Predictive Modelling for Auto Insurance Premium Estimation

Mehak Sharma

Contents

1 Introduction

In today's world, making smart decisions often means using data effectively. This is especially true for industries that handle a lot of information and need to understand risks, like insurance. Insurance companies constantly aim to set prices accurately, balancing what customers can afford with what the company needs to earn. This is where advanced data analysis, particularly using machine learning, comes in. It provides powerful ways to find hidden connections in large datasets, helping businesses make fairer and more informed choices.

This report details a machine learning project where we used historical data to improve how we estimate car insurance premiums. We'll walk through our approach step-by-step: first, exploring and getting the data ready, then building and comparing different prediction models. You'll see how well these models perform and what we learned about the key factors that influence auto insurance prices. To begin, let's look at the specific challenge we aimed to solve.

1.1 Problem Statement

In the insurance industry, pricing accuracy is crucial for balancing customer affordability with insurer profitability. Traditional actuarial methods may not capture complex relationships among customer, vehicle, and location factors that influence auto insurance premiums. This project aims to develop a predictive model using machine learning in R to estimate auto insurance premiums based on features such as driver demographics, vehicle characteristics, driving history, and regional risk factors. The model will be trained on real-world insurance data and evaluated to determine its predictive power and reliability. The objective is to identify significant variables, reduce pricing uncertainty, and provide a data-driven approach that could assist insurers in fair and accurate premium calculation.

2 Data Description and Initial Exploration

The project utilized a dataset (referred to as 'motor_data') comprising various attributes related to motor vehicles and insurance policies. The dataset initially contained approximately 407,000 observations and numerous variables, including details such as vehicle type, make, age, cubic capacity, insured value, and the target variable, 'PREMIUM'.

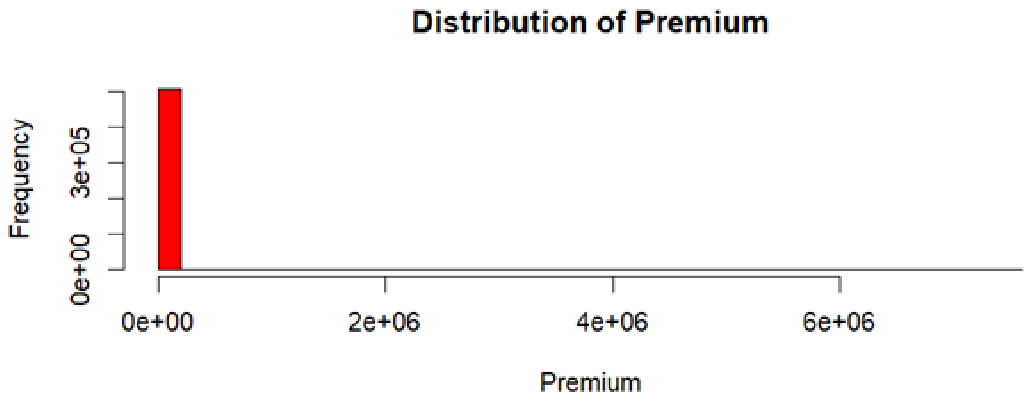
2.1 Initial Data Overview

Upon initial inspection, the dataset provided a rich set of features for predicting insurance premiums. Data types included numerical (e.g., 'INSURED_VALUE', 'CCM_TON', 'VEHICLE_AGE') and categorical (e.g., 'TYPE_VEHICLE', 'MAKE', 'USAGE', 'SEX').

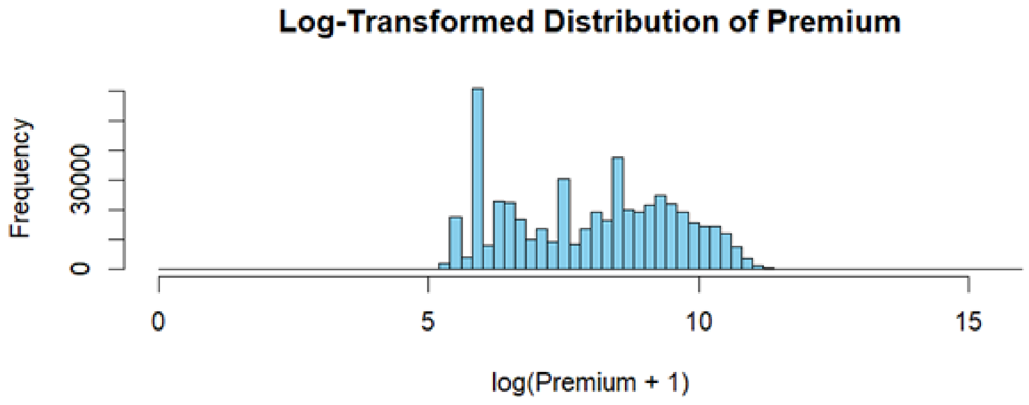
2.2 Target Variable (‘PREMIUM‘)

A critical observation from the initial exploration was the distribution of the target variable, ‘PREMIUM‘. As shown in Figure 1, the ‘PREMIUM‘ distribution was highly skewed to the right, with a large concentration of values at the lower end and a long tail extending towards higher values. This non-normal, skewed distribution is common in real-world financial data and poses challenges for many linear regression models that assume normally distributed residuals.

To address this, a logarithmic transformation was applied. Specifically, the ‘log(PREMIUM + 1)‘ transformation was used to handle cases where ‘PREMIUM‘ could be zero and to effectively normalize its distribution. As illustrated in Figure 1, this transformation successfully rendered the ‘log_PREMIUM‘ distribution much more symmetrical and bell-shaped, making it suitable for models that perform better with less skewed target variables.



