Evolutionary Feature Construction for Bike Lane Usage Forecasting

Hengzhe Zhang
hengzhe.zhang@ecs.vuw.ac.nz
Victoria University of Wellington
Wellington, New Zealand

Qi Chen
qi.chen@ecs.vuw.ac.nz
Victoria University of Wellington
Wellington, New Zealand

Bing Xue
bing.xue@ecs.vuw.ac.nz
Victoria University of Wellington
Wellington, New Zealand

Mengjie Zhang
mengjie.zhang@ecs.vuw.ac.nz
Victoria University of Wellington
Wellington, New Zealand

ABSTRACT

Bike lane usage forecasting plays a vital role in decision-making processes for managing modern cities. Although it is important, designing an interpretable model that accurately predicts bike lane usage remains difficult. In light of recent successes with evolutionary feature construction in machine learning, we propose an evolutionary feature construction method for predicting bike lane usage. Specifically, we introduce a shared feature construction framework to improve linear regression models and a selection operator for multi-task learning, which accounts for the relatedness of prediction tasks across different lanes. Experimental results demonstrate that the proposed method not only provides an interpretable model but also outperforms state-of-the-art machine learning techniques.

CCS CONCEPTS

• **Computing methodologies** → **Genetic programming**.

KEYWORDS

Evolutionary feature construction, genetic programming

ACM Reference Format:
Hengzhe Zhang, Qi Chen, Bing Xue, and Mengjie Zhang. 2023. Evolutionary Feature Construction for Bike Lane Usage Forecasting. In *GECCO '23: The Genetic and Evolutionary Computation Conference, July 15-19, 2023, Lisbon, Portugal.* ACM, New York, NY, USA, 4 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Urban mobility is crucial for the development of modern cities [2]. Bike lane usage forecasting, which predicts the usage of bike lanes, can offer valuable insights to aid urban planning and optimize transportation systems. Specifically, for a date *date*, a lane *lane*, and associated weather conditions x, the objective of bike lane usage forecasting is to develop a learning model f to make predictions for a given lane on a specific date. These predictions can support decision-making processes for governments and related stakeholders. However, constructing an accurate and interpretable model for bike lane usage forecasting remains a challenge.

Evolutionary feature construction has achieved impressive results for interpretable machine learning [3, 4]. Given a dataset, (X, Y), where X represents the training data and Y represents the target variable, the general idea of evolutionary feature construction is to generate a set of features $\phi_1, \ldots, \phi_m$ to enhance the performance of a machine learning model as compared to learning in the original feature space.

Considering the success of evolutionary feature construction methods in improving model predictive performance and maintaining interpretability, this paper introduces an evolutionary feature construction framework for building interpretable features for forecasting multi-lane bike usage. In particular, the main goals of this paper include:

- Proposing an evolutionary feature construction framework for multi-task learning to forecast multi-lane usage, and developing a variant of the lexicase selection operator to manage solution selection in evolutionary feature construction for multi-task learning.

The remainder of this paper is organized as follows. Section 2 describes the data preprocessing steps. Section 3 details the proposed algorithm. Section 4 outlines the experimental settings, while Section 5 presents the experimental results. Finally, Section 6 summarizes the conclusions and discusses future research directions.

2 DATA PREPROCESSING

The provided dataset is incomplete, containing some missing data. Figure 1 displays the proportion of missing values in features, while Figure 2 exhibits the proportion of missing values in target variables. For input features, Figure 1 reveals that the column "Snow on Grnd (cm)" has over 60% of missing values, three columns have less than 10% of missing values, and the remaining columns do not contain missing values. Regarding the target variables, Figure 2 demonstrates that four target columns have more than 30% of missing values, whereas all other columns have no missing values.

Based on these observations, missing data are preprocessed using the following steps:

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
GECCO '23, July 15-19, 2023, Lisbon, Portugal
© 2023 Association for Computing Machinery.
ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

