Choose Your Own Project: Interstate Traffic Volume Prediction Using Machine Learning

Mujahid Ali

2023-12-31

Contents

1 Project Overview

This assignment for the 'Data Science: Capstone' course (PH125.9x) by HarvardX through edX focuses on applying advanced machine learning techniques to a public dataset, moving beyond standard linear regression. The goal is to analyze the data effectively and communicate the insights clearly, demonstrating both technical proficiency and the ability to translate complex data-driven findings into understandable terms. This project emphasizes the practical application of data science skills and clear communication in the field.

1.1 Introduction

Traffic volume refers to the number of vehicles traversing a specific point on a road within a given timeframe. This metric is vital for local councils, as it provides insights into the usage intensity of various routes. By analyzing traffic counts, authorities can identify heavily utilized roads. This information is crucial for infrastructure planning, enabling decisions on road improvements or the development of alternative routes to alleviate excessive traffic. Such data-driven approaches are essential for effective urban planning and traffic management, aiming to enhance road efficiency and safety.

1.2 Project Description

This project aims to leverage machine learning models to predict traffic volume on an American interstate. A key part of the project involves understanding the critical features that influence traffic flow. To enhance the accuracy of predictions, the project will incorporate data transformation and feature engineering techniques. A range of machine learning approaches will be explored and evaluated. The selection of the best model will be based on performance metrics, with a particular focus on the Root Mean Square Error (RMSE). This approach will not only provide insights into traffic patterns but also help in identifying the most effective methods for traffic volume prediction.

1.3 Dataset Overview

This project utilizes the Metro Interstate Traffic Volume Dataset from the UCI Machine Learning Repository. The dataset comprises hourly traffic volume data for the westbound Interstate 94 (I-94), enriched with weather and holiday information spanning from 2012 to 2018.

Interstate 94 is a crucial east-west highway in the United States, linking the Great Lakes and northern Great Plains regions. It stretches from Billings, Montana to Port Huron, Michigan. The specific focus of this dataset is a point on the I-94 approximately midway between Minneapolis and St Paul, Minnesota. This location serves as the primary measurement site for the traffic volume data. The dataset's comprehensive nature, including various influencing factors like weather and holidays, provides a robust foundation for predictive modeling and analysis in this project.

1.4 Dataset Details

This project utilizes data from the UCI Machine Learning Repository, combining traffic information from the MN Department of Transportation and weather data from OpenWeatherMap. The dataset includes several key variables:

Response Variable:

- Traffic Volume: Numeric, representing hourly traffic volume.

Features:

- Holiday: Categorical - US national holidays and regional holidays.
- Temperature: Numeric - average temperature in Kelvin.
- Rain: Numeric - amount in mm of rain during the hour.
- Snow: Numeric - amount in mm of snow during the hour.
- Clouds: Numeric - percentage of cloud cover.
- Weather Main: Text - brief description of the current weather.
- Weather Description: Text - detailed description of the current weather.
- Date Time: DateTime - hour of data collection in local CST time.

In the following sections, we will explore and visualize this data, focusing on data transformation and feature engineering to improve our predictive model.

2 Methods and Analysis

2.1 Data Load, Analysis and Preparation

We start by loading essential libraries for data manipulation and analysis:

tidyverse: An ecosystem of packages for data manipulation, visualization, and data science workflows.

lubridate: Makes it easier to work with dates and times in R.

caret: Provides functions for training and plotting a wide variety of predictive models.

R.utils: Contains a variety of utility functions for programming and manipulation of R objects.

knitr: Allows for dynamic report generation in R.

kableExtra: Extends 'knitr::kable()' output by enabling additional styling and formatting of tables.

tictoc: Functions for timing R scripts, which can be used to monitor performance and bottlenecks.

xgboost: An optimized distributed gradient boosting library designed to be highly efficient, flexible, and portable.

Data Loading

We will download the data directly from UCI Machine Learning Repository.

```
temp <- tempfile()
download.file("https://archive.ics.uci.edu/ml/machine-learning-databases/00492/Metro_Interstate_Traffic
try(gunzip(temp, "Metro_Interstate_Traffic_Volume.csv"))
metro <- read.csv("Metro_Interstate_Traffic_Volume.csv")
rm(temp)

#Preview of the dataset structure (limited columns):
limited_glimpse <- function(data, max_cols = 10) {
  cols <- min(ncol(data), max_cols)
  glimpse(select(data, 1:cols))
  if (ncol(data) > cols) {
    cat("...\n")
  }
}

limited_glimpse(metro)
```

```
## Rows: 48,204
## Columns: 9
## $ holiday             <chr> "None", "None", "None", "None", "None", "None", "N~
## $ temp                <dbl> 288.28, 289.36, 289.58, 290.13, 291.14, 291.72, 29~
## $ rain_1h             <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ snow_1h             <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ clouds_all          <int> 40, 75, 90, 90, 75, 1, 1, 1, 20, 20, 20, 1, 1, 1, ~
## $ weather_main        <chr> "Clouds", "Clouds", "Clouds", "Clouds", "Clouds", ~
## $ weather_description <chr> "scattered clouds", "broken clouds", "overcast clo~
## $ date_time           <chr> "2012-10-02 09:00:00", "2012-10-02 10:00:00", "201~
## $ traffic_volume      <int> 5545, 4516, 4767, 5026, 4918, 5181, 5584, 6015, 57~
```

Before diving into the analysis, it is crucial to prepare the dataset. This involves several steps to ensure the data is in the desired format for effective analysis. The process includes:

Data Cleaning: We'll address any imputation problems in the dataset.