(a) Initial Distribution of Premium (Highly Skewed)



(b) Distribution of Log-Transformed Premium

Figure 1: Comparison of Premium Distributions: Original vs. Log-Transformed

3 Data Preprocessing and Feature Engineering

This phase was critical for transforming the raw data into a format suitable for machine learning models, addressing issues such as missing values, skewed distributions, and appropriate representation of categorical variables.

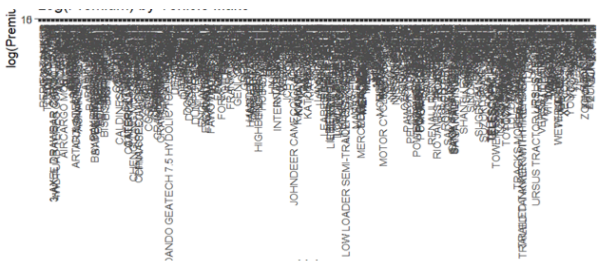
3.1 Missing Value Management

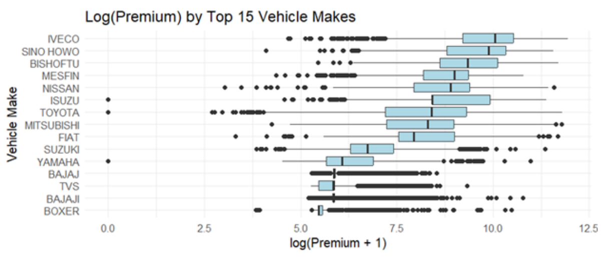
Missing values (‘NA‘s) were identified across some variables in the dataset. For this analysis, a straightforward approach was adopted where rows containing any missing values were removed using ‘na.omit()‘. This resulted in a clean dataset without missing entries. This strategy was chosen due to the relatively small proportion of missing data, ensuring minimal loss of valuable observations while simplifying the modeling process.

3.2 Categorical Feature Handling

The dataset contained several categorical features, most notably ‘MAKE‘, ‘TYPE_VEHICLE‘, and ‘USAGE‘. These variables needed to be converted into a numerical format for most machine learning algorithms.

- **Grouping ‘make‘:** The ‘MAKE‘ variable presented a challenge due to its very high cardinality (a large number of unique vehicle manufacturers), as shown in Figure 2a. To manage this, less frequent ‘make‘ categories were grouped into a single ’OTHERS’ category. This dimensionality reduction helped prevent sparsity and overfitting while retaining the predictive power of major manufacturers. The effect of this grouping is evident in Figure 2b, which shows a clearer distribution for the top manufacturers.


(a) Premium by Original Vehicle Make



(b) Premium by Grouped Vehicle Make

Figure 2: Comparison of Premium Distribution by Vehicle Make (Original vs. Grouped)

- **Dummy Variable Encoding:** All categorical variables (including the grouped ’MAKE’ and others like ’TYPE_VEHICLE’, ’USAGE’, ’SEX’, ’INSR_TYPE’) were converted into numerical dummy (binary 0/1) variables. This process expanded the number of predictors significantly, from the original raw features to 103 predictors in the final prepared dataset (’model_finaldata’).

3.3 Detailed EDA Visualizations

Beyond the target variable transformation, extensive exploratory data analysis was conducted to understand the relationships between various features and the transformed premium, as well as inter-feature dynamics.

3.3.1 Distribution of Log-Transformed Premium (Boxplot)

A boxplot of the log-transformed premium, as shown in Figure 3, provides a clear visual summary of its central tendency, spread, and presence of outliers. This plot confirms the reduced skewness and helps identify the interquartile range and median, demonstrating the effectiveness of the logarithmic transformation in achieving a more symmetric distribution.

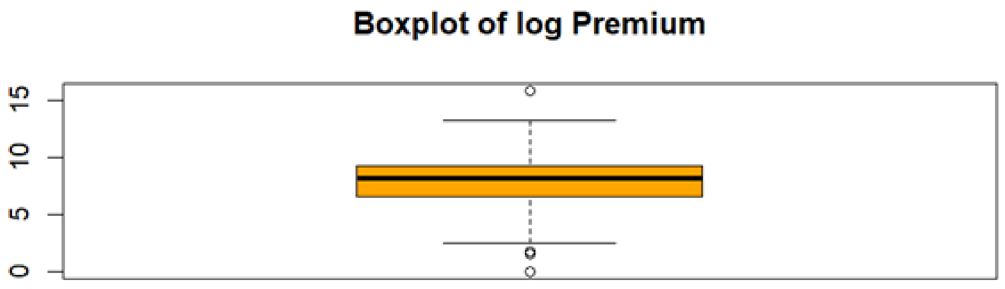


Figure 3: Boxplot of Log-Transformed Premium

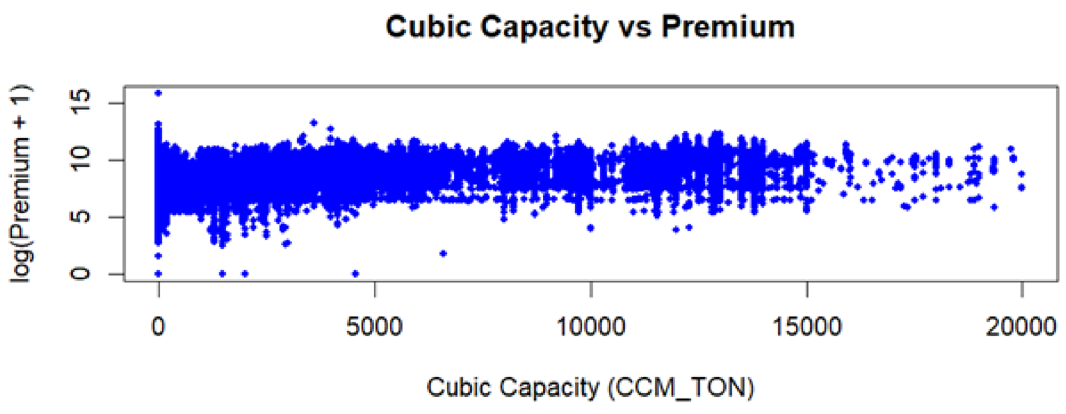
3.3.2 Relationship Between Log-Premium and Numerical Variables

Further investigation into numerical predictors revealed several key relationships with the log-transformed premium.

