Vision Zero Risk Analysis Model

ODN RAM Model Documentation

Carrie Anne Nadeau
Open Data Nation

September 2020

Research Project
Final Report 2020-01

Technical Report Documentation Page

1. Report No. DDOT-RDT-2020-01	2.	3. Recipient's Catalog No.	
4. Title and Subtitle Vision Zero Risk Analysis Model: ODN RAM Model Documentation		5. Report Date September 2020	
		6. Performing Organization Code	
7. Author(s) Carrie Anne Nadeau		8. Performing Organization Report No.	
9. Performing Organization Name and Address Open Data Nation		10.	
		11. Contract or Grant No. DCKA-2015-C-0011 Task 52	
12. Sponsoring Organization Name and Address District Department of Transportation Research, Development, & Technology Transfer Program 55 M Street, SE, 5th Floor Washington, DC 20003		13. Type of Report and Period Covered Final Report April 2019-May 2020	
		14. Sponsoring Agency Code	
15. Supplementary Notes Conducted in cooperation with the U.S. Department of Transportation, Federal Highway Administration.			

16. Abstract

Open Data Nation (ODN) worked with Howard University, on behalf of the District Department of Transportation (DDOT), to construct a predictive model to anticipate which roads are most likely to have traffic crashes in Washington, DC. This work is in support of the city's broader Vision Zero initiative and its deliverables are valuable tools for city planners to prioritize traffic safety engineering, education, and enforcement activities to prevent crashes and save lives.

This document aims to describe these deliverables and provide the technical methodology and requirements to operationalize them. The deliverables outlined here, include an:

- Exposure model, estimating average annual daily traffic
- Crash model, estimating the likelihood of a crash on a road segment

Separately a literature review documented the academic research that guided ODN's approach. In addition, a prototype application was developed to make interactions with data user-friendly.

17. Key Words Crash exposure, logits, machine learning		18. Distribution Statement No restrictions. This document is available from the Research Program upon request.	
19. Security Classification (of this report) Unclassified.	20. Security Classification (of this page) Unclassified.	21. No. of Pages 31	22. Price N/A

Disclaimer

This research was performed in cooperation with the District Department of Transportation (DDOT) and the Federal Highway Administration (FHWA). The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the data presented herein. The contents do not necessarily reflect the official view or policies of the FHWA or DDOT. This report does not constitute a standard, specification, or regulation.

TABLE OF CONTENTS

OVERVIEW

ODN worked with Howard University, on behalf of the District Department of Transportation (DDOT) to construct a predictive model to anticipate which roads are most likely to have traffic crashes in Washington DC. This work is in support of the city's broader Vision Zero initiative and its deliverables are valuable tools for city planners to prioritize traffic safety engineering, education, and enforcement activities to prevent crashes and save lives.

This document aims to describe these deliverables and provide the technical methodology and requirements to operationalize them. The deliverables outlined here, include an:

- Exposure model, estimating average annual daily traffic
- Crash model, estimating the likelihood of a crash on a road segment

Separately a literature review documented the academic research that guided ODN's approach. In addition, a prototype application was developed to make interactions with data user-friendly. A demo of this prototype is shared independently.

EXPOSURE MODEL

Exposure model overview

The purpose of exposure modeling is to estimate the traffic volumes associated with each of the road segments within the administrative boundary of Washington DC. The exposure model developed in this project builds upon traffic counts measured by DDOT and employs machine learning algorithms to predict the traffic volumes where direct measurement is not available.

An estimate of the average annual daily traffic (AADT) for every road segment is the output of the exposure modeling. An exposure model is necessary to estimate the traffic volume on the roadway, which can impact the probability of a crash (more traffic might increase crashes). An AADT value for each roadway may also be used to normalize crash data by the risk of exposure to another vehicle.

Data sources

Data to construct the crash model were sourced from DDOT.

Methodology

Preprocessing

ODN pre-processed data in preparation for exposure model development. This entailed selecting AADT measurements to train the model and excluding missing values.

Select AADT measurements to train model

ODN selected AADT measurements from 2014 to 2017, as these were the most complete. By compiling the data over these four years, we were able to achieve a more comprehensive coverage of different classes of roadways (38% of all roads). This also helped to minimize any year-to-year variation that may have affected the predicted values for each road segment.

Exclude missing values

Variables were first manually screened in order to exclude ones that were not qualified for modeling, such as variables describing data definitions, obsolete variables, redundant predictors, or variables with a high number of missing values. Exploratory analysis subsequently excluded variables that were either null for over.

Create unique identifiers

ODN assigned a unique identifier to each road for clarity and traceability. In this case ODN did not have to create a unique in-house identifier since the data provided by DDOT contained unique Sub-Block keys. As a result ODN used the Sub-Block Keys as a way to reference the smallest road-segment units.

Model selection

ODN aimed to estimate the traffic volume for all road segments within the boundary of Washington DC. Because the target outcome variable, traffic volume (AADT), is numeric and continuous, regression algorithms were deemed appropriate. Table 1 presents the performance scores of the evaluated regression algorithms. ODN trained and testing linear regression, K-nearest neighbor regression, and random forest regression. ODN then looked at those results and selected some algorithms for further evaluation using other techniques. Ultimately, and after applying more fine tuning methods, the random forest model was selected as the best fitting and most appropriate model.

Candidate algorithms were first trained using default hyperparameter settings[1] and tested against a training data set created by splitting the data set into 70% training / 30% testing. Five iterations with different seeds for train/test split were applied in the first round of evaluation. Algorithms that did not perform well were excluded and no longer considered for further evaluation.