Handling Duplications: It's essential to identify and remove any duplicate records to maintain data integrity.

Data Type Conversion: Some variables will be converted to factors (categorical variables) for more appropriate analysis.

Feature Engineering: We will create new features that might be more indicative of the patterns we are trying to analyze.

The dataset comprises 48,204 hourly records, including traffic volume, weather conditions, and holiday information. The records span from October 2, 2012, to September 30, 2018. However, it's important to note a gap in the data between August 2014 and June 2015 where no records are available.

In the following sections, we'll outline the specific steps taken in each of these areas to prepare our dataset for comprehensive analysis.

```
## Rows: 47,959
## Columns: 16
## $ holiday        <chr> "None", "None", "None", "None", "None", "None", "None",~
## $ temp           <dbl> 288.28, 289.36, 289.58, 290.13, 291.14, 291.72, 293.17,~
## $ rain_1h        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ snow_1h        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ clouds_all     <int> 40, 75, 90, 90, 75, 1, 1, 1, 20, 20, 20, 1, 1, 1, 1, 1,~
## $ weather_main   <chr> "Clouds", "Clouds", "Clouds", "Clouds", "Clouds", "Clea~
## $ date_time      <chr> "2012-10-02 09:00:00", "2012-10-02 10:00:00", "2012-10-~
## $ traffic_volume <int> 5545, 4516, 4767, 5026, 4918, 5181, 5584, 6015, 5791, 4~
## $ date           <date> 2012-10-02, 2012-10-02, 2012-10-02, 2012-10-02, 2012-1~
## $ hour           <fct> 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, ~
## $ month          <fct> 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10,~
## $ year           <fct> 2012, 2012, 2012, 2012, 2012, 2012, 2012, 2012, 2012, 2~
## $ weekday        <fct> Tuesday, Tuesday, Tuesday, Tuesday, Tuesday, Tuesday, T~
## $ is_holiday     <chr> "No", "No", "No", "No", "No", "No", "No", "No", "No", "~
## $ holiday_pre    <fct> No, No, No, No, No, No, No, No, No, No, No, No, No, No,~
## $ holiday_pos    <fct> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
```

This dataset exhibits significant duplication issues. It contains 17 instances of complete observation replication and a notable 7,629 cases where entries, although identical in date-time, differ solely in their weather descriptions. A further inconsistency is observed in weather data accuracy: several records are categorized under thunderstorms or rain-related phenomena, yet paradoxically report zero millimeters of precipitation within the same timeframe. This discrepancy is mirrored in the data concerning snowfall, underscoring the need for rigorous data validation and cleansing to ensure the dataset's integrity and utility in analysis.

We eliminated duplicate observations and experimented with models both including and excluding the weather description feature. Models that omitted this variable showed marginally superior performance. Consequently, this report will focus exclusively on analyses that exclude the weather description feature.

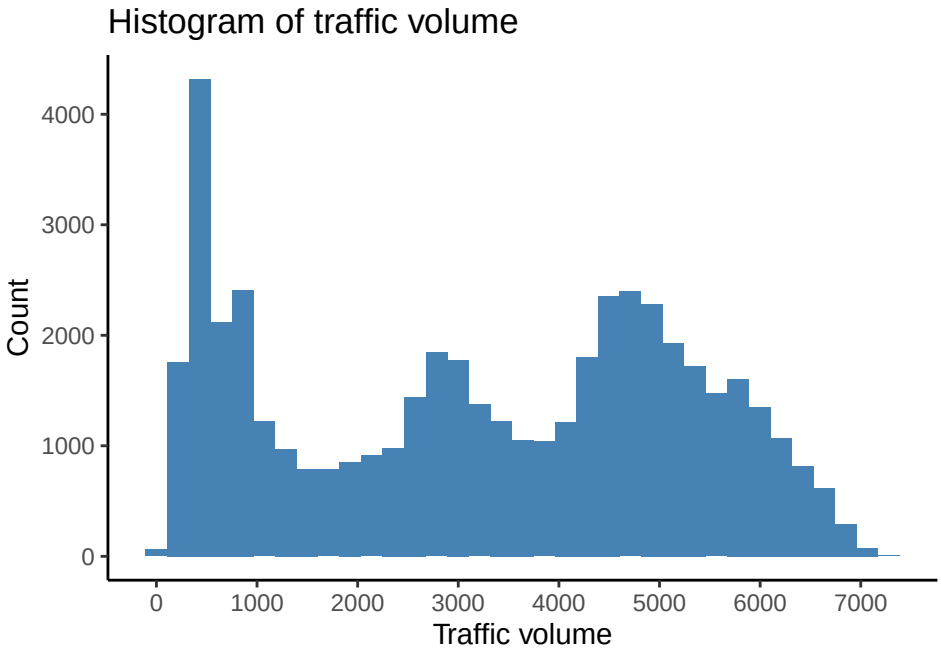
We derived new variables from the existing dataset, namely day, hour, month, year, and weekday. The date "2016-12-26" was erroneously labeled as Christmas Day; we corrected this to "2016-12-25". Additionally, we observed that holidays were initially marked only during their first hour (e.g., "2012-12-25 00:00:00" as Christmas Day, but not subsequent hours). This labeling was rectified, increasing holiday-tagged observations from 61 to 1,203. We also introduced a binary variable indicating whether a day is a holiday and two others to denote if it's adjacent to a holiday. In cases of multiple temperature, rain, or cloud cover measurements, we computed their average.