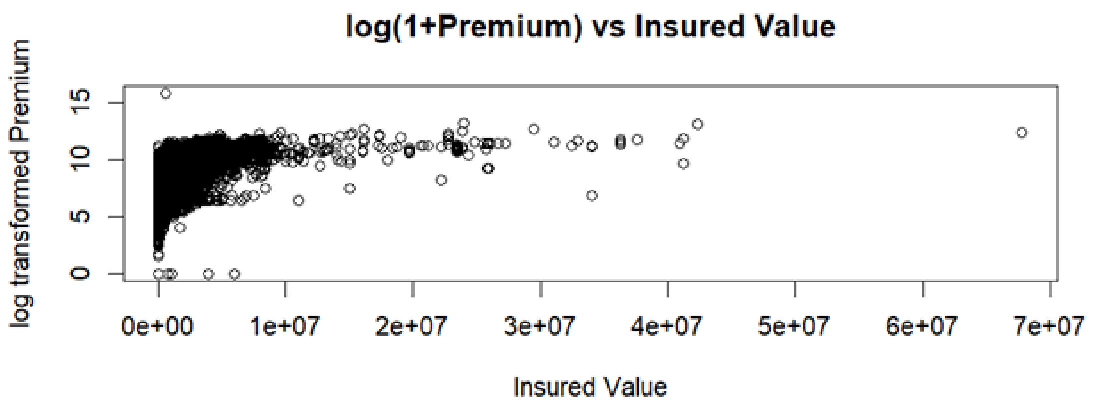
- **Cubic Capacity (`CCM_TON`) vs. Log-Premium:** The scatter plot in Figure 4a illustrates the relationship between vehicle cubic capacity (or tonnage) and log-premium. Generally, vehicles with higher `CCM_TON` tend to be associated with higher premiums, reflecting increased risk or value.

- **Insured Value vs. Log-Premium:** As intuitively expected, Figure 4b clearly shows a strong positive correlation between the `INSURED_VALUE` and `log_PREMIUM`. Higher insured values directly translate to higher potential payouts, thus commanding higher premiums. This was identified as a paramount feature.

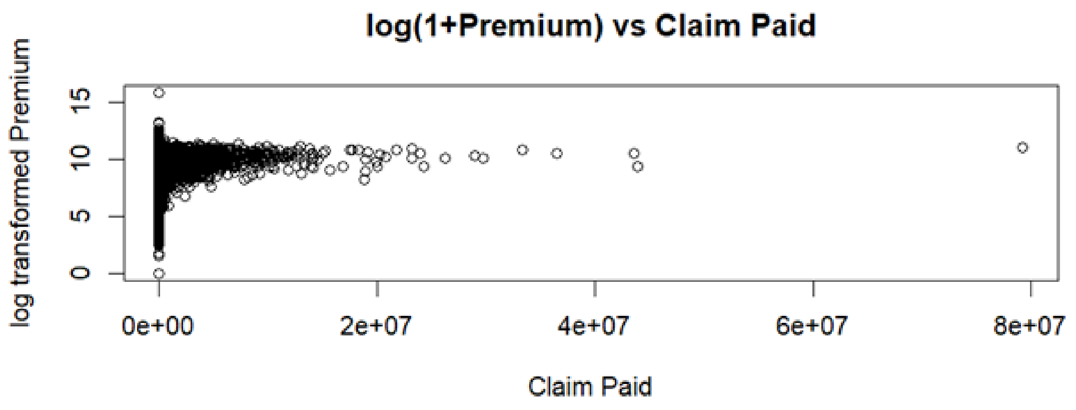
- **Claim Paid vs. Log-Premium:** The relationship between historical `CLAIM_PAID` and `log_PREMIUM`, displayed in Figure 4c, provides insights into how past claim experience might correlate with current premium levels. While premiums are forward-looking, past claims can indicate risk propensity.



(a) Log-Premium vs. Cubic Capacity (`CCM_TON`)



(b) Log-Premium vs. Insured Value



(c) Log-Premium vs. Claim Paid

Figure 4: Relationships Between Log-Premium and Key Numerical Variables

3.3.3 Correlation Plot

The correlation matrix, presented in Figure 5, provides a holistic view of linear relationships among all numerical predictors and the log-transformed target. This helps in identifying potential multicollinearity issues and highlights features strongly correlated with 'log_PREMIUM'.

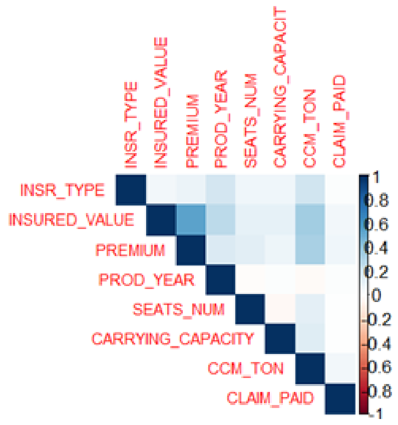


Figure 5: Correlation Matrix of Numerical Predictors

3.3.4 Impact of Categorical Variables on Premium

Categorical variables play a significant role in differentiating premium levels based on distinct groups.