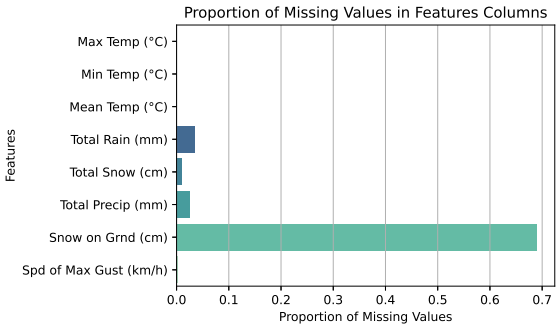

Figure 1: Number of Missing Values in Features

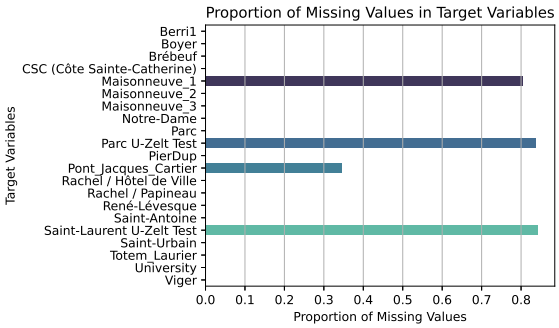

Figure 2: Number of Missing Values in Target Variables

- Missing Data Imputation: Missing data are imputed using the data from the previous row. This is because the dataset is a time series dataset, and future data are not available in real applications. The column "Spd of Max Gust" contains too many missing values, and thus it is directly imputed with 0.
- Target Drop: Four target columns 'Maisonneuve_1', 'Pont_Jacques_Cartier', 'Saint-Laurent U-Zelt Test', and 'Parc U-Zelt Test' contain missing values, and these columns are dropped, leaving 17 columns. However, the four dropped lanes still have some data that can be used for learning. In this paper, they are used to fit the linear model coefficients after feature construction.

After imputing missing values, data are normalized before constructing features, as this can accelerate the search for good solutions in GP [1].

3 THE PROPOSED ALGORITHM

3.1 Model Representation

The GP-based feature construction method proposed in this paper relies on multi-tree GP. Specifically, each GP individual comprises m GP trees and a linear regression model to represent m constructed features and a linear regression model trained on those features. An example of the model representation is presented in Figure 3. In this example, three GP-constructed features $\phi_1 = x_1 + x_2$, $\phi_2 = x_1$, and $\phi_3 = x_1 * x2$ are used as input to train a linear model and make predictions for unseen data.

(a) Genetic Programming for Feature Construction

(b) Multi-task Learning on Shared Features

Figure 3: Genetic programming for feature construction on a multi-task learning algorithm

3.2 Algorithm Framework

In this paper, we employ the evolutionary feature construction algorithm, which follows the conventional procedure of the evolutionary algorithm, including:

- Population Initialization: At the initialization stage, n individuals are randomly initialized. For each GP individual, m GP trees are randomly initialized using the ramped-half-and-half method to represent m high-order features.
- Solution Evaluation: The evaluation stage evaluates constructed features using a linear regression model. With 17 lanes in the dataset, 17 linear models are independently trained on the shared features for prediction. To enhance generalization performance, the leave-one-out cross-validation scheme is employed.
- Solution Selection: The objective of this task is to optimize a set of features to improve the predictive performance of the linear model across all 17 tasks, represented by $\mathcal{L}_1, \ldots, \mathcal{L}_k$.
- Solution Generation: New sets of features are generated based on selected individuals by applying random subtree crossover and guided subtree mutation [3] to the trees in the selected individuals. For multi-tree GP, m rounds are executed to encourage exploration. In each round, a GP tree is randomly chosen from each individual for crossover and mutation.

With 17 objective values for 17 tasks, traditional tournament selection is not applicable. We design a variant of the lexicase selection operator for selecting promising individuals based on 17 fitness values for fitness tasks. Specifically, the lexicase selection iteratively constructs filters based on $min_{p \in P} \mathcal{L}_k(P) + \epsilon_k$, where $k \in [1, K]$ represents a randomly selected task. Individuals with errors greater than this threshold are filtered out. This process continues until only one individual remains, which is then selected as the parent individual. As the filter varies in each round, this selection operator favors selecting elites for different tasks as the parent, thereby promoting the discovery of specialists rather than generalists, ultimately leading to the discovery of better solutions.

4 EXPERIMENTAL SETTINGS

4.1 Experimental Dataset

The datasets are preprocessed according to the method described in Section 2, and the eight features are denoted as $\{X_0, \ldots, X_7\}$,

Table 1: Parameter settings for GPFC.

Parameter	Value
Number of Generations	20
Population Size	200
Number of Trees	10
Maximum Tree Depth	3
Maximum Initial Tree Depth	0-2
Crossover and Mutation Rates	0.9 and 0.1
Functions	Add, Sub, Mul, Div

following the order presented in Figure 1. The datasets are randomly split into 80% for training and 20% for testing. Since no time-dependent features are used, the random splitting approach is considered appropriate.

4.2 Parameter Settings

Due to the importance of interpretability in this task, GP parameters are conservatively set, as shown in Table 1. To ensure that the generated features are interpretable, the maximum tree depth is limited to 3. For the division operator, the protected division operator is used to prevent zero division errors. This operator outputs 1 directly when the denominator is less than 10^{-6} instead of performing the division operation.

