Introduction

In the domain of machine learning and predictive modeling, the last few decades have witnessed remarkable advancements. These advancements have been driven by the increasing availability of complex datasets and the development of sophisticated algorithms capable of extracting meaningful patterns and making accurate predictions. This progress is particularly evident in fields requiring real-time analysis and prediction, such as traffic congestion management, speed prediction, and travel time estimation. The ability to predict traffic conditions accurately is crucial for enhancing urban mobility, optimizing transportation systems, and reducing environmental impacts.

This literature review aims to provide a detailed and comprehensive overview of the fundamental concepts, methodologies, and significant advancements relevant to developing a predictive model for traffic congestion, speed prediction, and travel time estimation using the Gradient Boosting Regressor (GBR). GBR, a powerful ensemble learning technique, has shown great promise in handling various predictive tasks due to its ability to model complex relationships within data and its robustness against overfitting.

The review will begin by exploring the evolution of predictive modeling techniques, tracing the development from traditional statistical methods to modern machine learning approaches. This historical perspective will highlight how advancements in computing power and data availability have transformed predictive modeling capabilities, enabling the handling of increasingly complex datasets.

Next, the focus will shift to the specific application of machine learning in traffic management. This section will delve into how machine learning algorithms, particularly GBR, have been applied to predict traffic congestion, estimate vehicle speeds, and forecast travel times. Case studies and examples from existing literature will be examined to illustrate the practical applications and benefits of these techniques in real-world traffic management scenarios.

A critical aspect of predictive modeling is data preprocessing, which involves preparing raw data for analysis. This review will cover various preprocessing strategies, including data cleaning, feature selection, and feature engineering. The importance of these steps in ensuring the quality and accuracy of the predictive models will be emphasized, as well as the challenges associated with dealing with large and heterogeneous traffic datasets.

Finally, the review will address model evaluation metrics, which are essential for assessing the performance of predictive models. Different metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared will be discussed in the context of their applicability to traffic prediction tasks. The importance of selecting appropriate evaluation metrics to ensure reliable and interpretable results will be highlighted.

Through this detailed examination of the key concepts, methodologies, and advancements in predictive modeling for traffic management using Gradient Boosting Regressor, this literature review aims to provide a solid foundation for researchers and practitioners in the field. By understanding the current state of the art and the underlying principles, future efforts can be directed towards developing more effective and efficient predictive models, ultimately contributing to smarter and more sustainable urban transportation systems.  
  
Predictive Modeling in Traffic Management

Predictive modeling plays a pivotal role in traffic management by anticipating congestion, estimating travel times, and optimizing traffic flow. As urban areas become increasingly congested, the ability to predict traffic conditions accurately has become essential for transportation planners, policymakers, and commuters. Over the years, the methodologies used in predictive modeling have evolved significantly, transitioning from traditional statistical approaches to advanced machine learning techniques.

Early Approaches: Statistical Methods and Time-Series Analysis

In the initial stages of traffic prediction, researchers primarily relied on statistical methods and time-series analysis. These methods, rooted in statistical theory, provided a framework for understanding and forecasting traffic patterns based on historical data. Notable techniques included Kalman filtering and AutoRegressive Integrated Moving Average (ARIMA) models.

Kalman Filtering: This method is well-known for its application in dynamic systems. Kalman filters recursively estimate the state of a process by minimizing the mean of the squared error. In traffic prediction, Kalman filters were employed to estimate traffic flow and speed by continuously updating predictions based on incoming traffic data. Despite its effectiveness in certain scenarios, the method struggled with the inherently nonlinear and stochastic nature of traffic flow.

ARIMA Models: ARIMA models are widely used in time-series analysis for their ability to handle data with trends and seasonality. These models apply differencing to make the time series stationary, then fit a combination of autoregressive (AR) and moving average (MA) models. While ARIMA models provided a structured approach to forecasting traffic data, their linear nature often fell short in capturing the complex, nonlinear relationships present in traffic systems (Vlahogianni et al., 2014).

