Supplemental Material

In this document, we provide a detailed analysis of the data we used in the paper. Here, we show how to code the machine learning model in R to predict the cycling volumes in the study located in Omaha, Nebraska.

Modeling Cycling Volumes Using Machine Learning

This is a sample code we used for training multiple models to predict the cycling Volumes on streets with signage added in March 2019. We show a description of the data used in the model. Moreover, we show how each model is built and a comparison between different models to extract the best model.

Import Libraries

```
library(dplyr)

library(reshape2)
library(ggplot2)
library(lubridate)\

library(ggpubr)

library(caret)

library(kableExtra)

library (ModelMetrics)

library(caTools)

library(fastDummies)
library (MLmetrics)
```

Data Sample

Table 1: Data Description

Feature	Type	Description	Data Source
Strava Trips	Int	Monthly total Number of cycling trips on streets segment	Strava (Target Variable)
Edge ID	Int	unique street segment ID	Strava

month	int	Chronological months of the year	Strava
One-way street	VARCHAR	Is this a oneway road? “F” means that only driving in direction of the linestring is allowed. “T”	OSM
Wind Speed	float	Average monthly wind speed	Weather
Rain precipitation	float	total monthly rain precipitation	Weather
Snow precipitation	float	Total monthly Snow precipitation	Weather
Average Outside Temperature	float	Average monthly temperature	Weather
fog	int	Number of foggy days in each month	Weather
Heavy fog	int	Number of days that encountered heavy fog in each month	Weather
thunder	int	Number of days that encountered thunderstorms in each month	Weather
Ice pellets	int	Number of days that encountered ice pellets in each month	Weather
hail	int	Number of days that encountered hail in each month	Weather
rime	int	Number of days that encountered rime in each month	Weather
Drifting Snow	int	Number of days that encountered drifting snow in each month	Weather
Heavy rain	int	Number of days that encountered heavy rain in each month	Weather
Snowy Days	int	Number of Snowy days in each month	Weather
Snow	int	Number of days that snow was still precipitated in each month	Weather

fclass	VARCHAR	Street road category (major, minor, no car)	OSM
Season	VARCHAR	Spring, summer, fall, and winter	Weather
Population	int	Number of people living in each zip code area	Census
Population Density	int	Population Density of the zip code area	Census
Number of Houses	int	Number of houses in the zip code area	Census
Living Cost	int	Estimated living cost in each zip code area	Census
Unemployment Rate	float	Unemployment rate in the zip code area	Census
Commute Time	float	Estimated commute time in the zip code area	Census
Median Age	float	Estimated Median age in the zip code area	Census
Gender	int	Estimated number of females/males in the zipcode area	Census

Table 2: Data Sample Summary

Numerical Data					
Feature	**Min**	**1st Quantile**	**Mean**	**3rd Quantile**	**Max**
Trips	0	20	60.74	85	515
Wind Speed	7.518	8.902	10.157	11.090	12.686
Rain precipitation	0.270	1.230	2.993	3.460	9.810

Snow precipitation	0	0	1.639	1.500	27.000
Average Outside Temperature	18.61	37.00	54.46	73.23	80.45
fog	2	7	9.769	12	18
Heavy fog	0	0	1.232	2	7
thunder	0	1	5.338	8	13
Ice pellets	0	0	0.454	0	4
hail	0	0	0.256	0	2
rime	0	0	0.666	1	5
Drifting Snow	0	0	0.442	0	5
Heavy rain	0	2	3.27	4	5
Snowy Days	0	0	1.643	3	13
Snow	0	0	2.9	3	24
Population	1498	131178	17746	23750	36516
Population Density	2751	4144	5333	6312	12912
Number of Houses	500	5832	8095	11024	15662
Living Cost	73.90	89.30	90.22	91.10	93.90
Unemployment Rate	0.0180	0.0370	0.0489	0.0570	0.0810
Commute Time	8.20	16.60	17.14	17.80	21.30

Median Age	19.60	29.40	31.46	33.00	38.40
Categorical Data					
Gender, Male	577	6780	9026	11940	17423
Gender, Female	920	6398	8720	11810	19092
fclass	Minor: 43%, Major: 55%, and no car: 2%				
One-way street	0: 66%, 1: 34%				