- **Log-Premium by Vehicle Type:** Figure 6 displays boxplots of log-premium across different `TYPE_VEHICLE` categories. Certain vehicle types (e.g., Motorcycles, Trucks) show notably different premium distributions, reflecting varying risk profiles and repair costs associated with them.
- **Log-Premium by Sex:** Figure 7 illustrates the distribution of log-premium based on the `SEX` of the policyholder. While differences may be subtle, such demographic factors are often considered in risk assessment.

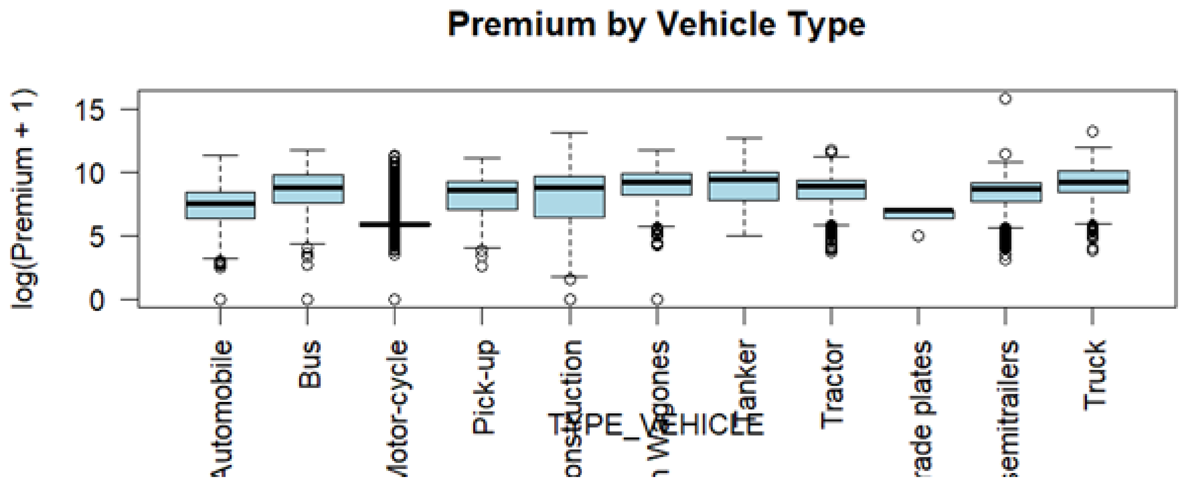


Figure 6: Log-Premium by Vehicle Type

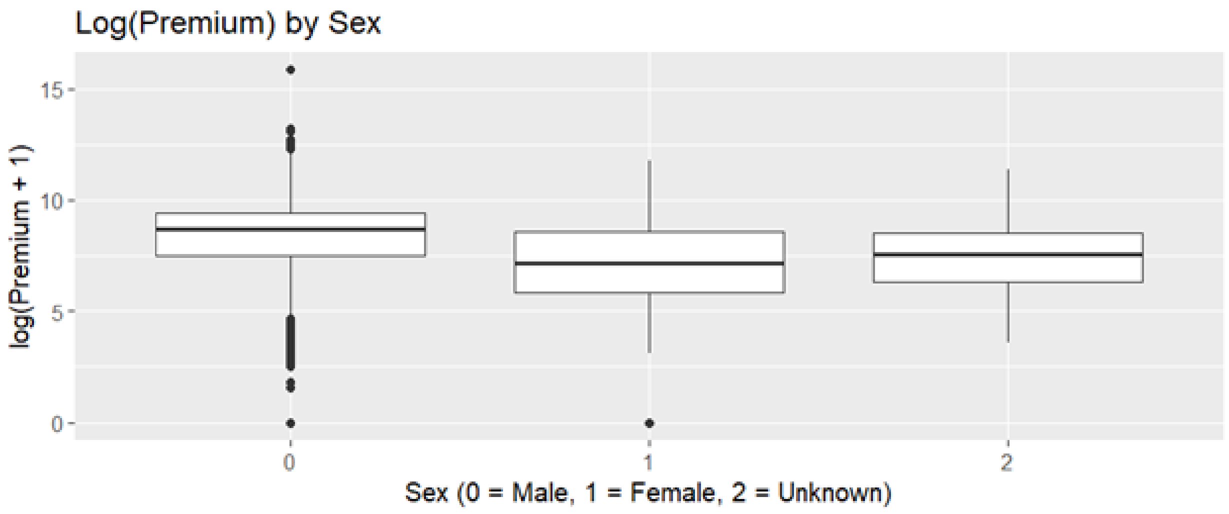


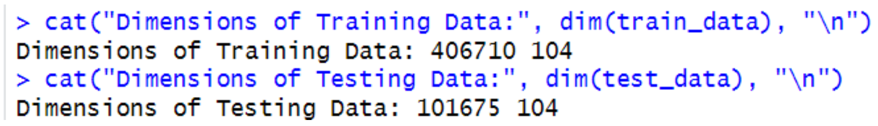
Figure 7: Log-Premium by Sex

4 Model Preparation

Before actual model training, the final preprocessed dataset ('model_finaldata'), containing 406,710 observations and 103 predictors, was meticulously prepared for robust model development and unbiased evaluation.