The Advent of Machine Learning: Neural Networks, Support Vector Machines, and Ensemble Methods

With the advent of machine learning, the landscape of predictive modeling in traffic management experienced a transformative shift. Machine learning algorithms, designed to learn from data and identify intricate patterns, offered significant improvements in predictive accuracy and robustness.

Neural Networks: Neural networks, particularly deep learning models, have shown exceptional performance in handling large-scale and high-dimensional traffic data. These models are capable of learning complex, nonlinear relationships through multiple layers of abstraction. Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks, in particular, have been effective in capturing temporal dependencies in traffic data, making them suitable for time-series forecasting.

Support Vector Machines (SVMs): SVMs are another class of powerful machine learning models used in traffic prediction. They work by finding the optimal hyperplane that separates different classes in the data. For regression tasks, SVMs find the hyperplane that best fits the data, providing robust predictions. SVMs have been used to predict traffic flow and speed, often outperforming traditional statistical methods.

Ensemble Methods: Ensemble methods combine multiple models to improve predictive performance. Among these, Gradient Boosting Regressor (GBR) has emerged as a particularly powerful tool. GBR builds an ensemble of decision trees in a sequential manner, where each tree corrects the errors of its predecessor. This approach enhances the model’s ability to handle various types of data and noise, making it well-suited for the dynamic and complex nature of traffic prediction (Lv et al., 2015).

Gradient Boosting Regressor: A Robust Tool for Traffic Prediction

Gradient Boosting Regressor (GBR) stands out among ensemble methods due to its robustness and flexibility. GBR operates by iteratively adding weak learners (typically decision trees) to form a strong predictive model. Each new tree is fitted to the residual errors of the previous trees, thereby focusing on the areas where the model’s performance is weakest.

The strengths of GBR in traffic management include:

Handling Nonlinearity: GBR effectively captures nonlinear relationships in traffic data, which are common due to the complex interactions between different traffic variables.

Reducing Overfitting: Through techniques such as regularization and tree pruning, GBR mitigates the risk of overfitting, ensuring that the model generalizes well to new, unseen data.

Flexibility: GBR can handle various types of data, including continuous and categorical variables, making it versatile for different traffic prediction tasks.

Noise Robustness: The iterative nature of GBR allows it to handle noisy data by focusing on minimizing prediction errors, making it resilient to anomalies and outliers in traffic data.

The evolution of predictive modeling techniques from traditional statistical methods to advanced machine learning algorithms has significantly enhanced the accuracy and reliability of traffic predictions. Gradient Boosting Regressor, with its robustness and flexibility, represents a cutting-edge approach in this domain, providing a powerful tool for anticipating traffic congestion, estimating travel times, and optimizing traffic flow. This progression underscores the importance of continuous innovation and adaptation in predictive modeling to meet the ever-growing demands of urban traffic management.  
  
Gradient Boosting Regressor

Gradient Boosting Regressor (GBR) is a powerful ensemble learning technique designed to enhance the predictive accuracy of models by combining the strengths of multiple weak learners, typically decision trees. The foundational concept behind GBR is boosting, a process that sequentially adds new models to correct the errors made by the previous models, thus improving overall performance and accuracy. This methodology was formally introduced by Friedman in 2001 and has since become a cornerstone in the machine learning community due to its robustness and versatility (Friedman, 2001).

Core Mechanism of Gradient Boosting Regressor

The GBR algorithm operates through a series of iterations, where each new decision tree is trained to predict the residual errors of the combined ensemble of all previous trees. This iterative process focuses on refining predictions by emphasizing areas where previous models underperformed, leading to a model that is both accurate and reliable.

Initialization: The process begins by initializing the model with a simple prediction, often the mean value of the target variable.

Sequential Model Building: In each subsequent iteration, a new decision tree is trained on the residuals (errors) of the current ensemble. The goal of this new tree is to correct the errors made by the previous ensemble of trees. This step involves:

Calculating the residuals for each data point.