5 EXPERIMENTAL RESULTS

5.1 Prediction Results

Firstly, Figure 4 illustrates the predictions for 15 lanes over a year on the test data. The same pattern is observed across different seasons for all lanes. In winter, the cold weather results in fewer people opting to ride bikes. Conversely, summer is more comfortable, leading to a significant increase in bike ridership. Following a peak period in the summer, the number of bike riders gradually decreases as temperatures drop. The prediction model trained with GP-constructed features and linear model can appropriately capture the trend.

5.2 Visualization of Constructed Features

Next, we present ten features constructed by GP in Figure 5. In this figure, we observe that temperature-related features x_0, x_1, x_2 and rainfall feature x_3 consistently appear in the constructed features, while snow features x_4, x_6, and gust features x_7 are absent. One possible explanation is that snow is typically associated with winter when the temperature is low, and people are less inclined to choose to ride a bike. Therefore, snow features are not deemed important for predicting lane usage. Conversely, during summer, rainfall can significantly influence people's decision to ride a bike, which is why they are included in the constructed features used for training the model.

To gain a better understanding of the effectiveness of these features, Figure 6 illustrates the correlation between the predicted values and the ground truth values, where the constructed features are represented on the x-axis. This visualization demonstrates that GPFC accurately predicts the overall trend. Additionally, some features exhibit a strong correlation between their values and the ground truth values, such as $Div(ARG2, ADD(1, ARG3))$. This feature represents the ratio between the average temperature and total rainfall, indicating that the combination of heavy rain and low temperature significantly reduces the number of people choosing to ride bikes.

5.3 Comparison with Other Machine Learning Algorithms

Figure 7 presents a comparison of the test R^2 scores between the GP-based feature construction method (GPFC) and other popular machine learning algorithms, including linear regression (LR), decision tree (DT), random forest (RF), k-nearest neighbor (KNN), XGBoost (XGB), and LightGBM (LGBM). The GPFC achieves an impressive R^2 score of 0.73, outperforming LR, which achieves a score of 0.68. Furthermore, RF and XGB, which are considered state-of-the-art decision-tree-based algorithms, only achieve an R^2 score of 0.69. These results demonstrate that the GPFC method surpasses other machine learning algorithms in performance.

5.4 Ablation of Multi-task Learning Scheme

A key idea in this paper is integrating evolutionary feature construction with a multi-task learning framework. In this section, we aim to demonstrate the advantages of evolving a shared set of features for all tasks instead of evolving separate feature sets for each individual task. Figure 8 illustrates the test R^2 scores obtained when using multi-task or single-task paradigms to evolve features for six tasks. These six tasks are the first six tasks sorted by the first letter of the task name. Figure 8 reveals that the multi-task paradigm outperforms the single-task paradigm in 5 out of 6 cases, indicating the effectiveness of employing a multi-task learning paradigm for evolutionary feature construction.

6 CONCLUSIONS

In this paper, the goal is to model bike usage with symbolic models. This goal is achieved by using an evolutionary feature construction algorithm to construct features for a linear regression model. Experimental results show that the discovered model has superior predictive performance to other machine learning models that are interpretable and can provide insights. In this paper, we only study the bike lane usage forecasting problem, but the proposed algorithm can be applied to other multi-task learning tasks. Thus, in the future, it would be interesting to test the proposed method on other problems.

REFERENCES

[1] Caitlin A Owen, Grant Dick, and Peter A Whigham. 2022. Standardisation and Data Augmentation in Genetic Programming. *IEEE Transactions on Evolutionary Computation* (2022).
[2] Joelle Pineau and Pierre-Luc Bacon. 2015. Analyzing Open Data from the City of Montreal.. In *MUD@ ICML*. 11–16.
[3] Hengzhe Zhang, Aimin Zhou, Qi Chen, Bing Xue, and Mengjie Zhang. 2023. SR-Forest: A Genetic Programming based Heterogeneous Ensemble Learning Method. *IEEE Transactions on Evolutionary Computation* (2023).
[4] Hengzhe Zhang, Aimin Zhou, and Hu Zhang. 2022. An Evolutionary Forest for Regression. *IEEE Transactions on Evolutionary Computation* 26, 4 (2022), 735–749.