4.1 Data Splitting

To ensure the model's ability to generalize to unseen data, the 'model_finaldata' was split into training and testing sets. An 80% portion was allocated for training ('train_data'), and the remaining 20% was reserved as a test set ('test_data'). The 'createDataPartition' function from the 'caret' package was used to maintain the proportion of the target variable's distribution in both sets, enhancing the representativeness of the split. A fixed random seed ('set.seed(123)') was used to ensure the reproducibility of this split.



```
> cat("Dimensions of Training Data:", dim(train_data), "\n")
Dimensions of Training Data: 406710 104
> cat("Dimensions of Testing Data:", dim(test_data), "\n")
Dimensions of Testing Data: 101675 104
```

Figure 8: dimensions of data

4.2 Training Control (Cross-Validation)

The 'trainControl' function from the 'caret' package was configured to manage the model training process, particularly for hyperparameter tuning and robust performance estimation.

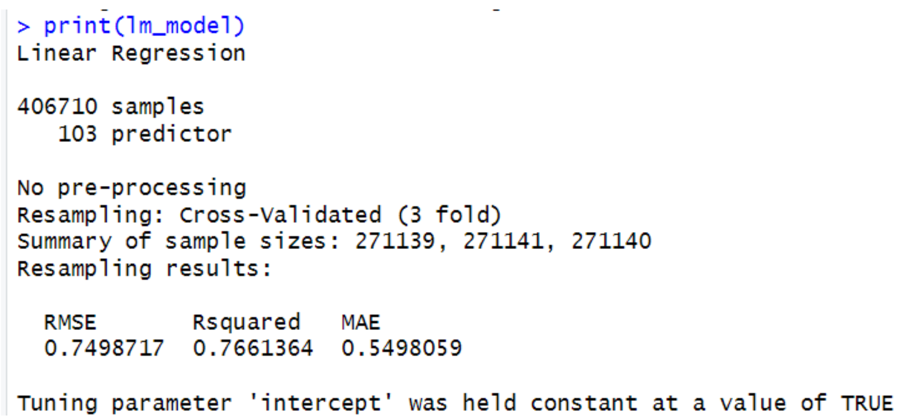
- **Cross-Validation Method:** A 3-fold cross-validation ('method = "cv", number = 3') was chosen. This means the training data was repeatedly split into three equal parts: two parts for training and one part for validation, cycling through all combinations. This process provides a more reliable estimate of model performance than a single train-validation split and helps in identifying optimal hyperparameters.
- **Parallel Processing:** Initially, parallel processing was attempted to speed up training. However, due to the very large dataset size and potential memory constraints or stability issues with many cores, the training was ultimately performed sequentially ('num_cores = 1' or effectively no 'registerDoParallel' cluster active during 'train()' calls) to ensure stability and completion on the available system resources.

5 Model Development and Training

Three distinct regression models were developed and trained using the prepared ‘train_data‘ to predict the ‘log_PREMIUM‘.

5.1 Linear Regression (LM)

Linear Regression served as a foundational baseline model. It models the relationship between the predictors and the target variable as a linear equation. While simpler, it provides a crucial reference point against which the performance of more complex non-linear models can be compared. The model was trained using the ’lm’ method in ’caret’.



```
> print(lm_model)
Linear Regression

406710 samples
   103 predictor

No pre-processing
Resampling: Cross-Validated (3 fold)
Summary of sample sizes: 271139, 271141, 271140
Resampling results:

  RMSE       Rsquared   MAE
  0.7498717  0.7661364  0.5498059

Tuning parameter 'intercept' was held constant at a value of TRUE
```

Figure 9: linear model

5.2 Random Forest (RF)

Random Forest is an ensemble learning method that builds multiple decision trees during training and outputs the mean prediction of the individual trees (for regression). It leverages the power of "bagging" and random feature selection at each split to reduce overfitting and enhance robustness. The ’ranger‘’implementation was used via ’caret’.

- **Key Hyperparameters Tuned:**
  - ’mtry’: The number of randomly selected predictors considered at each split. Values tested were 10, 25, and 40.
  - ’num.trees’: The number of trees in the forest, set to 250.
  - Other parameters like ’splitrule’ and ’min.node.size’ were set to default values suitable for regression.
- **Tuning Strategy:** A predefined ‘tuneGrid’ was used to explicitly test specific ‘mtry’ values, which allowed for targeted exploration of this critical parameter. Figure 10 illustrates the tuning process, showing the RMSE performance across different ‘mtry’ values. The lowest cross-validated RMSE was achieved at ‘mtry=40‘.

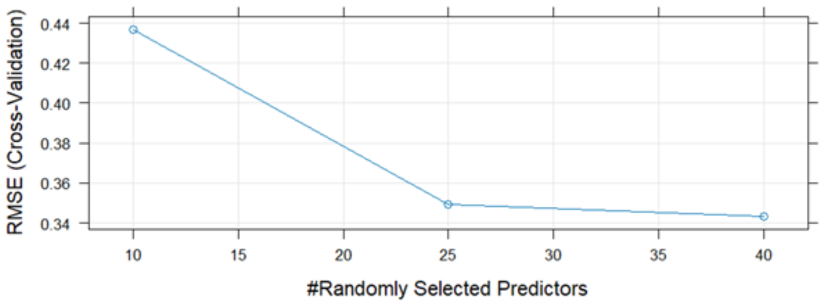
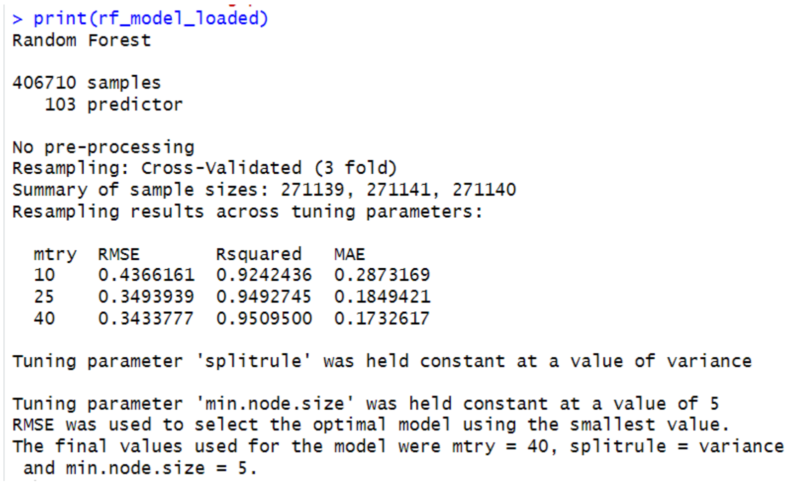


