in a regression model are highly correlated with each other. For example, in our dataset, Miles Travelled and Delivery Distance are nearly identical measurements that move together almost perfectly. When such highly correlated features are included together, the model struggles to distinguish their individual effects on the target variable.

This creates several problems: coefficient estimates become unstable and unreliable, small changes in the data can lead to large changes in coefficients, and it becomes difficult to interpret which features truly matter. By removing multicollinearity features, we create a cleaner model where each remaining feature provides unique, independent information. This should theoretically improve both model stability and prediction accuracy.

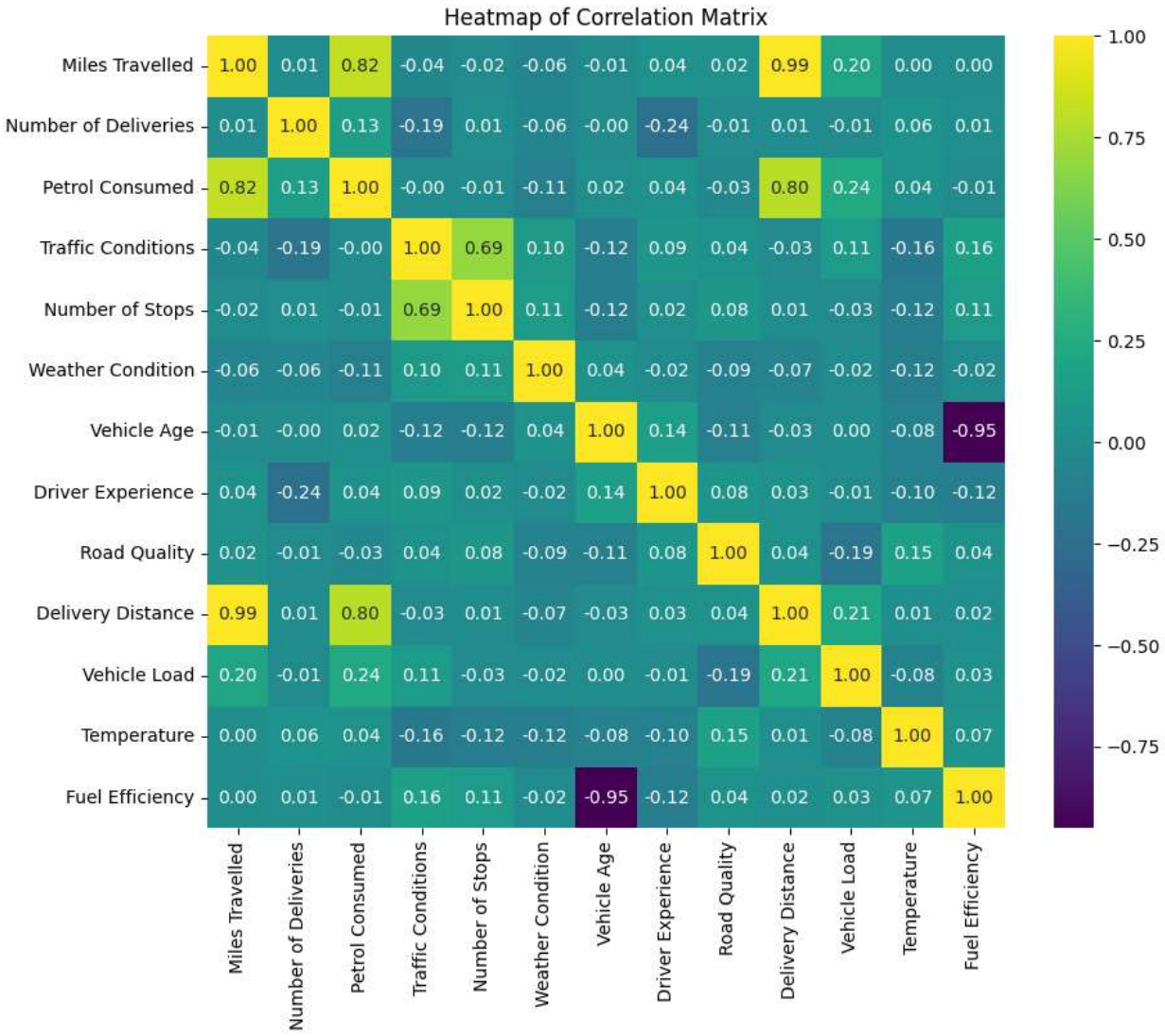
The hypothesis suggests that the redundancy created by multicollinearity features adds noise rather than value, and that a model using only independent, non-redundant features will perform better than both the all-features approach and the correlation-based selection approach.