Table 1. Performance of the first-round of evaluation of candidate exposure models.

Machine learning models[2]	Performance (root mean square error)[3]	Performance (mean square log error)[4]
k-nearest neighbor regression	5709.47	0.31
random forest regression	4550.58	0.23
linear regression	4932.692	-

Variable selection

Feature selection for the exposure model followed a hybrid method. First, ODN reviewed each variable in isolation, and employed statistical methods such as Lasso Regularization for Feature Selection and conducted a variance check to remove variables with low variance.

Next, variables were considered in relation to one another, as opposed to in isolation. A correlation matrix check was performed to get rid of highly correlated variables since they carry the same amount of 'information' and are duplicative in the value they add to a model.

[1] A model hyperparameter is a configuration that is external to the model and whose value cannot be estimated from data. For example, in a KNN classification, the number of classes K (clusters) is a hyperparameter.

[2] For a high-level overview of the types of machine learning models that exist, we find this resource helpful: https://www.dataquest.io/blog/top-10-machine-learning-algorithms-for-beginners/

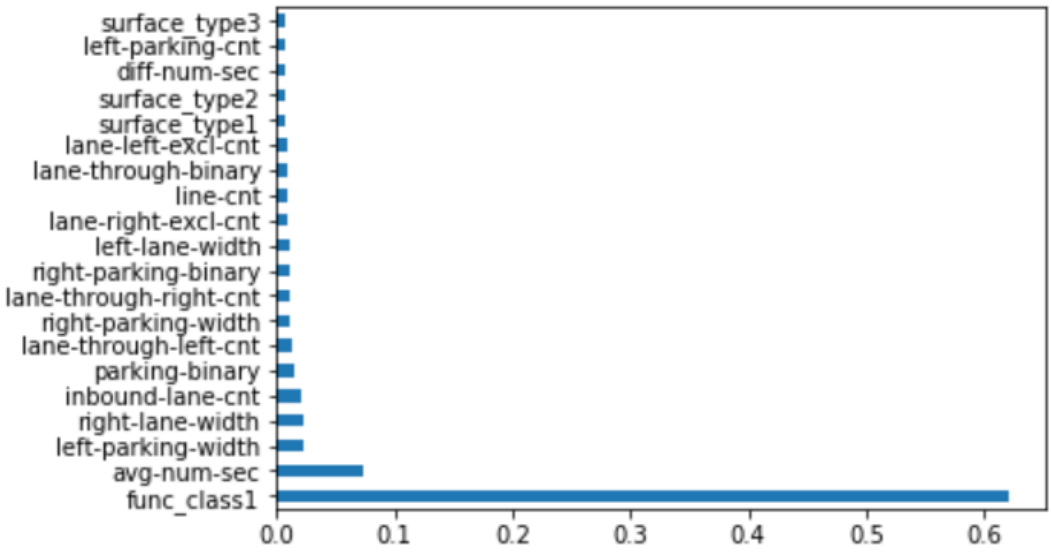
[3] Root Mean Square Error (RMSE) is the standard deviation of the residuals (prediction errors), measuring how spread out residuals are. It may help to diminish the effect of extreme outlier values.

[4] Mean squared logarithmic error (MSLE) can be interpreted as the ratio between the true and predicted values. Because MSLE cares about the relative difference between the real and the predicted value, it treats small differences between small true and predicted values approximately the same as big differences between large true and predicted values.

Lastly, ODN used a tree-based model to further reduce the feature set. Upon examining the variable ranking generated from this analysis, decisions on whether to include variables was further based on investigating the performance of the algorithms initially selected as well as taking into consideration the extent to which the model can be reasonably explained. Following the principle that the model should be sufficiently simple, variables that have marginal contribution to the overall performance were not included.

Graph 1 is a bar chart that displays the top 20 variables resulting from exposure feature selection methods described above. The contributory importance to the models is normalized and ranges from 0 to 1, with higher value meaning the feature is more important.

Graph 1. Feature Selection for Exposure Model



Model evaluation

A k-fold cross validation (with k as 5 and grid search cv[5]) were performed to determine the best hyperparameter(s) for the candidate machine learning algorithms selected from the first step and to evaluate their performances.

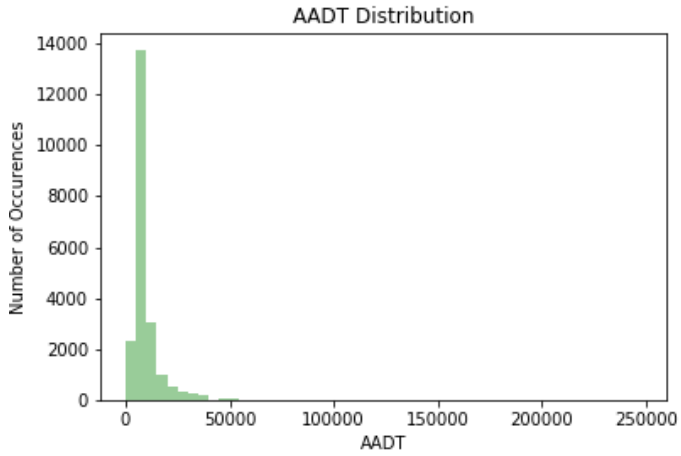
[5] K-fold cross validation is a cross-validation technique that is used to evaluate the model performance by splitting data into K number of sections and each section will be used once as a test set while the rest serve as training set. Grid search cv is the process of tuning hyperparameter to determine the optimal values or the optimal combination of values of a given model.
Please refer to the documentation by the python sklearn library for more detailed explanations and examples: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html

The random forest regression algorithm with the number of estimators (number of trees) as 100 and maximum depth of trees as 6 was selected to be implemented in the final exposure model after evaluating its performance. This selection produces stable results when fed randomly generated training and testing sets. More importantly, compared to the k-nearest-neighbor algorithm, the optimum values for its key hyperparameters showed minimum variation in response to the perturbation of the training dataset and produced lower root mean square error and mean square log error.