Figure 10: Random Forest Tuning: RMSE vs. Number of Randomly Selected Predictors (mtry)



```
> print(rf_model_loaded)
Random Forest

406710 samples
   103 predictor

No pre-processing
Resampling: Cross-Validated (3 fold)
Summary of sample sizes: 271139, 271141, 271140
Resampling results across tuning parameters:

  mtry  RMSE       Rsquared   MAE
  10    0.4366161  0.9242436  0.2873169
  25    0.3493939  0.9492745  0.1849421
  40    0.3433777  0.9509500  0.1732617

Tuning parameter 'splitrule' was held constant at a value of variance

Tuning parameter 'min.node.size' was held constant at a value of 5
RMSE was used to select the optimal model using the smallest value.
The final values used for the model were mtry = 40, splitrule = variance
 and min.node.size = 5.
```

Figure 11: Random Forest

5.3 XGBoost (eXtreme Gradient Boosting)

XGBoost is a highly optimized and efficient open-source implementation of the gradient boosting framework. It builds trees sequentially, with each new tree trying to correct the errors of the previous ones. It is known for its speed and strong predictive performance. The 'xgbTree' method was used via 'caret'.

- **Key Hyperparameters Tuned:**
  - 'nrounds': The number of boosting iterations (trees). Values tested were 100 and 200.
  - 'max_depth': The maximum depth of a tree. Values tested were 3 and 5.
  - 'eta': The learning rate, fixed at 0.1.
  - Other parameters like 'gamma', 'colsample_bytree', 'min_child_weight', and 'subsample' were set.

- **Tuning Strategy:** The 'tuneGrid' explored combinations of 'nrounds' and 'max_depth'. Figure 12 displays the cross-validation RMSE across these combinations. The optimal parameters were found to be 'nrounds = 200' and 'max_depth = 5', yielding the best cross-validated performance.

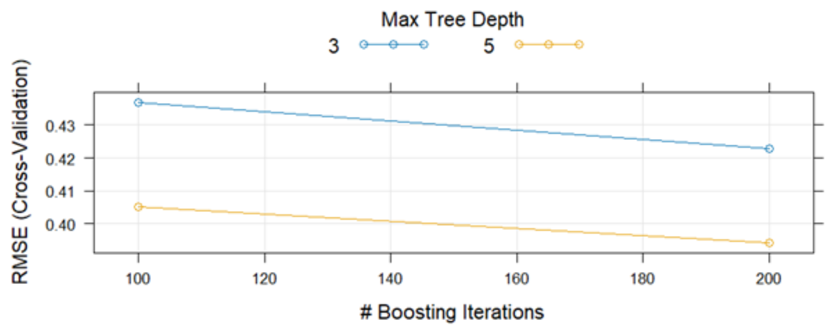
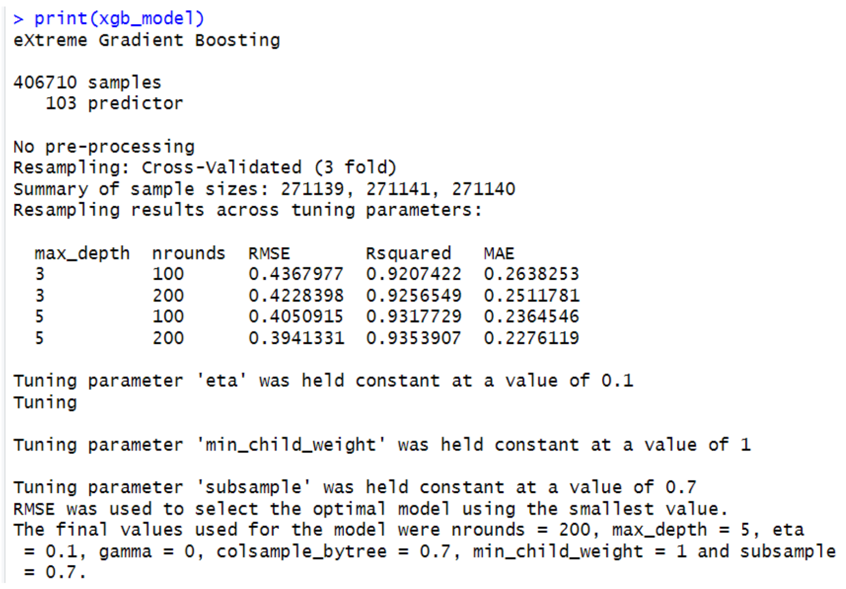


Figure 12: XGBoost Tuning: RMSE vs. Boosting Iterations for Different Max Depths



```
> print(xgb_model)
eXtreme Gradient Boosting

406710 samples
   103 predictor

No pre-processing
Resampling: Cross-Validated (3 fold)
Summary of sample sizes: 271139, 271141, 271140
Resampling results across tuning parameters:

  max_depth  nrounds  RMSE       Rsquared   MAE
  3          100      0.4367977  0.9207422  0.2638253
  3          200      0.4228398  0.9256549  0.2511781
  5          100      0.4050915  0.9317729  0.2364546
  5          200      0.3941331  0.9353907  0.2276119

Tuning parameter 'eta' was held constant at a value of 0.1
Tuning

Tuning parameter 'min_child_weight' was held constant at a value of 1

Tuning parameter 'subsample' was held constant at a value of 0.7
RMSE was used to select the optimal model using the smallest value.
The final values used for the model were nrounds = 200, max_depth = 5, eta
 = 0.1, gamma = 0, colsample_bytree = 0.7, min_child_weight = 1 and subsample
 = 0.7.
```

Figure 13: XGBoost

6 Model Evaluation and Selection

To obtain an unbiased assessment of each model's generalization capability, all trained models were evaluated on the previously held-out 'test_data', which they had not encountered during training or tuning.