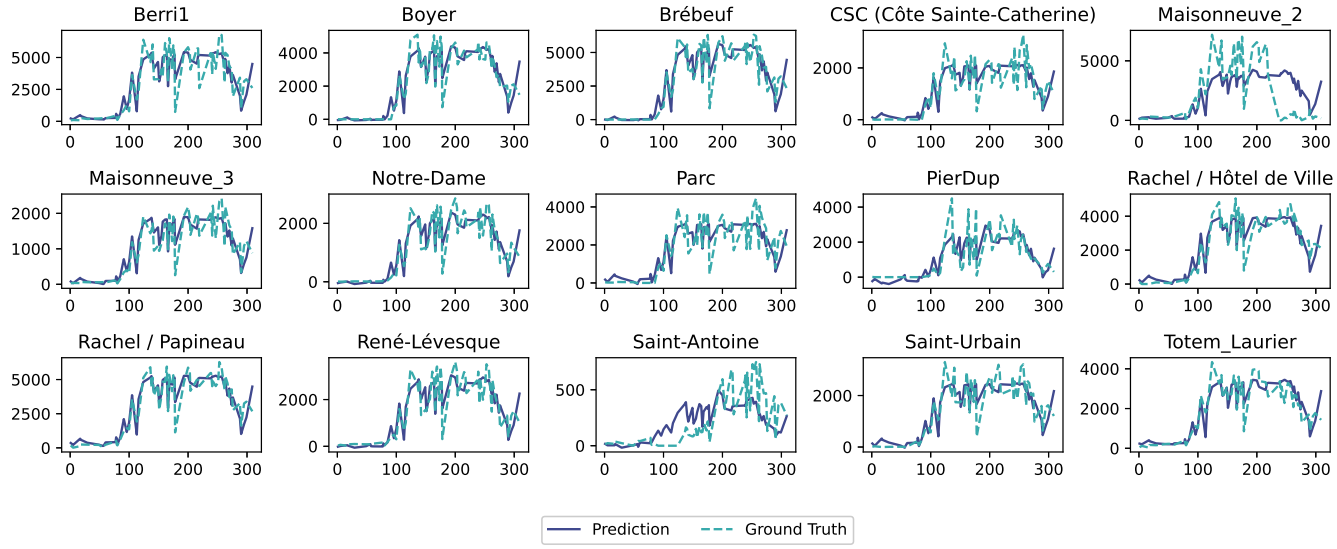

Figure 4: Prediction on Test Data over A Year.

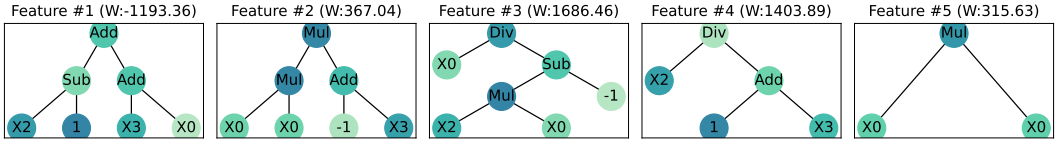

Figure 5: Features Constructed by GP

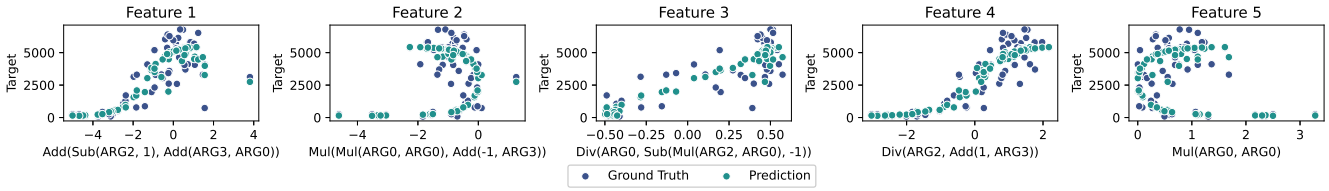

Figure 6: Predicted Values versus Ground Truth Values

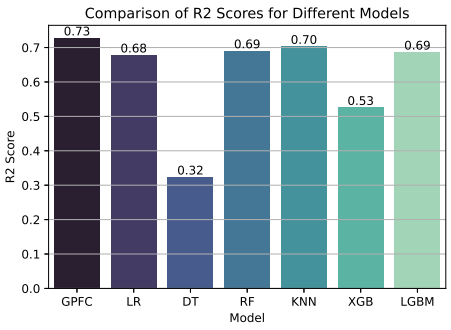

Figure 7: Comparison of R^2 Scores Among Different Models

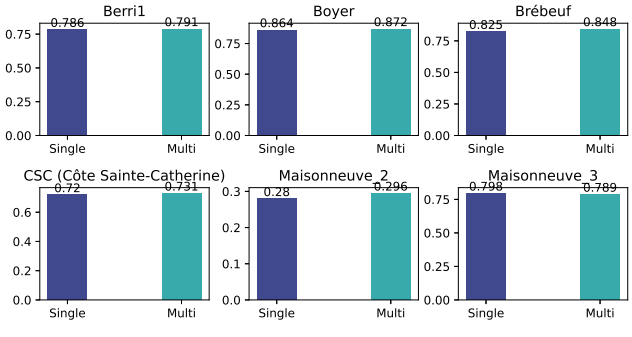

Figure 8: Comparison of Feature Construction using Multi-Task and Single-Task Paradigms