Results

In Graph 2, a histogram describes the distribution of AADT in Washington DC compiled using January 2020 predictions. As one might expect, the majority of road segments had a low traffic volume. The overall distribution has a significant right skew.

Graph 2. Map of the distribution of AADT 2020



(Mean = 10581.94, Median = 8010.96, 75th Percentile = 10278.35, 95th Percentile = 24564.39)

https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

CRASH MODEL

Crash model overview

ODN's crash model aims to estimate the probability of at least one crash occurring on a road segment in a one-month timeframe. All streets are scored and then indexed on a scale of 0 to 1. For example, a score of 0.4 would indicate that the probability of a crash on a road in the month is 40%.

Data sources

Data to construct the crash model were sourced from DDOT, DC open data portal and Global Surface Summary of the Day dataset (GSOD) that is produced by the National Oceanic and Atmospheric Administration (NOAA) available through BigQuery on Google Cloud Platform. From the raw data, ODN was able to create 192 variables that contained data that was relevant to modeling (e.g., street name would not qualify) and there was not a significant number of missing values (<20% missing values).

Roadway Information file

DDOT provided ODN a shapefile and CSV of the roadway information. This dataset includes 125 variables covering aspects such as traffic volume, road conditions, roadway characteristics etc.

Annual Crash Reports in Washington DC[6]

Annual crash reports were made publicly available by DDOT via the District's open data portal. The full crash dataset includes features regarding the crash, as well as vehicles and persons involved in the crash. In some cases, latitude and/or longitude geographic identifiers are missing from the records and ODN was unable to locate a crash. In these cases, representing about 10% of all crashes, ODN dropped records.

[6] The annual crash reports are publicly available: http://opendata.dc.gov/datasets/crashes-in-dc

311 Maintenance Requests in Washington DC [7]

The 311 datasets were made publicly available through the DC open data portal in csv format. Data from 311 maintenance requests evaluated and ingested in this model include: potholes on the road, road repair and cleaning, sign/marking modification, tree debris, tree trimming request, street light(s) issues and construction.

Weather Data [8]

Weather data are sourced from the National Oceanic and Atmospheric Administration (NOAA). These data include: temperature, precipitation, wind speed, snow depth, visibility, and other descriptive weather metrics. Data are extracted via operating SQL queries in DataLab on Google Cloud Platform BigQuery and are saved in csv format.

Average Annual Daily Traffic (AADT)

In addition to using the AADT available from DDOT, ODN estimates AADT from the exposure model where observed AADT is not available, detailed earlier in this document.

Methodology

Preprocessing

ODN pre-processed data in preparation for model development. This entailed creating new variables, matching data that were not reported at the same level of geographic specificity, constructing a training dataset, and outputting a model-ready dataset. In the pre-processing step, ODN aimed to create a dataset where each row represents a unique road and each column represents a characteristic of the road.

[7] 311 Data (City service requests):

2014: http://opendata.dc.gov/datasets/city-service-requests-in-2014

2015: http://opendata.dc.gov/datasets/city-service-requests-in-2015

2016: http://opendata.dc.gov/datasets/city-service-requests-in-2016

2017: http://opendata.dc.gov/datasets/city-service-requests-in-2017

2018: http://opendata.dc.gov/datasets/city-service-requests-in-2018

[8] For more information on Google Cloud Platform BigQuery, visit: https://cloud.google.com/bigquery/

Feature engineering

In addition to the data extracted from primary data sources, ODN created a number of variables, listed in the Appendix 1.

The values for variables ingested into the model must be contemporaneous or contemporaneous values must be estimated through prediction. For example, when predicting January 2019, all variables must be available, for this model, through December 2019. Most predictors, such as road network information, are relatively invariant and do not need to be adjusted. However, there is a time lag in reporting 311 values and weather data, so contemporaneous values must be estimated.

To estimate missing 311 and weather data, we tested multiple methodologies, and determined that imputing the values from the same month in the previous year, was the best choice. Across the four years of data, the values of predictors were most consistent year over year, rather than month over month.

All variables were finally encoded into formats that are compatible with the model. Specifically, categorical variables stored as strings or booleans were first encoded as numbers (e.g. “yes”, “no” were converted into numerical values such as 1, 0 etc.), and then one hot encoded.[9] Finally, all the variables were scaled to the range of 0 to 1.

Spatial data to road segment matching

For data that is not reported by road segment, ODN also needed to match data to road segments. For example, crash reports are reported with latitude and longitude coordinates and need to be assigned to a road in order to be considered in the crash model. ODN undertook the following process to match data. First, ODN paired data that fell within the bounds of the road segment plus a 100-foot buffer width. For any remaining crashes that are not paired to a road, the crash is assigned to the segment that is closest.

Training dataset and testing dataset

[9] One hot encoding is a data processing step which converts categorical variables into multiple new variables, each representing one of the levels of the categorical variables. For example, a categorical variable with three levels: low, median, high can be one-hot encoded as three variables: low, medium, and high. A sample with the value of “low” can be expressed as: low: 1, median: 0, high: 0 in such an encoding system.