Training a new decision tree to predict these residuals.

Updating the model by adding this new tree to the ensemble with a specific weight (learning rate).

Model Update: The predictions from the new tree are scaled by the learning rate and added to the existing ensemble’s predictions. The learning rate is a crucial hyperparameter that controls the contribution of each new tree, balancing between the speed of learning and the risk of overfitting.

Iteration: This process repeats for a predefined number of iterations or until the model’s performance no longer improves on a validation set.

Final Model: The final predictive model is a weighted sum of the individual decision trees, each contributing to refining the overall prediction.

Advantages of Gradient Boosting Regressor

GBR offers several significant advantages that make it a preferred choice for various predictive modeling tasks:

High Prediction Accuracy: Due to its iterative refinement process, GBR often achieves high levels of accuracy by effectively minimizing prediction errors across iterations.

Handling Missing Data: GBR can manage datasets with missing values without the need for imputation, leveraging the flexibility of decision trees to handle incomplete data.

Robustness to Overfitting: When appropriately tuned, GBR includes mechanisms like regularization and tree pruning that help mitigate the risk of overfitting, ensuring that the model generalizes well to new data.

Flexibility: GBR can handle mixed data types, including both numerical and categorical variables, making it suitable for complex datasets with diverse features.

Applications of Gradient Boosting Regressor in Traffic Management

GBR has found extensive applications across various domains, including finance, healthcare, and particularly traffic management. In the context of traffic management, GBR’s ability to handle large, heterogeneous datasets and its robustness to noise make it an ideal choice for several predictive tasks:

Traffic Congestion Prediction: GBR has been effectively used to forecast traffic congestion levels, enabling city planners and transportation authorities to anticipate and mitigate traffic jams. By analyzing historical traffic data along with real-time inputs, GBR models can provide accurate congestion predictions, helping to optimize traffic flow and reduce delays.

Travel Time Estimation: Accurate travel time prediction is crucial for both commuters and logistics companies. GBR models can estimate travel times by considering various factors such as current traffic conditions, weather, time of day, and road characteristics. This capability improves route planning and enhances the efficiency of transportation systems.

Accident Hotspot Identification: By analyzing historical accident data and identifying patterns associated with high-risk areas, GBR can help identify accident hotspots. This information is invaluable for implementing targeted safety measures and improving overall road safety (Zhu et al., 2018).

Handling Large and Diverse Datasets

Traffic data is inherently complex, encompassing a wide range of features that can influence traffic patterns. These features include numerical data (e.g., traffic volume, vehicle speeds), categorical data (e.g., road types, traffic signals), and contextual information (e.g., weather conditions, special events). GBR’s flexibility in handling such diverse data types is a significant advantage in traffic management applications. The ability to process and integrate various data sources ensures that the models are comprehensive and accurate, capturing the multifaceted nature of traffic systems.

Gradient Boosting Regressor stands out as a robust and versatile tool for predictive modeling in traffic management. Its high prediction accuracy, ability to handle missing data, and robustness to overfitting make it an essential technique for developing reliable traffic prediction models. By leveraging GBR, transportation authorities can improve traffic flow, reduce congestion, and enhance overall road safety, ultimately contributing to more efficient and sustainable urban transportation systems.  
  
Data Preprocessing and Feature Engineering

Effective data preprocessing and feature engineering are foundational to building successful predictive models. These steps transform raw data into a clean and structured format, suitable for machine learning algorithms. Proper preprocessing ensures data integrity, reduces biases, and enhances the model's predictive performance. This section provides a detailed overview of the key preprocessing techniques and their significance in developing robust predictive models, particularly focusing on traffic data.

Data Cleaning

Data cleaning is the initial step in preprocessing, aimed at identifying and rectifying errors and inconsistencies in the dataset. Common issues include missing values, outliers, duplicate records, and irrelevant features.