5.3.2 Multicollinearity Detection via Correlation Matrix

To detect multicollinearity, we first examined the correlation matrix, which shows the correlation coefficients between all pairs of independent variables. This matrix helps identify which features are highly correlated with each other (not with the target variable, as in Hypothesis 2).

The correlation matrix revealed several instances of high inter-feature correlation. Most notably, Petrol Consumed, Delivery Distance, and Fuel Efficiency showed correlation coefficients greater than 0.8, confirming they measure essentially the same thing. We used a **0.8 threshold to flag potential multicollinearity, following common practice in regression diagnostics (Senaviratna & Cooray, 2019)**.

The correlation matrix provided a visual and numerical way to identify redundant features that are good candidates for removal. The complete correlation matrix can be viewed inside the notebook (linked on page 4). However, to visualize the correlation matrix, we generated a heatmap, pasted below.



5.3.3 Model A: Correlation-Based Multicollinearity Removal

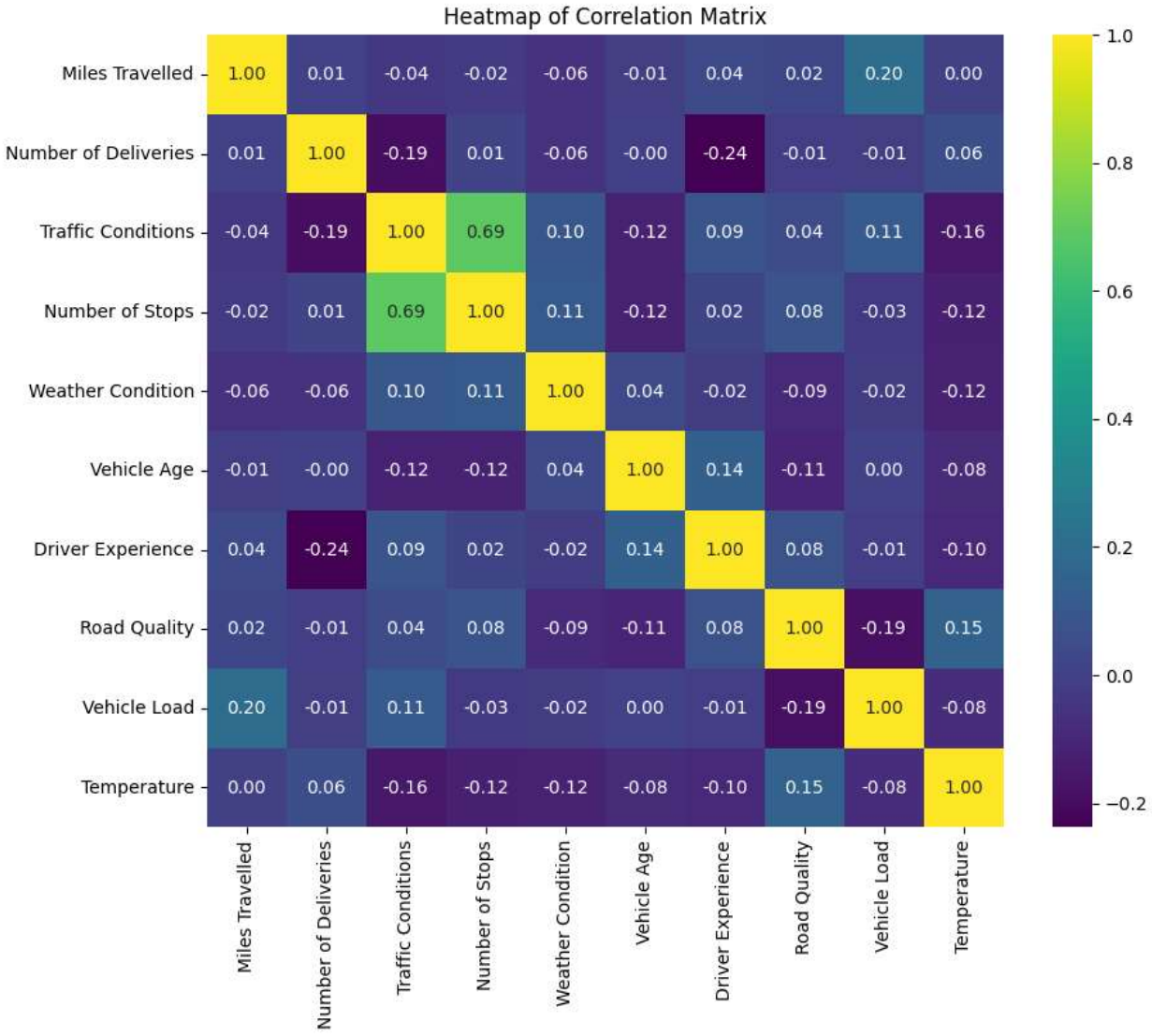
Model Implementation

For the first approach to Hypothesis 3, we used the correlation matrix to systematically remove highly correlated features. When two features showed high correlation with each other, we examined which one had a stronger correlation with the target variable (Time Taken) and retained that feature while removing the other.