The crash model was trained with a dataset that covers the time period from January 2014 to November 2018. December 2018 was withheld to use as a testing dataset. The risk scores associated with each road segment for December 2018 were then generated by the trained model.

For any road segment that did not experience a crash in a particular month, ODN created a negative sample, setting the value equal to zero. The final dataset has more than two times the negative samples (roads without crashes) compared to positive samples (roads with crashes). We balanced the data by randomly upsampling the roads with crashes in both training and testing data and kept all the qualified roads with no crashes in the training dataset and testing data to optimize the model, which effectively avoided issues associated with imbalanced classes.[10]

Output dataset

At the conclusion, an output table containing risk scores for each road segment was obtained. In this dataset, each road segment is represented twelve (12) times, one record for each month and including both stable (e.g. road width) and time dependent (e.g. weather) variables.

Model and variable selection

Logistic regression with L1 (LASSO) regularization was selected as the primary modeling framework due to its relatively unbiased calculation of probabilities compared to other machine learning algorithms. In addition, logistic regression can be readily implemented with LASSO which effectively manages feature selection and enhances the interpretability.[11] A detailed description of Logistic regression with L1 (LASSO) regularization is available in Appendix 2.

[10] Imbalanced classes in the training dataset for a classification model may lead to "falsely" accurate results if the model is trained to achieve best accuracy. In an extreme case, for example, if one class is 99 times in sample size compared to the other class, the trained model is likely to predict every sample to be in the majority class and still able to achieve 99% accuracy. While in less extreme situations, imbalanced classes may not necessarily cause such problems, common practices are to manually balance the two classes using statistical techniques such as over-sampling or down-sampling.

[11] The penalty parameter is 1 after balancing the prediction accuracy and number of "active" predictors (i.e. predictors with regression coefficients larger than zero). In the L1 regularization, the penalty parameter indicates the "intensity" of penalty imposed on the regression coefficients. In general, a smaller penalty parameter results in stronger penalty and more coefficients close/equal to zero.

Other classification model frameworks, such as random forest, were considered. However, random forest was excluded from further consideration because of its relatively biased probability prediction and although the Python *sklearn* library provides a class and associated functions and method to calibrate the trained classifiers, it adds to the total structural and time complexity of the crash model.

All of the variables included in the model as predictors are listed in the Appendix 3.

Model evaluation

The dependent variable represents whether there was a crash on this road segment during this month and is reported as a binary classification (e.g., 1-yes, 0-no).

The Brier score, a function that measures the accuracy of probabilistic predictions, was used to quantify the performance of the crash model. It can be applicable to binary classification problems and the common formulation can be written as:

$$Brier\ score = \frac{1}{N}\sum_{n=1}^{N} (P_n - O_n)^2$$

where *P* is the predicted probability (0 to 1), *O* the actual outcome (0 and 1), and N is the total number of samples. Brier score ranges from 0 to 1 with 0 suggesting optimal probability prediction. For example, when the crash model predicted the probability of a crash on a road segment as 100% whereas no crash occurred, the resulted Brier score is 1 (perfectly inaccurate).

In the final fitted model, the Brier score is equal to 0.205. Because this value is closer to 0, this means that the model is producing accurate probabilities for each road's crash predictions.

Feature importance

In ODN's crash model, the relative importance of variables in the model ("feature importance") could have been evaluated using multiple approaches. Ultimately, the mean decrease node impurity in random forest method, explained in detail below, was implemented as the primary

approach for the evaluation of feature importance. The feature importance chart and table, listing the importance score for each variable, is available in Appendix 4.

Mean decrease node impurity

Mean decrease node impurity was calculated during the training process of tree-based machine learning methods, such as random forest. Random forest consists of many decision trees. Every tree is designed to split the dataset into pieces that will end up having similar response values. The measure based on which the optimal condition is chosen is called impurity. Therefore, when training a tree, how much each feature decreases the weighted impurity in a tree (improve the purity of a node) is computed. For a forest, the impurity decrease from each feature can be averaged and the features are ranked according to the measure.

Results

In Table 2, ODN displays a confusion matrix to compare our predicted results for December 2018 to the actual results withheld from the model. We used a cutoff value of 0.5, meaning any predicted probability of a crash on a road that is greater than 0.5 is classified as a road likely to have a crash, whereas values below 0.5 are classified as a road not likely to have a crash.

Table 2. Confusion Matrix[12]

True Negatives: 14179	False Positives: 6086
False Negatives: 6866	True Positives: 13399

Graph 3 is a histogram that contains predicted probability distribution of road segments without crashes (actual zeros) and with crashes (actual ones) in December 2018 (testing set). The model performs well on estimating risk scores on roads without crashes, and also assigns many roads that had crashes before a high risk score.

[12] For reference:

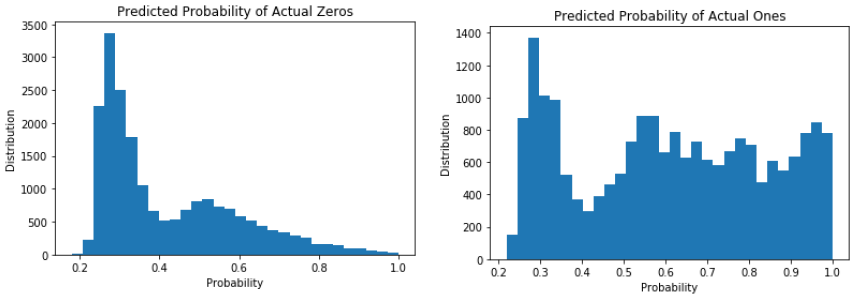
True positives: Road segments ODN predicted would have crashes and did have crashes.