6.1 Performance Metrics

The following standard regression metrics were used for evaluation:

- **RMSE (Root Mean Squared Error):** Measures the average magnitude of the errors. It is sensitive to large errors. Lower RMSE indicates better performance.
- **MAE (Mean Absolute Error):** Represents the average absolute difference between predicted and actual values. It is less sensitive to outliers than RMSE. Lower MAE indicates better performance.
- **R-squared (R^2):** Indicates the proportion of the variance in the dependent variable that is predictable from the independent variables. A higher R-squared value (closer to 1) indicates a better fit.

6.2 Test Set Performance Comparison

The performance of all three models on the test set is summarized in Table 1.

Table 1: Model Performance Comparison on Test Set

Model	**RMSE**	**MAE**	**R-squared**
Linear Regression	0.73211	0.54788	0.77720
Random Forest	0.33039	0.16644	0.95462
XGBoost	0.39025	0.22765	0.93669

6.3 Model Selection

Based on the comprehensive evaluation on the unseen test set, the **Random Forest model ('rf_model') emerged as the superior performer**. It achieved the lowest RMSE (0.33039), the lowest MAE (0.16644), and the highest R-squared (0.95462) among all tested models. This indicates that the Random Forest model provides the most accurate and reliable predictions for 'log_PREMIUM' in this context. While XGBoost also performed very well, Random Forest demonstrated a slight edge in all key metrics. Linear Regression, as expected, served as a weaker baseline.

6.4 Visualizing Predictions

To further assess the Random Forest model's performance visually, plots comparing actual vs. predicted values and residuals vs. predicted values were generated.

Figure 14 shows the actual 'log_PREMIUM' values plotted against the predicted 'log_PREMIUM' values on the test set. The strong clustering of data points along the 45-degree dashed red line clearly illustrates the high accuracy of the Random Forest model's predictions.

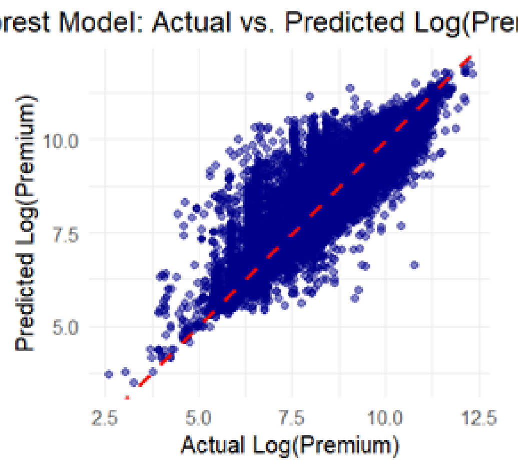


Figure 14: Random Forest Model: Actual vs. Predicted Log(Premium) (Test Set)

Figure 15 displays the residuals (actual minus predicted values) against the predicted 'log_PREMIUM'. The random scattering of points around the zero-line, without any discernible patterns (e.g., U-shape, funnel shape), indicates that the model's errors are randomly distributed and do not exhibit systematic biases. This further reinforces the robustness of the Random Forest model.

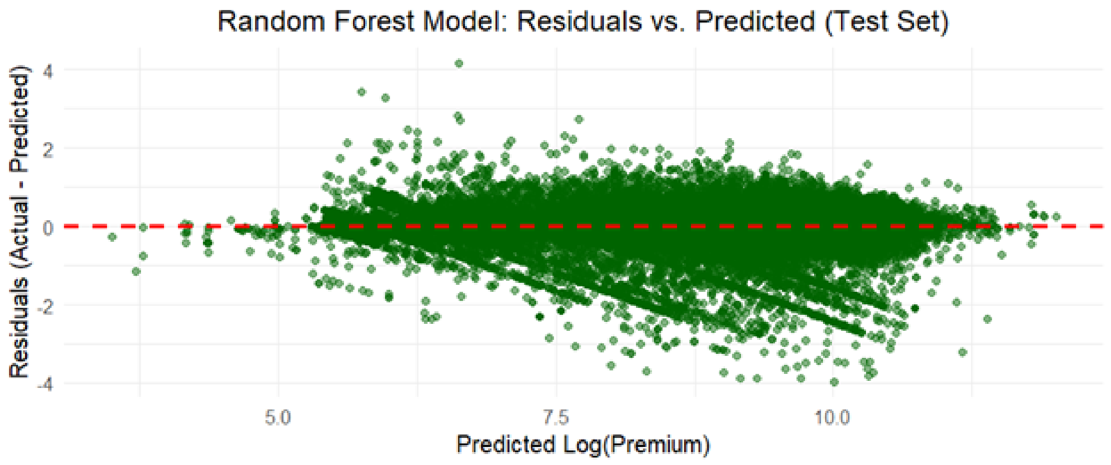


Figure 15: Random Forest Model: Residuals vs. Predicted (Test Set)

7 Importance Analysis

Understanding which factors contribute most to the predictions is crucial for actionable insights. Variable importance analysis was performed on the selected Random Forest model. This technique quantifies the contribution of each predictor to the model's predictive accuracy, typically by measuring the reduction in error when that variable is used in tree splits.