Our review identified ten records with an implausible temperature of 0 Kelvin, a phenomenon never recorded on Earth. Similarly, one instance showed rainfall at 9831.3 mm per hour, far exceeding the highest known record of 305 mm/hour. We have removed all these eleven anomalous observations from our dataset.

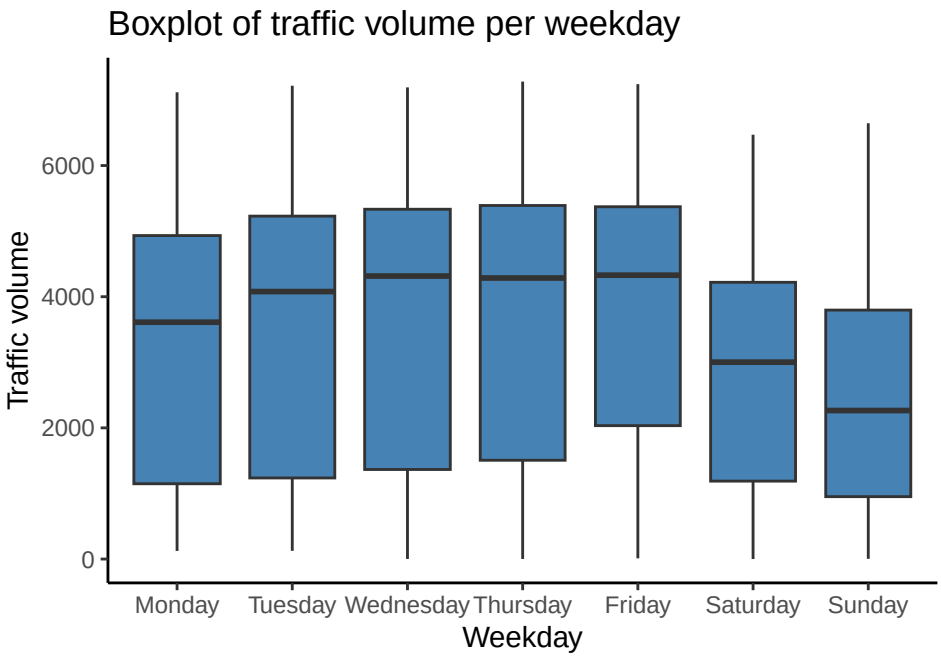
2.2 Exploratory Data Analysis

Exploratory Data Analysis (EDA) is an essential preliminary step in data investigation, where patterns are identified, anomalies detected, hypotheses tested, and assumptions verified through summary statistics and visual methods.