True negatives: Road segments ODN predicted would not have crashes, and they didn't have crashes.

False positives: Road segments ODN predicted would have crashes, but did not actually have crashes.

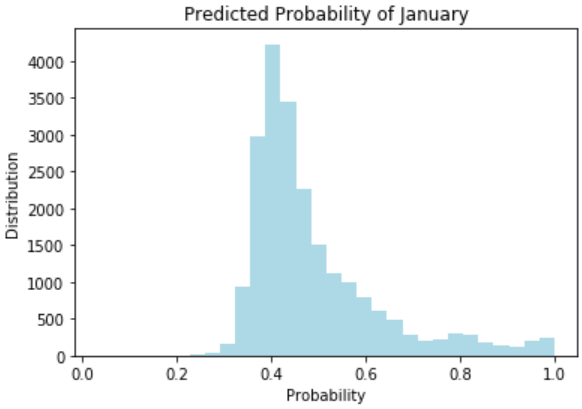
False negatives: Road segments ODN predicted would not have crashes, but actually did have crashes.

Graph 3. Predicted Probability Distribution of Road Segments with/without Crashes



Graph 4 visualizes the distribution of crash estimates in a histogram. As expected, a high proportion of roads have low probabilities of a crash in the next month (January 2020). The distribution is significantly skewed to the right.

Graph 4. Crash estimate distribution for January



In evaluating these results, ODN has determined that the predictions are performing well, with explainable results, and consistent performance.

Output data

The output files from the crash model includes one master dataset with all the variables and records that were compiled and formatted into a “model-ready” structure and one dataset with risk scores calculated for January 2020.

RUNNING MODELS

In order to replicate the results of this analysis, a user will need access to data and Python code. In Appendix 5, an ETL diagram describes the extract, transfer, and load process.

The code is built in a modular way to allow for new data with the same structure to be ingested over time. For reference, the majority of the Python code was run on a 2.3 GHz Intel Core i5, 8 GB 2133 MHz LPDDR3, computer running Mac OS. The weather data was run on Google Cloud Platform DataLab with the setting of a standard 1vCPU, 3.75 GB Memory and 10 GB standard persistent disk. The weather information extraction takes approximately 1 hour for the code to extract, preprocess, clean and save data. The rest of the code takes approximately 10 hours for the code to run end-to-end, including model refitting, hyper parameter exploration, and testing. It takes approximately 2 hours for new data to be ingested and outputs updated on an already fitted model. The process required 5 GB of free space on the hard drive to store all input and output data.

WEB APPLICATION

ODN developed a prototype application layer available. The credentials for this application are shared with users by ODN separately.

APPENDICES

APPENDIX 1. All Generated Features

Variable	Description	Type	Python Data Type
Parking-90degree-binary	Presence (yes or no) of 90 degree parking	Categorical	0/1: Boolean
Parking-Back-In-binary	Presence (yes or no) of back-in parking	Categorical	0/1: Boolean
Parking-Head-In-binary	Presence (yes or no) of head-in parking	Categorical	0/1: Boolean
Parking-Parallel-binary	Presence (yes or no) of parallel parking	Categorical	0/1: Boolean
Parking-UND-binary	Presence (yes or no) of undefined parking	Categorical	0/1: Boolean
Parking-90degree-cnt	Total number of 90 degree parking spaces	Numerical	Integer: int64
Parking-Back-In-cnt	Total number of back-in parking spaces	Numerical	Integer: int64
Parking-Head-In-cnt	Total number of head-in parking spaces	Numerical	Integer: int64
Parking-Parallel-cnt	Total number of parallel parking spaces	Numerical	Integer: int64
Parking-UND-cnt	Total number of undefined parking spaces	Numerical	Integer: int64
Parking-binary	Presence (yes or no) of any parking	Categorical	0/1: Boolean
Parking-cnt	Total number of cross-sections with parking (of any type)	Numerical	Integer: int64
prop-parking-in-num	Proportion of cross-sections that has parking	Numerical	Percentage: float64
left-parking-binary	Presence (yes or no) of parking on the left-most side	Categorical	0/1: Boolean
left-parking-cnt	Total number of cross-sections that have parking on the left-most side	Numerical	Integer: int64
prop-left-parking-in-num	Proportion that has parking on the left-most side	Numerical	Percentage: float64
left-parking-width	Average width of parking on the left-most side	Numerical	Float: float64
right-parking-binary	Presence (yes or no) of parking on the right-most side	Categorical	0/1: Boolean
right-parking-cnt	Total number of cross-sections that have parking on the right-most side	Numerical	Integer: int64
prop-right-parking-in-num	Proportion that has parking on the right-most side	Numerical	Percentage: float64
right-parking-width	Average width of parking on the right-most side	Numerical	Float: float64
parking-width	Average width of cross-sections with parking	Numerical	Float: float64
prop-parking-in-width	Proportion that has parking, by width	Numerical	Percentage: float64
Bike-Bus-Bike-binary	Presence (yes or no) of bike/bus lanes	Categorical	0/1: Boolean
Bike-Conventional-binary	Presence (yes or no) of conventional bike lanes	Categorical	0/1: Boolean
Bike-Green-Paint-binary	Presence (yes or no) of green painted bike lanes	Categorical	0/1: Boolean
Bike-UND-binary	Presence (yes or no) of an undefined bike lane	Categorical	0/1: Boolean
pocket-bike-binary	Presence (yes or no) of a pocket bike lane	Categorical	0/1: Boolean
Bike-Bus-Bike-cnt	Total number of bike/bus lanes	Numerical	Integer: int64