Data Splitting

```
trainData <- Data %>%

  filter(datetime < ymd("2018-01-01"))

validationData <- Data %>%

  filter(datetime < ymd("2019-01-01") & datetime > ymd("2017-12-31"))

testData <- Data %>%

  filter(datetime > ymd("2018-12-31"))
```

Model Training and Testing

We train the model using four different well-known machine learning algorithms, Support Vector Machines (SVM), Random Forrest, XGBoost (Linear Booster), and XGBoost (Tree Booster). To extract the best performance, we tune each model separately to minimize mainly the Mean Absolute Error (MAE). The following figures shows the trained model performance for each machine learning algorithm.

XGBoost (Tree Booster)

```
set.seed(100)

tunegrid <- expand.grid(nrounds=c(700),
                        max_depth=c(20),
                        eta=c(0.3),
                        colsample_bytree=c(1),
                        min_child_weight=c(0),
                        subsample=c(0.75),
                        gamma=c(0))
```

```
model_xgb = caret::train(tactcnt ~ .,
                             data=trainData,
                             method="xgbTree",
                             tuneGrid= tunegrid,
                             trControl=trainControl,
                             metric="MAE", verbose=T, nthread =30,tuneLength =6)

# Extract Predictions
model_xgb_pred_18 <- predict(model_xgb, validationData) # 2018 predictions
model_xgb_pred_19 <- predict(model_xgb, testData) # 2019 predictions
```

XGBoost (Linear Booster)

```
set.seed(100)

tunegrid <- expand.grid(nrounds=c(600),
                        eta=c(0.3),
                        lambda=c(1),
                        alpha=c(0))


model_xgblm = caret::train(tactcnt ~ . ,
                             data=trainData,
                             method="xgbLinear",
                             tuneGrid=tunegrid,
                             trControl=trainControl,
                             metric="MAE", verbose=T, nthread =30,
                             tuneLength = 6)


# Extract Predictions
model_xgblm_pred_18 <- predict (model_xgblm, trainData_18) # 2018 predictions
model_xgblm_pred_19 <- predict (model_xgblm, testData_19) # 2019 predictions
```

Random Forrest

```
set.seed(100)

tunegrid_rf <- expand.grid(mtry = c(34))


model_rf = caret::train(tactcnt ~ .,
                             data=trainData,
                             method="rf",
                             tuneGrid=tunegrid_rf,
                             trControl=trainControl,
                             metric="MAE", verbose=T, nthread =30,
                             tuneLength = 6)
```

```
# Extract Predictions
model_rf_pred_18 <- predict (model_rf, trainData_18) # 2018 predictions
model_rf_pred_19 <- predict (model_rf, testData_19) # 2019 predictions
```

SVM

```
set.seed(100)

tunegrid_svm <- expand.grid(sigma = c(0.02464009),
                               C = c(15))


model_svm = caret::train(tactcnt ~ .,
                             data=trainData_pop,
                             method="svmRadial",
                             tuneGrid=tunegrid_svmpop,
                             preProcess = c("center", "scale"),
                             trControl=trainControl,
                             metric="MAE", verbose=T, nthread =30,
                             tuneLength = 6)

# Extract Predictions
model_svm_pred_pop_18 <- predict (model_svm, trainData_18) # 2018 predictions
model_svm_pred_pop_19 <- predict (model_svm, testData_19) # 2019 predictions
```

Model Comparison

the error analysis for the 2018 validation set shows that XGBLinear performs better than other algorithms. Moreover, it provides the least error among all predictive models used in this study.