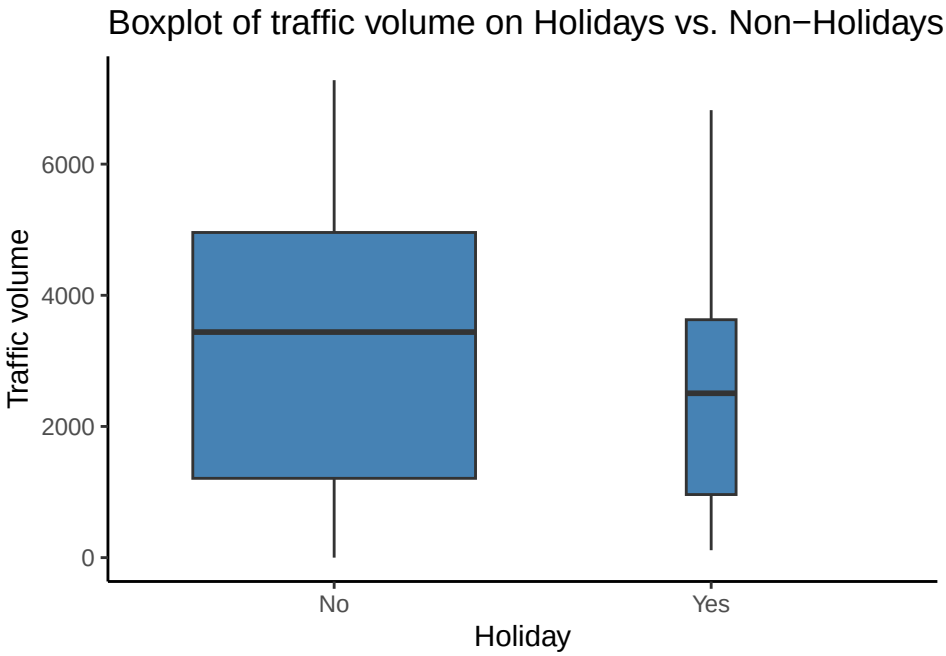
The analysis reveals that the traffic volume data exhibits a multimodal distribution characterized by three distinct peaks. The first peak, representing the most frequent traffic volume, falls below 2500 vehicles per hour. The second peak is observed at approximately 3000 vehicles per hour, while the third peak occurs around the 4500 vehicles per hour mark.



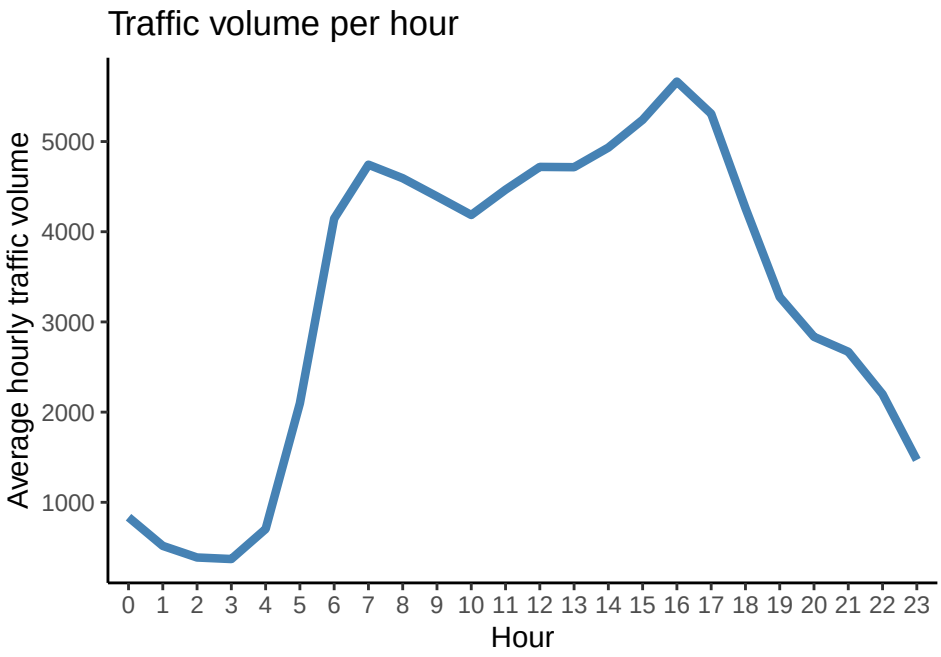
Some new features were created from the original dataset, such as the weekday. As shown in the boxplot below, the traffic volume appears to increase slowly over the weekdays and is considerably lower on weekends.



As presented in the boxplot below, the traffic volume appears to be slightly lower during the holidays.



The traffic volume demonstrates significant hourly variations, suggesting its potential as a valuable predictor in our model. Notably, the day's first major peak occurs early, from 6 to 7 am. Following a slight decrease in late morning, traffic volume surges post-lunch, reaching its zenith between 4 and 5 pm. This pattern underscores the importance of incorporating time of day as a key variable in our predictive analysis.



In machine learning, it's crucial to differentiate between two types of datasets: the training dataset and the test dataset. The training dataset is used to develop and fine-tune the algorithm, while the test dataset provides an unbiased assessment of the final model's performance. For our study, the training set comprises all data up to the year 2017, encompassing 34,031 observations, and the test set includes data from 2018, with 6,533 observations.

Given that only 31 observations recorded snow, and none were in the test data from the last year, we have decided not to include this feature in our analysis. Thus, the selected features for modeling are holiday, proximity to a holiday (either the previous or following day), temperature, cloud cover percentage, hour of the day, and weekday. These choices are geared towards creating a robust and relevant predictive model.

3 Modeling Approaches

To guide our selection of the most suitable machine learning model, preliminary experiments were conducted on a one-year subset of the data using various algorithms, including elastic net, bagged tree, and SVM. Based on a balance between Root Mean Square Error (RMSE) and execution time, Caret's eXtreme Gradient Boosting (xgbTree) was selected as the optimal choice. To maintain the conciseness of this report, and considering the extended run times of some models, details of these initial experiments are excluded. The focus will instead be on fine-tuning the chosen boosting model.