For example, if Miles Travelled and Delivery Distance were highly correlated with each other, we would keep whichever one had a stronger relationship with Time Taken and exclude the other. This process was repeated for all pairs of highly correlated features, progressively building a feature set where no two features were highly correlated with each other. The following variables were dropped from the dataset because of high multicollinearity.

Dropped Features		
Petrol Consumed	Delivery Distance	Fuel Efficiency

The heatmap for only the remaining features after dropping multicollinear features is shown below.



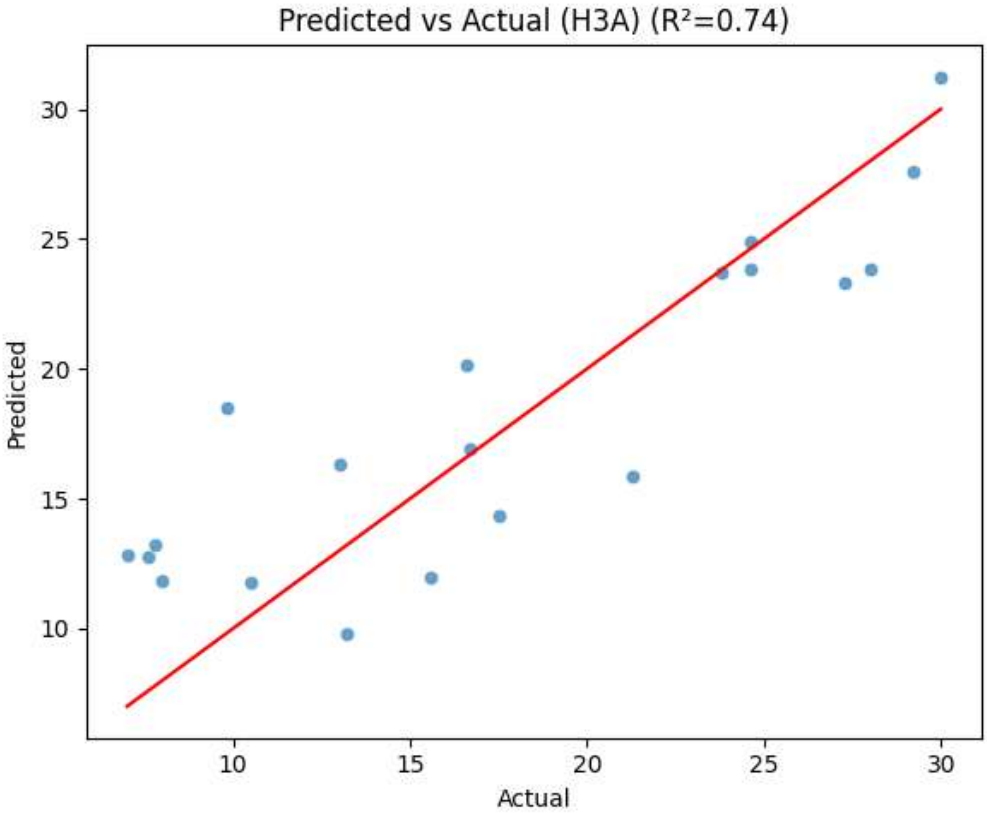
Model Training and Results

After removing redundant features based on the correlation matrix, we created a new feature set containing only the non-multicollinear variables. This reduced dataset was then split into training and testing sets using the same 80-20 ratio, and a Linear Regression model was trained on this refined feature set.

The model trained on correlation-based non-multicollinearity features created a regression equation with 9 independent variables. Each coefficient represented the unique contribution of a feature without interference from highly correlated counterparts. This should theoretically make the model more interpretable and stable.