Missing Values: Missing data is a prevalent issue in traffic datasets. Handling missing values appropriately is crucial for maintaining the quality of the data. Several techniques are used to impute missing values:

Mean/Median Imputation: Replacing missing values with the mean or median of the respective feature. Median imputation is particularly useful for skewed data as it is less sensitive to outliers (Batista & Monard, 2003).

Mode Imputation: For categorical data, missing values can be replaced with the most frequent category.

K-Nearest Neighbors (KNN) Imputation: This technique imputes missing values based on the values of the nearest neighbors, considering the similarity of other features.

Outliers: Outliers can significantly affect model performance. Identifying and handling outliers can involve:

Statistical Methods: Techniques such as the Z-score or the IQR method can identify outliers based on statistical properties of the data.

Domain Knowledge: Leveraging domain expertise to recognize and address data points that are not plausible within the context of traffic data.

Duplicate Records: Removing duplicate entries is essential to avoid biased results and ensure the dataset accurately represents the true population.

Feature Scaling

Feature scaling is essential for algorithms sensitive to the magnitude of features, such as Gradient Boosting Regressor (GBR). Scaling ensures that all numerical features contribute equally to the model, preventing features with larger ranges from dominating the model's learning process.

StandardScaler: This method standardizes features by removing the mean and scaling to unit variance. It is particularly effective when the features follow a Gaussian distribution.

Formula: X\_scaled = (X - μ) / σ

Min-Max Scaling: This technique scales features to a fixed range, usually [0, 1].

Formula: X\_scaled = (X - X\_min) / (X\_max - X\_min)

Feature scaling is critical for maintaining consistency across features, ensuring that the model treats each feature with appropriate importance.

Encoding Categorical Features

Categorical features must be converted into a numerical format to be processed by machine learning algorithms. One of the most effective methods for this conversion is one-hot encoding.

One-Hot Encoding: This technique transforms categorical variables into a binary matrix, where each category is represented by a separate binary feature.

Example: A categorical feature with three categories [A, B, C] will be transformed into three binary features [is\_A, is\_B, is\_C].

Label Encoding: Assigns a unique integer to each category. While simpler, it can introduce an ordinal relationship between categories, which is often not desirable.

One-hot encoding prevents the model from assuming any ordinal relationship between categories, enabling it to learn from categorical data effectively.

Feature Engineering

Feature engineering involves creating new features or modifying existing ones to improve model performance. This process leverages domain knowledge and insights derived from data analysis.

Derived Features: Creating new features from existing ones. For instance, in traffic data, features such as 'hour of the day' or 'day of the week' can be derived from timestamp data.

Interaction Features: Combining multiple features to capture interactions that might be significant for the model. For example, the interaction between 'weather conditions' and 'time of day' might be crucial in predicting traffic congestion.

Binning: Converting continuous variables into categorical ones by dividing them into intervals. This can help in capturing non-linear relationships.

Implementing a Unified Preprocessing Pipeline

Combining various preprocessing steps into a unified pipeline ensures a streamlined and reproducible workflow. The scikit-learn library provides powerful tools like ColumnTransformer and Pipeline to facilitate systematic preprocessing.

ColumnTransformer: Allows the application of different preprocessing steps to different subsets of features. For example, numerical features can be scaled, while categorical features can be one-hot encoded within a single transformer.

Example:

python

from sklearn.compose import ColumnTransformer

from sklearn.preprocessing import StandardScaler, OneHotEncoder

preprocessor = ColumnTransformer(

transformers=[

('num', StandardScaler(), numerical\_features),

('cat', OneHotEncoder(), categorical\_features)

])

Pipeline: Enables chaining multiple preprocessing steps and model training into a single, cohesive workflow. This ensures that all steps are executed in the correct order and simplifies the process of model fitting and evaluation.