3.1 Linear model and xgbTree model with default hyperparameters

To establish benchmarks, we configured two baseline models: a straightforward linear regression model and the xgbTree model using its default hyperparameters. The purpose of these baselines is to gauge the impact of hyperparameter tuning on model performance. Both models were fitted using the same training control, employing 3-fold cross-validation for robustness.

The eXtreme Gradient Boosting (xgbTree) model incorporates seven tunable parameters: the number of boosting iterations, maximum tree depth, shrinkage, gamma (minimum loss reduction required for further partition), column subsample ratio, minimum sum of instance weight needed in a child, and subsample percentage of the training instance. We plan to adjust these parameters incrementally to keep the size of

	Predictor	RMSE..train.	RMSE..test.	R.squared..train.	Time.secs.
elapsed	Linear model	791.14	801.81	0.84	4.29
elapsed1	xgbTree - Default	529.70	559.60	0.92	72.21

our hyperparameter grid manageable and focused. This approach aims to optimize the model's performance without excessively complicating the tuning process.

The implementation of the default eXtreme Gradient Boosting model showed a significant improvement in RMSE, dropping from 791.14 in the linear regression model to 529.7 in the training set.

Default model hyperparameters:

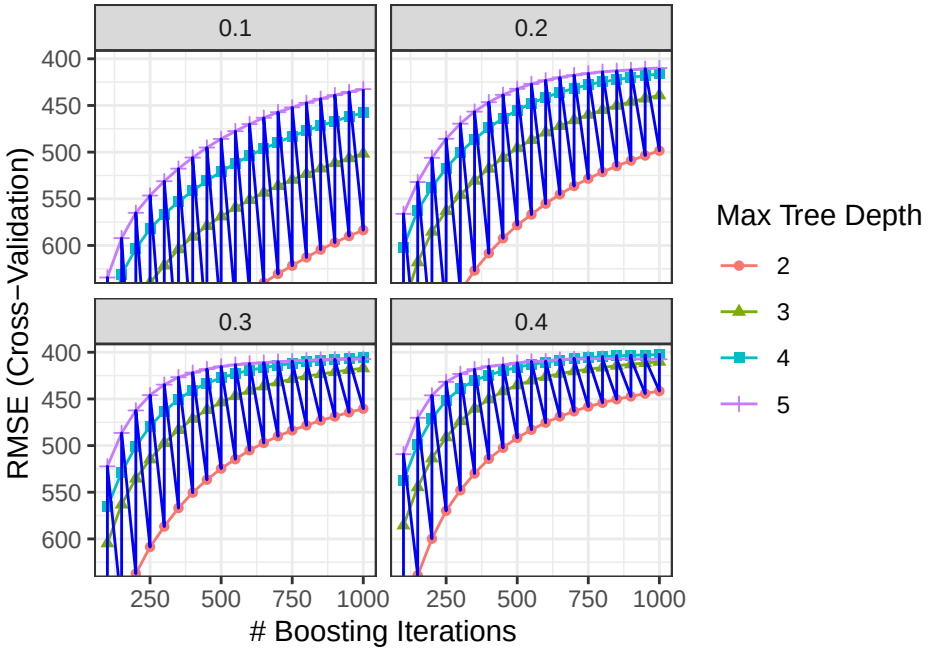
- nrounds: 150
- max_depth: 3
- eta: 0.4
- gamma: 0
- colsample_bytree: 0.8
- min_child_weight: 1
- subsample: 0.5

Hyperparameter tuning strategy:

To balance the trade-off between model performance and computational efficiency, the number of boosting iterations is capped at 1000. Post tuning the other parameters, this limit will be re-evaluated.

3.2 xgbTree - Step 1: Number of iterations and the learning Rate

For the first step, we created a grid search with different boosting iterations, shrinkage and max tree depth.



	Predictor	RMSE..train.	RMSE..test.	R.squared..train.	Time.secs.
elapsed	Linear model	791.14	801.81	0.84	4.29
elapsed1	xgbTree - Default	529.70	559.60	0.92	72.21
elapsed2	xgbTree - Step 1	306.62	455.04	0.95	317.87

The best model found within the grid search had the tuning parameters nrounds = 1000, max_depth = 3 and eta = 0.4. As shown in the graph above, for lower shrinkage the model does not seem stable. This first tuning already improved the RMSE considerably, from 529.7 to 306.62. This is a 42.11% decrease.

3.3 xgbTree - Step 2: Maximum Depth and Minimum Child Weight

With the shrinkage value set to the optimal level identified previously, we now proceed to a focused grid search. This search will concentrate on two key hyperparameters: minimum child weight and maximum tree depth.