After training and testing, this model achieved an R^2 score of 0.74 (explained 74% of the variation in delivery times).

Hypothesis 3A Performance Metrics	
MAE	3.25
MSE	15.4
RMSE	3.92
R^2	0.74



5.3.4 Multicollinearity Detection via Variance Inflation Factor (VIF)

VIF Methodology

While correlation matrices are useful, they only examine pairs of variables at a time. Multicollinearity can also occur when a feature is predicted by a combination of several other features, which pairwise correlation might miss. The Variance Inflation Factor (VIF) is a more comprehensive measure that detects all forms of multicollinearity.

Variance Inflation Factor (VIF)	
constant	722.168364
Miles Travelled	41.710693
Number of Deliveries	1.230650
Petrol Consumed	3.489333
Traffic Conditions	2.325464
Number of Stops	2.168150
Weather Condition	1.055842
Vehicle Age	11.599851
Driver Experience	1.116665
Road Quality	1.153415
Delivery Distance	37.845556
Vehicle Load	1.170702
Temperature	1.092595
Fuel Efficiency	11.494265

VIF quantifies the extent to which the variance of a coefficient estimate is inflated due to multicollinearity. A VIF value of 1 indicates no correlation with other features, while higher values indicate increasing multicollinearity. Generally, VIF values above 5 (threshold in our case) or 10 are considered problematic and suggest that the feature should be removed.

We calculated VIF for each feature in our dataset. Features showing high VIF values were identified as problematic contributors to multicollinearity.

5.3.5 Model B: VIF-Based Multicollinearity Removal

Model Implementation

We systematically removed 4 features exceeding this value. The process was iterative: after removing the feature with the highest VIF, we recalculated VIF values for all remaining features, as removing one multicollinearity feature affects the VIF of others.

This iterative process continued until all remaining features had acceptable VIF values below the threshold. The result was a feature set where each variable provided independent, unique information without being predictable from combinations of other features.

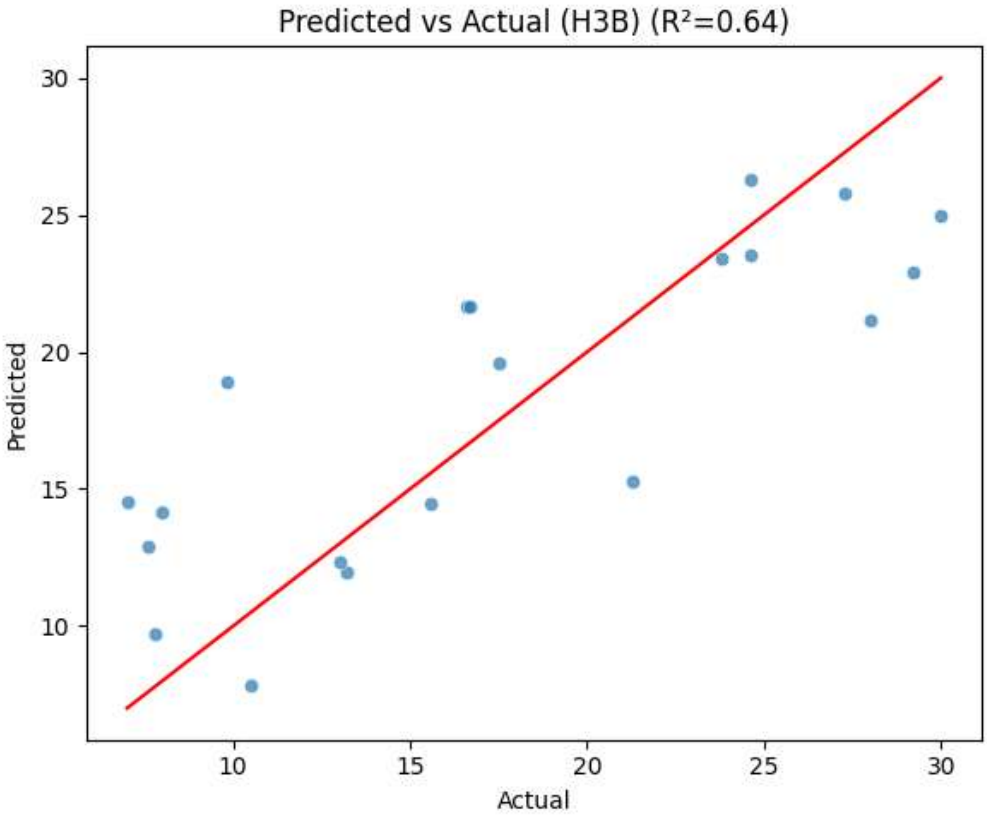
Dropped Features			
Miles Travelled	Delivery Distance	Vehicle Age	Fuel Efficiency

Model Training and Results

After identifying the final set of 9 low-VIF features, we created our feature matrix (100 × 9) and split it into training and testing sets. A new Linear Regression model was trained specifically on this VIF-filtered dataset, with the expectation that removing all forms of multicollinearity would produce the most stable and accurate model.