Example:

python

from sklearn.pipeline import Pipeline

from sklearn.ensemble import GradientBoostingRegressor

model\_pipeline = Pipeline(steps=[

('preprocessor', preprocessor),

('regressor', GradientBoostingRegressor())

])

This approach ensures that preprocessing is consistently applied across training and testing datasets, reducing the risk of data leakage and enhancing model reliability.

Effective data preprocessing and feature engineering are crucial for building robust predictive models. By cleaning the data, handling missing values, scaling numerical features, encoding categorical features, and engineering relevant features, we can significantly improve the performance and reliability of machine learning models. The integration of these steps into a unified pipeline ensures a systematic, reproducible, and efficient workflow, paving the way for the successful application of advanced predictive modeling techniques in traffic management and beyond.  
  
Hyperparameter Tuning and Model Evaluation

Hyperparameter tuning and model evaluation are crucial processes in machine learning model development. These steps ensure the model is optimized for performance and rigorously evaluated to determine its effectiveness. This section explores hyperparameter tuning using GridSearchCV and discusses the significance of various model evaluation metrics, particularly focusing on Mean Squared Error (MSE).

Hyperparameter Tuning

Hyperparameter tuning involves selecting the parameter set maximizing a machine learning model's performance. Unlike model parameters learned during training, hyperparameters are predefined and need separate optimization.

Understanding Hyperparameters: In the context of Gradient Boosting Regressor (GBR), key hyperparameters include:

n\_estimators: The number of boosting stages to run. More stages can increase model complexity and accuracy, but also the risk of overfitting.

learning\_rate: Shrinks the contribution of each tree by this factor. Smaller values require more boosting stages.

max\_depth: Maximum depth of individual regression estimators, limiting the tree's node count to prevent overfitting.

min\_samples\_split: Minimum number of samples required to split an internal node.

min\_samples\_leaf: Minimum number of samples required to be at a leaf node.

GridSearchCV: GridSearchCV systematically searches through a specified hyperparameter grid to find the optimal combination.

Process:

Define Parameter Grid: Specify the range of values for each hyperparameter to be tested.

Cross-Validation: Split the dataset into training and validation sets multiple times, enhancing result reliability.

Evaluation: Train the model for each hyperparameter combination and evaluate performance using a chosen metric, like MSE.

Selection: Identify the hyperparameter set yielding the best performance on the validation sets.

Example:

python

from sklearn.model\_selection import GridSearchCV

from sklearn.ensemble import GradientBoostingRegressor

param\_grid = {

'n\_estimators': [100, 200, 300],

'learning\_rate': [0.01, 0.1, 0.2],

'max\_depth': [3, 4, 5],

'min\_samples\_split': [2, 5, 10],

'min\_samples\_leaf': [1, 2, 4]

}

grid\_search = GridSearchCV(estimator=GradientBoostingRegressor(), param\_grid=param\_grid, cv=5, scoring='neg\_mean\_squared\_error')

grid\_search.fit(X\_train, y\_train)

best\_params = grid\_search.best\_params\_

print("Best hyperparameters:", best\_params)

Benefits:

Optimization: Identifies the most effective hyperparameter combination, enhancing model performance.

Efficiency: Automates tuning, saving time compared to manual methods.

Reliability: Utilizes cross-validation for robust evaluation, reducing overfitting risk.

Model Evaluation

Model evaluation metrics are vital for assessing predictive model accuracy and reliability. They offer quantitative measures of model performance, with MSE being a widely used metric for regression tasks.

Mean Squared Error (MSE): MSE measures the average squared difference between predicted and actual values.

Formula:

MSE = (1/n) \* Σ(yi - ŷi)^2

Interpretation: Lower MSE values indicate better predictive performance, signifying closer predictions to actual values.

Advantages:

Sensitivity: Effectively captures large errors due to squaring, highlighting models consistently making large errors.

Widely Used: Standard metric facilitating comparison across different studies and models.

Other Metrics:

Mean Absolute Error (MAE): Measures the average absolute difference between predicted and actual values.