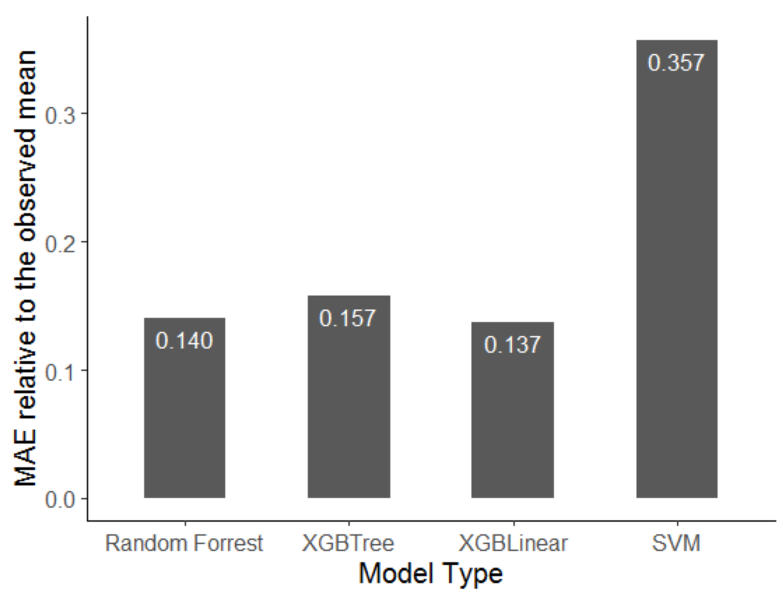

Figure 1: Error analysis of each model tested on the 2018 validation set showing that Xgboost (linear Booster) achieved the lowest error values.

we show every model prediction compared to the monthly accumulative sum of cycling activities in the entire spatial area.

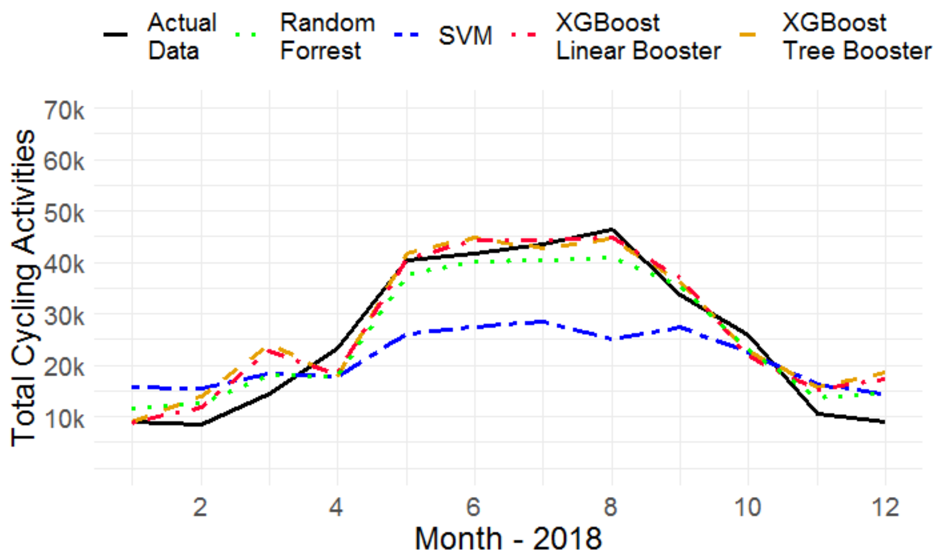

Figure 2: Every Model prediction compared to the monthly accumulative sum of cycling activities in the entire spatial area.

Training 2018 and 2017

To ensure that the model has enough data for training, we use a second model that utilizes the data of 2017 and 2018 as a training set. We use the outcome of models' comparison from Figure 1 and 2, which showed that XGBLinear was the best model for this type of data.

XGBoost (Linear Booster)

```
set.seed(100)

tunegrid_xgblm1718 <- expand.grid(nrounds=c(600),
                        eta=c(0.3),
                        lambda=c(1),
                        alpha=c(1))

model_xgblm1718 = caret::train(tactcnt ~ ., data=trainData1718,

                              method="xgbLinear",
                              tuneGrid=tunegrid_xgblm1718_pop,
                              trControl=trainControl, metric="MAE")
```

```
# Extract Predictions
model_xgblm1718_pop_pred1 <- predict(model_xgblm1718, testData_19)
```