After training on the VIF-filtered features and testing on data, this model generated predictions and achieved a 0.64 R^2 score.

Hypothesis 3B Performance Metrics	
MAE	3.83
MSE	21.43
RMSE	4.62
R^2	0.64


Predicted vs Actual (H3B) (R²=0.64)
Predicted
Actual
30
25
20
15
10
10
15
20
25
30

Comparison and Interpretation of Models

The following table represents the performance metrics of all four models.

	Hypothesis 01	Hypothesis 02	Hypothesis 03	
			Model A	Model B
MAE	3.03	3.23	3.25	3.83
MSE	14.47	15.50	15.40	21.43
RMSE	3.80	3.94	3.92	4.62
R^2	0.76	0.74	0.74	0.64

While Hypothesis 1 (All Features Model) performs best with an R^2 of 0.76 (76%), it requires all 13 features to achieve this accuracy. In contrast, Hypothesis 2 (Correlated Features Model) achieves nearly comparable performance with R^2 of 0.74 (74%) using only 5 features - Number of Stops, Traffic Conditions, Delivery Distance, Miles Travelled, and Petrol Consumed. This represents an excellent trade-off where H2 sacrifices just 2% accuracy while reducing model complexity by more than half, making it significantly more efficient and practical for real-world implementation with an average prediction error of only 3.2 minutes.

Hypothesis 3A (Correlation-based Multicollinearity Removal) achieved R^2 of 0.74 (74%) with 10 features, performing the same as H2 (by 0.2%) while using twice as many features. This indicates that removing the three multicollinear features (Petrol Consumed, Delivery Distance, Fuel Efficiency) through correlation analysis maintained strong predictive power while addressing redundancy issues, though it did not outperform the simpler H2 approach in terms of efficiency.

Hypothesis 3B (VIF-based Multicollinearity Removal) showed a significant performance drop with R^2 of 0.64 (64%) despite using 9 features. This 10% decrease in explanatory power compared to H2 and the substantially higher prediction errors (MAE = 3.833 minutes, RMSE = 4.629 minutes) indicate that the four features removed during VIF analysis (Miles Travelled, Delivery Distance, Vehicle Age, and Fuel Efficiency) contained crucial predictive information. The notable performance degradation suggests that at least one of these dropped variables has a prominent impact on prediction accuracy and warrants further investigation. Specifically, Miles Travelled and Delivery Distance, while multicollinear with each other, appear to be essential predictors that cannot be removed without significantly compromising model performance.

The results demonstrate that the feature selection strategy matters significantly. **H2 emerges as the most efficient model, achieving 74.1% accuracy with minimal features, while H1 provides the highest accuracy at the cost of complexity.** H3B's poor performance highlights the risk of overly aggressive multicollinearity removal; eliminating redundant features can inadvertently remove critical predictive information, emphasizing the need for careful balance between statistical independence and predictive power.

References

1. AlOmari, H., Yaseen, Q., & Al-Betar, M. A. (2023). A Comparative Analysis of Machine Learning Algorithms for Android Malware Detection. *Procedia Computer Science*. (Research Gate)
2. Senaviratna, N., & Cooray, T. M. (2019). Diagnosing Multicollinearity of Logistic Regression Model. *Asian Journal of Probability and Statistics*. (Research Gate)
3. Menard, S. (2002). *Applied logistic regression analysis* (2nd ed.). Sage University Paper. (ResearchGate)
4. Azar, Y. (2017). Some new methods to solve multicollinearity in logistic regression. *Communications in Statistics—Theory and Methods, 46*(4), 2576–2586. (ResearchGate)
5. Schaefer, R. L., Roi, L. D., & Wolfe, R. A. (1984). A ridge logistic estimator. *Communications in Statistics, 13*(1), 99–113. (ResearchGate)