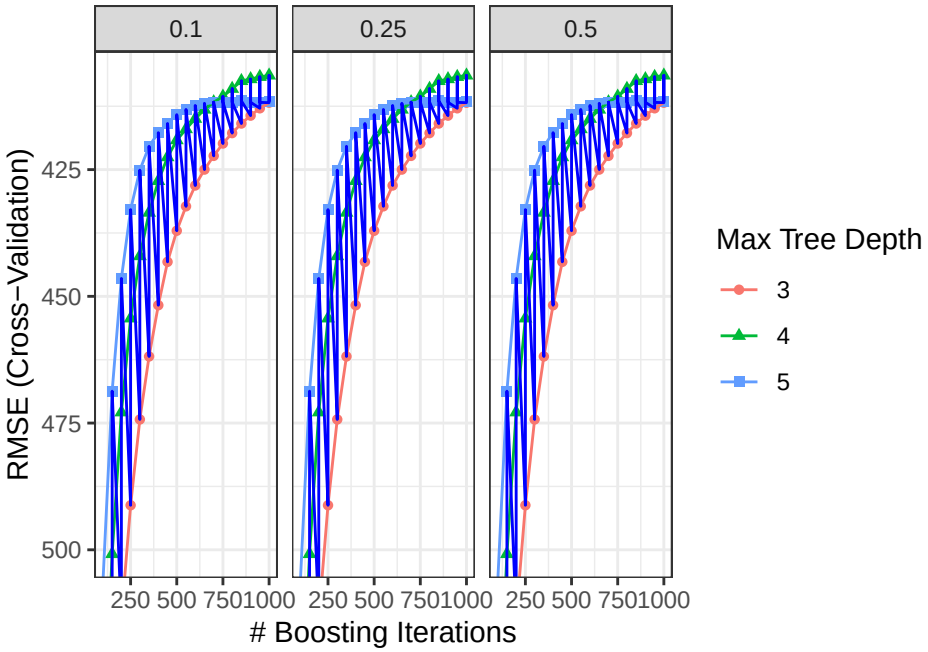
Grid search strategy:

Shrinkage: Fixed to the optimal value (insert the optimal value here).

Minimum Child Weight: This parameter will be varied to find the most effective setting.

Maximum Tree Depth: Set to 3 ± 1, exploring one level above and one below the best tune identified in the earlier step.

This grid search aims to refine our model further by meticulously adjusting these parameters within a targeted range, thereby optimizing the model's performance.



The grid search concluded with the identification of the best-performing model within the explored parameter space. This model's tuning parameters and corresponding RMSE are detailed below:

Maximum Tree Depth (`max_depth`): 4

Minimum Child Weight (`min_child_weight`): 0.1

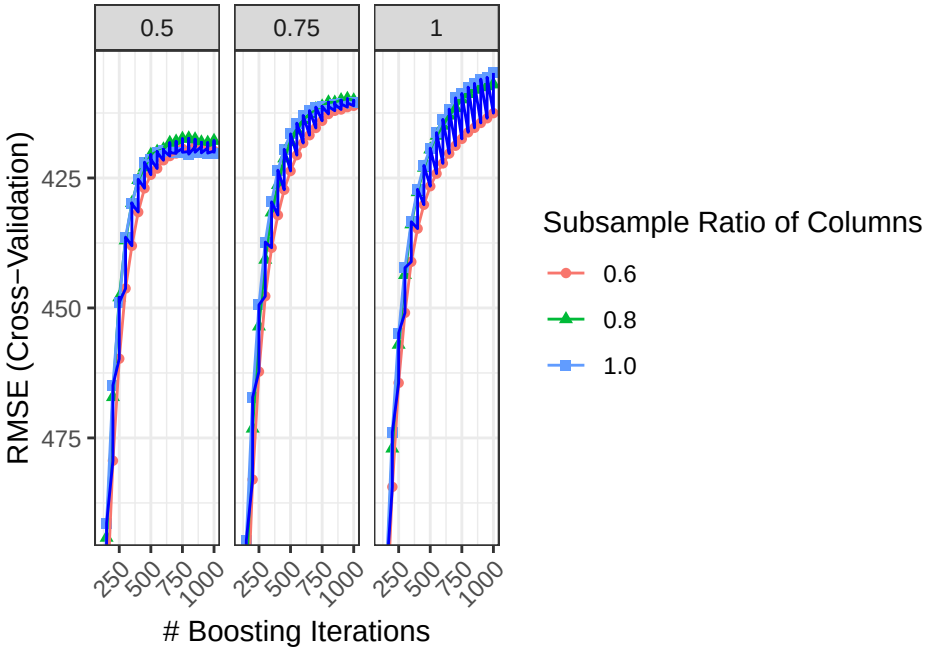
	Predictor	RMSE..train.	RMSE..test.	R.squared..train.	Time.secs.
elapsed	Linear model	791.14	801.81	0.84	4.29
elapsed1	xgbTree - Default	529.70	559.60	0.92	72.21
elapsed2	xgbTree - Step 1	306.62	455.04	0.95	317.87
elapsed3	xgbTree - Step 2	306.62	455.04	0.96	205.28