Bike-Conventional-cnt	Total number of conventional bike lanes	Numerical	Integer: int64
Bike-Green-Paint-cnt	Total number of green painted bike lanes	Numerical	Integer: int64
Bike-UND-cnt	Total number of undefined bike lanes	Numerical	Integer: int64
Bike-binary	Presence (yes or no) of any bike lanes	Categorical	0/1: Boolean
bike-cnt	Total number of cross-sections that have bike lanes	Numerical	Integer: int64
prop-bike-in-num	Proportion of cross-sections that has parking	Numerical	Percentage: float64
left-bike-binary	Presence (yes or no) of bike lanes on the left-most side	Categorical	0/1: Boolean
left-bike-cnt	Total number of cross-sections that have bike lanes on the left-most side	Numerical	Integer: int64
prop-left-bike-in-num	Proportion that has bike lanes on left-most side	Numerical	Percentage: float64
left-bike-width	Average width of bike lanes on the left-most side	Numerical	Float: float64
right-bike-binary	Presence (yes or no) of bike lanes on the right-most side	Categorical	0/1: Boolean
right-bike-cnt	Total number of cross-sections that have a bike lane on the right-most side	Numerical	Integer: int64
prop-right-bike-in-num	Proportion that has bike lane on right-most side	Numerical	Percentage: float64
right-bike-width	Average width of bike lanes on the right-most side	Numerical	Float: float64
bike-width	Average width of cross-sections with bike lanes	Numerical	Float: float64
prop-bike-in-width	Proportion that has bike lanes, by width	Numerical	Percentage: float64
Buffer-Bol-Flx-binary	Presence (yes or no) of flexible bollards (flexible posts used to separate traffic)	Categorical	0/1: Boolean
Buffer-Curb-Raise-Cont-binary	Presence (yes or no) of continuously raised curbs on a buffer road type	Categorical	0/1: Boolean
Buffer-Curb-Raise-Land-binary	Presence (yes or no) of non-continuously raised curbs on a buffer road type	Categorical	0/1: Boolean
Buffer-Curb-Raise-Seg-binary	Presence (yes or no) of a raised curb segment on a buffer road type	Categorical	0/1: Boolean
Buffer-Linear-Barrier-binary	Presence (yes or no) of a linear barrier on a buffer road type	Categorical	0/1: Boolean
Buffer-NoCurb-Land-binary	Presence (yes or no) of a 'no curb-land' on a buffer road type	Categorical	0/1: Boolean
Buffer-Paint-White-binary	Presence (yes or no) of white lines on a buffer road type	Categorical	0/1: Boolean
Buffer-Paint-Yellow-binary	Presence (yes or no) of yellow lines on a buffer road type	Categorical	0/1: Boolean
Buffer-UND-binary	Presence (yes or no) of an undefined on a buffer road type	Categorical	0/1: Boolean
Buffer-Bump-Obl-Low-binary	Presence (yes or no) of oblong, low bumps (small solid protuberance found between lanes)	Categorical	0/1: Boolean
raise-curb-binary	Presence (yes or no) of a raised curb	Categorical	0/1: Boolean

raise-curb-cnt	Total number of raised curbs	Numerical	Integer: int64
Buffer-Bol-Flx-cnt	Total number of flexible bollards on a buffer road type	Numerical	Integer: int64
Buffer-Curb-Raise-Cont-cnt	Total number of raised continuous curbs on a buffer road type	Numerical	Integer: int64
Buffer-Curb-Raise-Land-cnt	Total number of raised curbs on a buffer road type	Numerical	Integer: int64
Buffer-Curb-Raise-Seg-cnt	Total number of raised curbs on a segment of a buffer road type	Numerical	Integer: int64
Buffer-Linear-Barrier-cnt	Total number of linear barriers on a buffer road type	Numerical	Integer: int64
Buffer-NoCurb-Land-cnt	Total number of areas without a curb on a buffer road type	Numerical	Integer: int64
Buffer-Paint-White-cnt	Total number of white lines on a buffer road type	Numerical	Integer: int64
Buffer-Paint-Yellow-cnt	Total number of yellow lines on a buffer road type	Numerical	Integer: int64
Buffer-UND-cnt	Total number of undefined on a buffer road type	Numerical	Integer: int64
Buffer-Bump-Obl-Low-cnt	Total number of oblong, low bumps (small solid protuberance found between lanes)	Numerical	Integer: int64
buffer-binary	Presence (yes or no) of any buffer road type	Categorical	0/1: Boolean
buffer-cnt	Total number of cross-sections that have buffer road types	Numerical	Integer: int64
prop-buffer-in-num	Proportion of cross-sections that has buffers	Numerical	Percentage: float64
left-buffer-binary	Presence (yes or no) of a left buffer	Categorical	0/1: Boolean
left-buffer-cnt	Total number of left buffers	Numerical	Integer: int64
prop-left-buffer-in-num	Proportion that has buffers on left-most side	Numerical	Percentage: float64
left-buffer-width	Average width of buffers on the left-most side	Numerical	Float: float64
right-buffer-binary	Presence (yes or no) of a right buffer	Categorical	0/1: Boolean
right-buffer-cnt	Total number of right buffers	Numerical	Integer: int64
prop-right-buffer-in-num	Proportion that has buffers on the right-most side	Numerical	Percentage: float64
right-buffer-width	Average width of buffers on the right-most side	Numerical	Float: float64
buffer-width	Average width of cross-sections with buffer	Numerical	Float: float64
prop-buffer-in-width	Proportion that has buffers, by width	Numerical	Percentage: float64
Barrier-GR-binary	Presence (yes or no) of guardrail barriers	Categorical	0/1: Boolean
Barrier-JB-binary	Presence (yes or no) of jersey barriers	Categorical	0/1: Boolean
Barrier-RB-binary	Presence (yes or no) of rigid bollard barriers	Categorical	0/1: Boolean
Barrier-Bol-Rig-binary	Presence (yes or no) of rigid bollards (stiff, nonflexible barriers)	Categorical	0/1: Boolean
Barrier-GR-cnt	Total number of guardrail barriers	Numerical	Integer: int64
Barrier-JB-cnt	Total number of jersey barriers	Numerical	Integer: int64
Barrier-RB-cnt	Total number of rigid bollards barriers	Numerical	Integer: int64
Barrier-Bol-Rig-cnt	Total number of rigid bollards (stiff, nonflexible barriers)	Numerical	Integer: int64