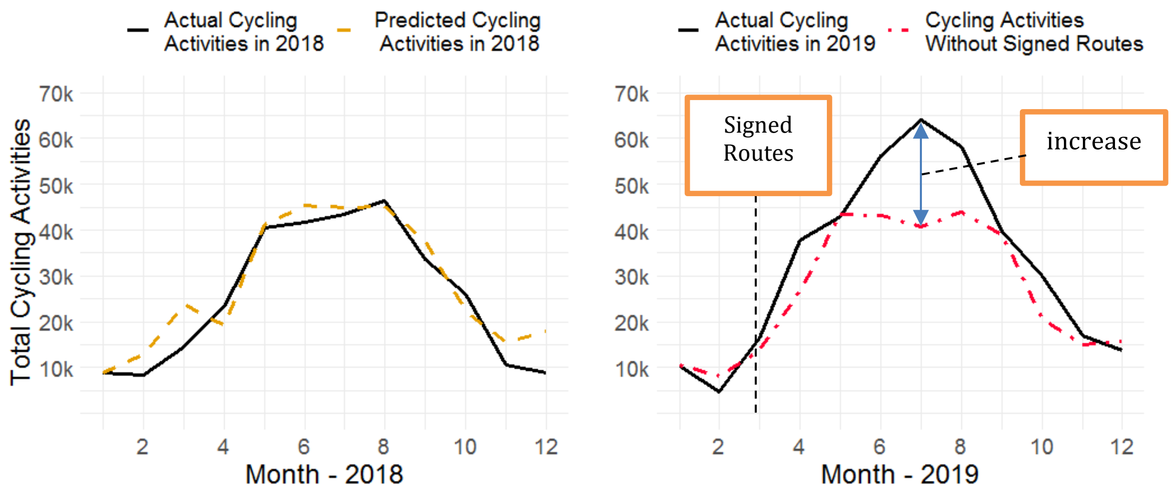

Figure 3: Comparison between 2018 and 2019 cycling volumes predictions for all streets type with added signage in 2019. a) the accurate predictions of 2018 cycling volumes. b) the increase in cycling volumes after installing signage.

Table 3: Tuning Hyperparameters

Tuning Parameter	Function	Value
nrounds	Controlling the maximum number of iterations	600
lambda	to avoid overfitting	1
alpha	Feature selection	1
eta	Learning rate	0.3

In addition to signed routes, other weather and spatial parameters affect cycling. In Figure 4, we show that temperature is the most crucial factor in predicting cycling activities followed by the population. Additionally, the figure indicates that thunderstorms and rain play a role in decreasing cycling. Also, wind speed is more critical in the winter season because of the wind chill factor, which makes it dangerous to bike in Nebraska. Moreover, it is better to distinguish between street types in cycling analysis. For instance, we can see that separating between major, minor roads, one-way streets affect the model's predictions.

```
varimp_xgblm1718_pop<- varImp(model_xgblm1718)

plot (varimp_xgblm1718_pop, top = 8, main="Variable Importance")
```

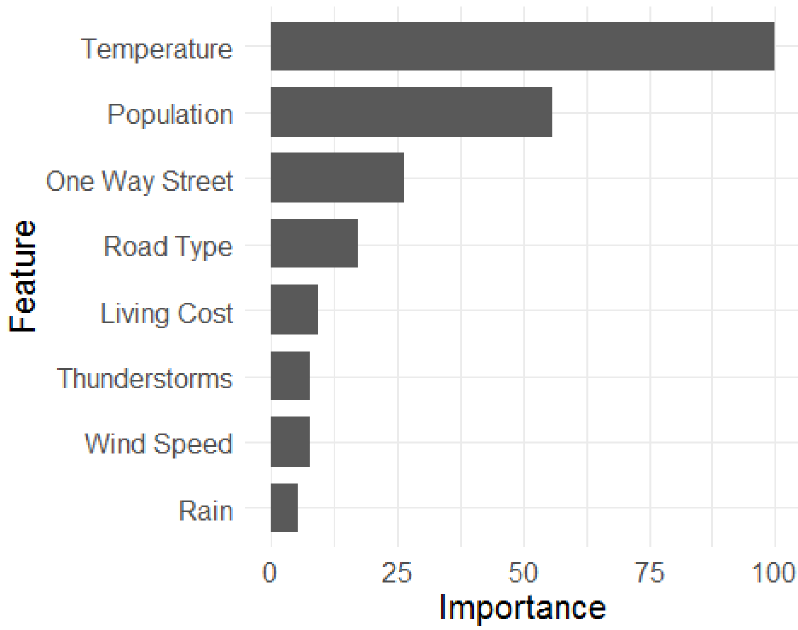

Figure 4: The most essential temporal and spatial factors affecting the prediction of cycling volumes.