	Predictor	RMSE..train.	RMSE..test.	R.squared..train.	Time.secs.
elapsed	Linear model	791.14	801.81	0.84	4.29
elapsed1	xgbTree - Default	529.70	559.60	0.92	72.21
elapsed2	xgbTree - Step 1	306.62	455.04	0.95	317.87
elapsed3	xgbTree - Step 2	306.62	455.04	0.96	205.28
elapsed4	xgbTree - Step 3	306.62	455.04	0.96	199.64

Interestingly, the RMSE remained consistent (306.62) across variations in minimum child weight. This suggests that adjusting `min_child_weight` within the tested range does not significantly impact the model's prediction error. This finding provides valuable insights into the sensitivity of our model to changes in this specific parameter, guiding future tuning and model refinement efforts.

3.4 xgbTree - Step 3: Subsample ratio of columns and subsample percentage

In the next step, we fix the minimum child weight to the optimal value found previously, set the maximum tree depth to 3 and do a grid search on the subsample ratio of columns and subsample percentage.



The completion of the grid search has resulted in the identification of an optimal model, which interestingly aligns with the parameters used in Step 2. The details of the tuning parameters and the model's performance are as follows:

- ***Column Sample by Tree (`colsample_bytree`)***: 1

	Predictor	RMSE..train.	RMSE..test.	R.squared..train.	Time.secs.
elapsed	Linear model	791.14	801.81	0.84	4.29
elapsed1	xgbTree - Default	529.70	559.60	0.92	72.21
elapsed2	xgbTree - Step 1	306.62	455.04	0.95	317.87
elapsed3	xgbTree - Step 2	306.62	455.04	0.96	205.28
elapsed4	xgbTree - Step 3	306.62	455.04	0.96	199.64
elapsed5	xgbTree - Step 4	306.62	455.04	0.94	161.10

- ***Subsample***: 1

Given these parameters are identical to those employed in Step 2, it's notable that the RMSE has remained unchanged. This outcome reinforces the effectiveness of the previously established settings and indicates that further adjustments in these particular parameters may not yield significant improvements in model accuracy.

This consistency in RMSE underscores the robustness of our model's performance with these specific hyperparameter settings.

3.5 xgbTree - Step 4: Gamma

Now we will fix the colsample_bytree and subsamples tuning parameters and perform a grid search on gamma (minimum loss reduction parameter).

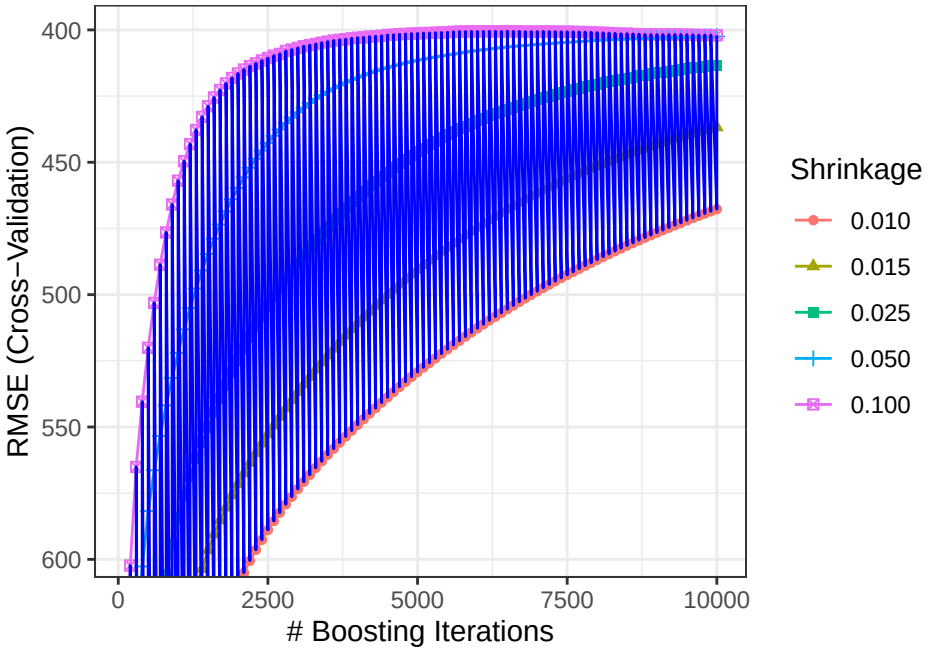
Different gamma values did not have any effect on the model fit (RMSE), so we continue with the previous value.