R-Squared (R²): Represents the proportion of variance in the dependent variable predictable from the independent variables.

Model Evaluation Process:

Train-Test Split: Divide the dataset into training and testing sets to evaluate the model on unseen data.

Cross-Validation: Further split the training set into multiple folds for robust evaluation.

Evaluation Metrics: Calculate metrics like MSE, MAE, and R² on the test set to assess model performance.

Comparison: Compare model performance across different metrics to understand strengths and weaknesses comprehensively.

Example:

python

from sklearn.metrics import mean\_squared\_error, mean\_absolute\_error, r2\_score

y\_pred = grid\_search.best\_estimator\_.predict(X\_test)

mse = mean\_squared\_error(y\_test, y\_pred)

mae = mean\_absolute\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

print("Mean Squared Error:", mse)

print("Mean Absolute Error:", mae)

print("R-Squared:", r2)

Hyperparameter tuning via methods like GridSearchCV is crucial for optimizing machine learning models. Model evaluation metrics such as MSE provide critical insights into model performance, ensuring models are not only finely tuned but also rigorously tested for reliability and accuracy.

Case Studies Demonstrating the Effectiveness of Gradient Boosting Regressor (GBR) in Traffic Prediction

Gradient Boosting Regressor (GBR) has proven to be highly effective in various traffic prediction applications. Several case studies have demonstrated its ability to outperform traditional methods by leveraging its advanced capabilities to handle complex datasets and capture intricate patterns in traffic data. This section provides a detailed examination of these case studies, illustrating the practical applications and benefits of GBR in traffic management.

Case Study 1: Short-Term Traffic Congestion Prediction by Zheng et al. (2013)

In a landmark study by Zheng et al. (2013), GBR was employed to predict short-term traffic congestion levels. The study aimed to enhance traffic management systems by providing accurate and timely predictions of congestion, allowing for better traffic flow and reduced delays.

Objective: To predict short-term traffic congestion levels using real-time traffic data.

Data and Features:

Traffic Volume: Real-time data on the number of vehicles passing through a specific road segment.

Weather Conditions: Information on weather conditions, including temperature, precipitation, and visibility.

Road Incidents: Data on accidents, construction work, and other incidents affecting traffic flow.

Methodology:

Data Collection: Collected real-time traffic data from urban road networks.

Feature Engineering: Engineered features such as average speed, vehicle count, and historical congestion levels.

Model Training: Trained a GBR model using the engineered features to predict congestion levels.

Results:

Performance: The GBR model significantly outperformed traditional methods, such as time-series analysis and linear regression, in predicting short-term traffic congestion.

Accuracy: The model demonstrated higher accuracy and lower prediction errors, as indicated by metrics such as Mean Squared Error (MSE) and Mean Absolute Error (MAE).

Conclusion: The study concluded that GBR is highly effective in capturing the dynamic and nonlinear nature of traffic congestion, providing valuable insights for traffic management systems.

Case Study 2: Travel Time Estimation on Urban Road Networks by Wang et al. (2016)

Another significant study by Wang et al. (2016) applied GBR to estimate travel times on urban road networks. The study focused on improving the accuracy of travel time predictions, which is crucial for route planning and traffic management.

Objective: To estimate travel times on urban road networks using GBR.

Data and Features:

Traffic Volume: Data on the flow of vehicles through different road segments.

Weather Conditions: Information on current weather conditions affecting traffic.

Road Incidents: Records of accidents, road closures, and other disruptions.

Historical Travel Times: Past travel time data for various routes.

Methodology:

Data Collection: Aggregated traffic data from multiple sources, including sensors and GPS devices.

Feature Engineering: Created features such as time of day, day of the week, and seasonal patterns.

Model Training: Developed a GBR model to predict travel times based on the engineered features.

Results:

Performance: The GBR model outperformed traditional travel time estimation methods, such as regression models and heuristic approaches.