barrier-binary	Presence (yes or no) of any barrier	Categorical	0/1: Boolean
barrier-cnt	Total number of cross-sections that have barriers	Numerical	Integer: int64
prop-barrier-in-num	Proportion of cross-sections that has barriers	Numerical	Percentage: float64
left-barrier-binary	Presence (yes or no) of barrier on the left-most side	Categorical	0/1: Boolean
left-barrier-cnt	Total number of cross-sections that have barriers on the left-most side	Numerical	Integer: int64
prop-left-barrier-in-num	Proportion that has barriers on left-most side	Numerical	Percentage: float64
left-barrier-width	Average width of barriers on the left-most side	Numerical	Float: float64
right-barrier-binary	Presence (yes or no) of barriers on the right-most side	Categorical	0/1: Boolean
right-barrier-cnt	Total number of cross-sections that have a barrier on the right-most side	Numerical	Integer: int64
prop-right-barrier-in-num	Proportion that has barrier on right-most side	Numerical	Percentage: float64
right-barrier-width	Average width of barriers on the right-most side	Numerical	Float: float64
barrier-width	Average width of cross-sections with barrier	Numerical	Float: float64
prop-barrier-in-width	Proportion that has barriers, by width	Numerical	Percentage: float64
Lane-Bus-binary	Presence (yes or no) of a bus lane	Categorical	0/1: Boolean
Lane-Interch-Aux-binary	Presence (yes or no) of interchange auxiliary (a lane used for both merging onto and off of an interstate)	Categorical	0/1: Boolean
Lane-Left-Excl-binary	Presence (yes or no) of exlusive left turn (left turn only)	Categorical	0/1: Boolean
Lane-Left-Right-Turn-Excl-binary	Presence (yes or no) of left or right turn exclusive (a lane that leads traffic to an intersection with a left turn or right turn only decision)	Categorical	0/1: Boolean
Lane-Right-Excl-binary	Presence (yes or no) of exclusive right turn (right turn only)	Categorical	0/1: Boolean
Lane-Sharrows-binary	Presence (yes or no) of a sharrows	Categorical	0/1: Boolean
Lane-Sharrows-Right-Excl-binary	Presence (yes or no) of sharrows and also a right turn only lane	Categorical	0/1: Boolean
Lane-Shoulder-binary	Presence (yes or no) of a shoulder	Categorical	0/1: Boolean
Lane-Through-binary	Presence (yes or no) of a through lane	Categorical	0/1: Boolean
Lane-Through-Left-binary	Presence (yes or no) of a left through lane	Categorical	0/1: Boolean
Lane-Through-Right-binary	Presence (yes or no) of a right through lane	Categorical	0/1: Boolean
Lane-Through-Right-Left-binary	Presence (yes or no) of left and right turn through (a lane that leads traffic to an intersection with a through, left turn or right turn decision)	Categorical	0/1: Boolean
Lane-UND-binary	Presence (yes or no) of an undefined lane	Categorical	0/1: Boolean
Lane-Center-Turn-binary	Presence (yes or no) of a lane with a center turn	Categorical	0/1: Boolean