3.6 xgbTree - Step 5: Reducing the Learning Rate

Now that we have tunned all hyperparameters parameters, we can go back and try different values for the number of boosting iterations and shrinkage. Before, we tried up until 1000 iterations to save running time, but now the grid search executes up to 10000 iterations.

nrounds	eta	max_depth	gamma	colsample_bytree	min_child_weight	subsample
6500	0.1	4	0	1	0.1	1

	Predictor	RMSE..train.	RMSE..test.	R.squared..train.	Time.secs.
elapsed	Linear model	791.14	801.81	0.84	4.29
elapsed1	xgbTree - Default	529.70	559.60	0.92	72.21
elapsed2	xgbTree - Step 1	306.62	455.04	0.95	317.87
elapsed3	xgbTree - Step 2	306.62	455.04	0.96	205.28
elapsed4	xgbTree - Step 3	306.62	455.04	0.96	199.64
elapsed5	xgbTree - Step 4	306.62	455.04	0.94	161.10
elapsed6	xgbTree - Step 5	283.58	457.24	0.96	1110.83
elapsed7	xgbTree - Final model	283.58	457.24	0.96	197.94



3.7 Results

The final model had the following tuning parameters:

We now assess the model's performance on the test dataset. A well-fitted model that doesn't overfit is expected to exhibit similar performance metrics on both training and test sets. While a slight increase in RMSE on the test set is anticipated, the key is to maintain a reasonable level of accuracy.

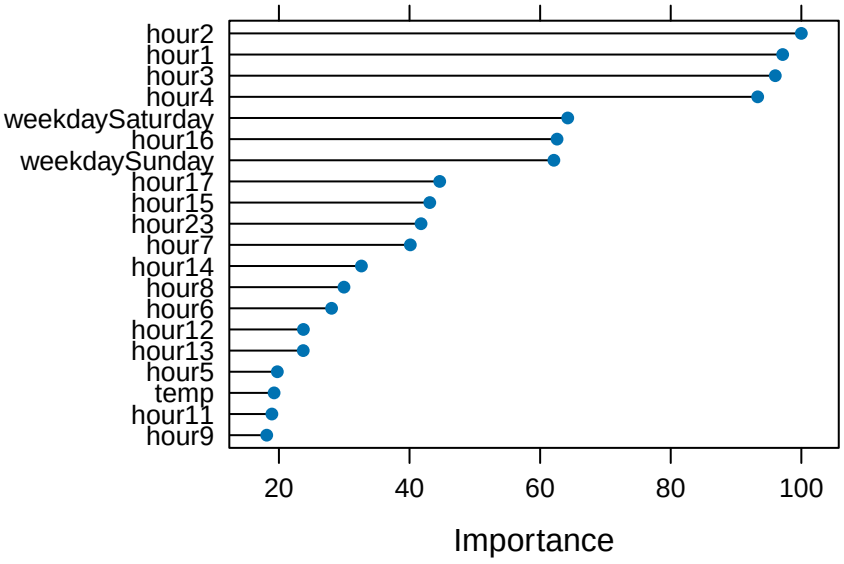
Training Dataset RMSE: 283.58

Test Dataset RMSE: 457.24

The observed RMSE for the test dataset is slightly higher than that of the training dataset, which is a typical and expected result. However, the test RMSE of 457.24 still signifies a robust fit. Notably, it outperforms the training RMSE of the default hyperparameter xgbTree model. This indicates that our model, even after

fine-tuning, maintains a strong predictive capability and effectively handles unseen data. This outcome is promising and underscores the model's generalization ability, a critical aspect of machine learning models.

Beyond the modeling aspect, gaining insights into the factors that significantly influence traffic volume is crucial. This understanding can inform more targeted traffic management and planning strategies.From the analysis represented in the accompanying graph, we identify the 20 most impactful variables. The findings highlight 'Hour' as the most influential factor in determining traffic volume. This is a critical insight, as it underscores the time-of-day dependency of traffic flow. Additionally, the days 'Sunday' and 'Saturday' emerge as significant variables. This observation suggests a noticeable variation in traffic patterns during weekends compared to weekdays.



4 Conclusion

The aim of this project was to employ machine learning techniques to forecast traffic volume on an American interstate, while also identifying key features influencing traffic patterns. Throughout this study, we experimented with various models, engineered new features, and ultimately selected Caret's eXtreme Gradient Boosting (xgbTree) model, balancing Root Mean Square Error (RMSE) and computational efficiency. We fine-tuned seven different hyperparameters of the model, achieving an RMSE of 306.62 on the training set and 455.04 on the test set. The analysis revealed that the most critical feature for explaining traffic was the hour of the day.

5 References

- James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). "An Introduction to Statistical Learning with Applications in R." Springer.
- Trevor Hastie, Robert Tibshirani, Jerome Friedman. (2021). "The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition." Springer Series in Statistics.
- Papacostas, C. S., & Prevedouros, P. D. (2020). "Transportation Engineering and Planning." Pearson.
- Levinson, D., & Krizek, K. J. (2008). "Planning for Place and Plexus: Metropolitan Land Use and Transport." Routledge.

- Elements of AI (2020). “Understanding the basics of XGBoost and Gradient Boosting.”