Accuracy: Demonstrated superior predictive accuracy with lower MSE and higher correlation between predicted and actual travel times.

Conclusion: The study highlighted GBR's ability to handle diverse data types and complex interactions between features, making it an excellent tool for travel time estimation in urban environments.

Incorporating Real-World Data to Enhance Predictive Power

The effectiveness of GBR in traffic prediction is further enhanced by incorporating real-world data, such as traffic volume, weather conditions, and road incidents. These real-time and historical data sources provide a rich context for the model to learn from, capturing the multifaceted nature of traffic dynamics.

Traffic Volume: Real-time traffic volume data from sensors and cameras provides critical information about the current state of the road network. GBR can utilize this data to make accurate predictions about congestion and travel times.

Weather Conditions: Weather has a significant impact on traffic patterns. By incorporating weather data, GBR can adjust its predictions based on conditions such as rain, snow, and visibility, which affect driving behavior and road safety.

Road Incidents: Data on accidents, construction, and other road incidents help the model account for sudden changes in traffic flow. GBR's ability to handle noisy and varied data ensures that these incidents are effectively integrated into the predictive model.

Flexibility and Resilience of GBR

The flexibility of GBR in handling different types of data and its resilience to noise make it particularly valuable for traffic management applications. Key advantages include:

Handling Mixed Data Types: GBR can process both numerical and categorical data, making it suitable for the diverse features present in traffic datasets.

Robustness to Noise: Traffic data can be noisy and incomplete. GBR's iterative learning process and robust algorithms help it to make accurate predictions even in the presence of data irregularities.

Scalability: GBR can be scaled to handle large datasets, which is essential for city-wide traffic management systems.

Conclusion

In summary, case studies by Zheng et al. (2013) and Wang et al. (2016) have demonstrated the effectiveness of Gradient Boosting Regressor in traffic prediction applications. By incorporating real-world data and leveraging the advanced capabilities of GBR, these studies have shown significant improvements in predictive accuracy and reliability. The flexibility and resilience of GBR in handling complex, noisy datasets make it an invaluable tool in the field of traffic management, paving the way for more efficient and intelligent transportation systems.  
  
References

Batista, G. E., & Monard, M. C. (2003). An analysis of four missing data treatment methods for supervised learning. Applied Artificial Intelligence, 17(5-6), 519-533.

Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. Journal of Machine Learning Research, 13(Feb), 281-305.

Chai, T., & Draxler, R. R. (2014). Root mean square error (RMSE) or mean absolute error (MAE)? Arguments against avoiding RMSE in the literature. Geoscientific Model Development, 7(3), 1247-1250.

Dumitrescu, E., Harb, B., & Toubia, O. (2018). On the aggregation of features. Marketing Science, 37(3), 430-447.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. Annals of Statistics, 29(5), 1189-1232.

Lv, Y., Duan, Y., Kang, W., Li, Z., & Wang, F. Y. (2015). Traffic flow prediction with big data: A deep learning approach. IEEE Transactions on Intelligent Transportation Systems, 16(2), 865-873.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12(Oct), 2825-2830.

Vlahogianni, E. I., Karlaftis, M. G., & Golias, J. C. (2014). Short-term traffic forecasting: Where we are and where we’re going. Transportation Research Part C: Emerging Technologies, 43, 3-19.

Wang, Y., Zheng, B., Yao, Z., & Ni, M. (2016). A data-driven system for long-term and short-term travel time prediction. Journal of Big Data, 3(1), 1-15.

Zhu, D., Xie, K., Yang, L., & Wang, Y. (2018). Predicting traffic congestion using gradient boosting regression tree. In Proceedings of the 2018 IEEE International Conference on Big Data (Big Data), 3910-3914.

Zheng, Z., Liu, H., & Qi, Y. (2013). A new framework for short-term traffic congestion prediction using gradient boosting machine. In Proceedings of the 16th International IEEE Conference on Intelligent Transportation Systems (ITSC), 163-168.