Figure 16 displays the top 20 most important variables identified by the Random Forest model. The most impactful predictors are:

- **INSURED_VALUE:** Consistently ranked as the most significant predictor by a substantial margin. This is highly intuitive, as the value of the asset being insured directly correlates with potential claim payouts.
- **TYPE_VEHICLE.Motor.cycle:** The type of vehicle, specifically motorcycles, emerged as the second most important factor. Different vehicle types carry different risk profiles, and motorcycles often have distinct accident rates and repair costs.
- **CCM_TON:** This likely refers to cubic capacity or tonnage, representing the size or power of the vehicle. This variable also showed high importance, reflecting that larger or more powerful vehicles might be associated with higher premiums.
- **VEHICLE_AGE:** The age of the vehicle is another key predictor, influencing factors like depreciation, safety features, and likelihood of mechanical issues.
- **MAKE_GROUPED.BAJAJI:** Specific vehicle manufacturers, even after grouping, retained high importance, indicating that certain brands have unique risk characteristics influencing premium costs.

These findings align well with common understanding of factors influencing car insurance premiums and provide valuable insights for business strategy.

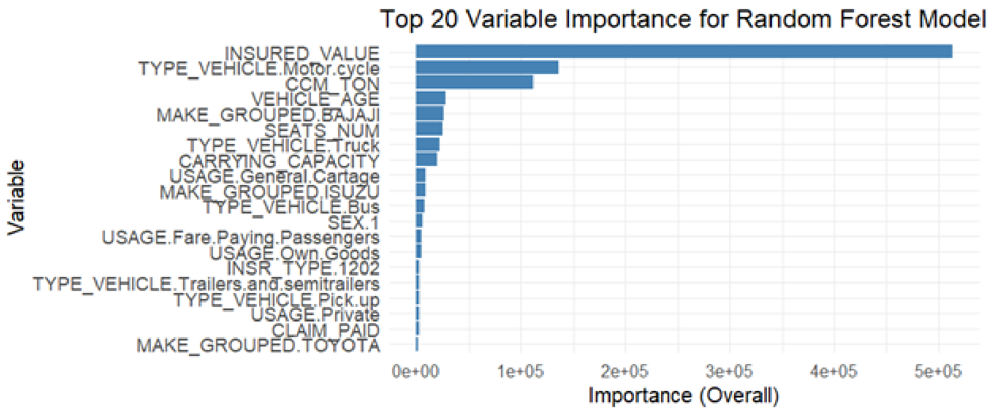


Figure 16: Top 20 Variable Importance for Random Forest Model

8 Conclusion

This project successfully demonstrated the robust application of machine learning for highly accurate car insurance premium prediction. Through a structured data science workflow, including rigorous data preprocessing and comprehensive model evaluation, the objective of developing a reliable predictive tool was met. Among the models developed, the **Random Forest Regressor** proved to be the standout performer, significantly outperforming both Linear Regression and XGBoost. It achieved an impressive **R-squared value of 0.954** and the lowest RMSE and MAE on the unseen test set, indicating its strong capability to explain over 95% of the variance in the log-transformed premium. This performance underscores how advanced, non-linear ensemble models can vastly improve upon traditional actuarial methods for complex premium determination.

The successful implementation of this model offers profound implications for the insurance industry. Firstly, it enables **Risk-Based Pricing**, allowing insurers to determine fairer, more individualized premiums that directly reflect real risk indicators for each policyholder. Secondly, the automation of the premium prediction process streamlines underwriting, leading to enhanced **Operational Efficiency**. Lastly, by providing accurate predictions for optimized pricing strategies, the model significantly bolsters **Market Competitiveness**, which can help reduce customer churn and improve overall profitability.

Further insights were derived from the variable importance analysis of the Random Forest model. This analysis identified the key factors that most significantly influence insurance premiums. These top influential variables and their interpretations are summarized in Table 2.

Table 2: Top Influential Variables in Premium Prediction

Variable	**Interpretation**
INSURED_VALUE	Major predictor, represents insurer's financial exposure.
TYPE_VEHICLE.Motor.cycle	Highlights vehicle-type based premium segmentation.
CCM_TON	Correlates with engine size and vehicle power.
VEHICLE_AGE	Captures depreciation and risk variability over time.

This project not only showcases robust technical proficiency in data manipulation, model development, and performance evaluation, but also yields concrete, actionable intelligence for strategic decision-making in the insurance sector. The insights gained and the high-performing model developed lay a strong foundation for future advancements, paving the way for more efficient, equitable, and competitive market practices.

References

[1] Max Kuhn et al. Predictive modeling with r and the caret package. *Google Scholar*, 2013.

[2] Brett Lantz. *Machine learning with R: expert techniques for predictive modeling*. Packt publishing ltd, 2019.

[3] Suresh Kumar Mukhiya and Usman Ahmed. *Hands-On Exploratory Data Analysis with Python: Perform EDA techniques to understand, summarize, and investigate your data*. Packt Publishing Ltd, 2020.

[4] Sinan Ozdemir and Divya Susarla. *Feature Engineering Made Easy: Identify unique features from your dataset in order to build powerful machine learning systems*. Packt Publishing Ltd, 2018.

[5] Priyanka P Shinde, Kavita S Oza, and RK Kamat. Big data predictive analysis: Using r analytical tool. In *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*, pages 839–842. IEEE, 2017.