Lane-Opp-Left-Bays-binary	Presence (yes or no) of opposing, protected left turn bays (two opposite direction left turns with a curb in between	Categorical	0/1: Boolean
Lane-Reversible-binary	Presence (yes or no) of reversible lanes	Categorical	0/1: Boolean
Lane-Shoulder-Rumble-binary	Presence (yes or no) of shoulder with rumble strips	Categorical	0/1: Boolean
pocket-bus-binary	Presence (yes or no) of a pocket bus lane	Categorical	0/1: Boolean
sharrow-binary	Presence (yes or no) of any sharrows	Categorical	0/1: Boolean
sharrow-cnt	Total number of sharrows	Numerical	Integer: int64
Lane-Bus-cnt	Total number of bus lanes	Numerical	Integer: int64
Lane-Interch-Aux-cnt	Total number of interchange auxiliary	Numerical	Integer: int64
Lane-Left-Excl-cnt	Total number of exclusive left turn (left turn only)	Numerical	Integer: int64
Lane-Left-Right-Turn-Excl-cnt	Total number of left or right turn exclusive (a lane that leads traffic to an intersection with a left turn or right turn only decision)	Numerical	Integer: int64
Lane-Right-Excl-cnt	Total number of exclusive right turn (right turn only)	Numerical	Integer: int64
Lane-Sharrows-cnt	Total number of sharrows	Numerical	Integer: int64
Lane-Sharrows-Right-Excl-cnt	Total number of sharrows with right turns exclusive	Numerical	Integer: int64
Lane-Shoulder-cnt	Total number of shoulders	Numerical	Integer: int64
Lane-Through-cnt	Total number of through lanes	Numerical	Integer: int64
Lane-Through-Left-cnt	Total number of left through lanes	Numerical	Integer: int64
Lane-Through-Right-cnt	Total number of right through lanes	Numerical	Integer: int64
Lane-Through-Right-Left-cnt	Total number of right and left turn through (a lane that leads traffic to an intersection with a through, left turn or right turn decision)	Numerical	Integer: int64
Lane-UND-cnt	Total number of undefined lanes	Numerical	Integer: int64
Lane-Center-Turn-cnt	Total number of lanes with a center turn	Numerical	Integer: int64
Lane-Opp-Left-Bays-cnt	Total number of of opposing, protected left turn bays (two opposite direction left turns with a curb in between)	Numerical	Integer: int64
Lane-Reversible-cnt	Total number of reversible lanes	Numerical	Integer: int64
Lane-Shoulder-Rumble-binary	Total number of shoulders with rumble strips	Numerical	Integer: int64
lane-binary	Presence (yes or no) of any lane road type	Categorical	0/1: Boolean
lane-cnt	Total number of cross-sections that have lane road type	Numerical	Integer: int64
prop-lane-in-num	Proportion of cross-sections that has lane road type	Numerical	Percentage: float64
left-lane-binary	Presence (yes or no) of lane road type on the left-most side	Categorical	0/1: Boolean

left-lane-cnt	Total number of cross-sections that have lane road type on the left-most side	Numerical	Integer: int64
prop-left-lane-in-num	Proportion that has lane road type on left-most side	Numerical	Percentage: float64
left-lane-width	Average width of lane road type on the left-most side	Numerical	Float: float64
right-lane-binary	Presence (yes or no) of lane road type on the right-most side	Categorical	0/1: Boolean
right-lane-cnt	Total number of cross-sections that have a lane road type on the right-most side	Numerical	Integer: int64
prop-right-lane-in-num	Proportion that has lane road type on right-most side	Numerical	Percentage: float64
right-lane-width	Average width of lane road type on the right-most side	Numerical	Float: float64
lane-width	Average width of cross-sections with lane road type	Numerical	Float: float64
prop-lane-in-width	Proportion that has lane road type, by width	Numerical	Percentage: float64
Line-Double-Yellow-binary	Presence (yes or no) of double yellow lines	Categorical	0/1: Boolean
Line-UND-binary	Presence (yes or no) of an undefined line	Categorical	0/1: Boolean
Line-Double-White-binary	Presence (yes or no) of a double white line	Categorical	0/1: Boolean
Line-Single-White-binary	Presence (yes or no) of a single white line	Categorical	0/1: Boolean
Line-Single-Yellow-binary	Total number of single yellow lines	Categorical	0/1: Boolean
Line-Double-Yellow-cnt	Total number of double yellow lines	Numerical	Integer: int64
Line-UND-cnt	Total number of undefined lines	Numerical	Integer: int64
Line-Double-White-cnt	Total number of double white lines	Numerical	Integer: int64
Line-Single-White-cnt	Total number of single white lines	Numerical	Integer: int64
Line-Single-Yellow-cnt	Total number of yellow lines	Numerical	Integer: int64
line-binary	Presence (yes or no) of any line road type	Categorical	0/1: Boolean
line-cnt	Total number of cross-sections that have line road type	Numerical	Integer: int64
prop-line-in-num	Proportion of cross-sections that has line road type	Percentage	Percentage: float64
left-line-binary	Presence (yes or no) of line road type on the left-most side	Categorical	0/1: Boolean
left-line-cnt	Total number of cross-sections that have line road type on the left-most side	Numerical	Integer: int64
prop-left-line-in-num	Proportion that has line road type on left-most side	Percentage	Percentage: float64
left-line-width	Average width of line road type on the left-most side	Numerical	Float: float64
right-line-binary	Presence (yes or no) of line road type on the right-most side	Categorical	0/1: Boolean
right-line-cnt	Total number of cross-sections that have a line road type on the right-most side	Numerical	Integer: int64
prop-right-line-in-num	Proportion that has line road type on right-most side	Numerical	Percentage: float64

right-line-width	Average width of line road type on the right-most side	Numerical	Float: float64
line-width	Average width of cross-sections with line road type	Numerical	Float: float64
prop-line-in-width	Proportion that has line road type, by width	Numerical	Percentage: float64
num-subsubblock	Total number of cross-sections in each sub-block	Numerical	Integer: int64
total-num-of-sec	Total number of road types in each sub-block	Numerical	Integer: int64
outbound-lane-cnt	Total number of outbound lanes	Numerical	Integer: int64
inbound-lane-cnt	Total number of inbound lanes	Numerical	Integer: int64
closest-curb-left	String describing the road type on the left-most side	String	String: object
closest-curb-right	String describing the road type on the right-most side	String	String: object
subblock-width	Average width of sub-blocks	Numerical	Float: float64
FromMeasure	From measure	Numerical	Float: float64
ToMeasure	To measure	Numerical	Float: float64
max-num-sec	Maximum number of road types for each cross-section (proxy for width or number of lanes, road types may be repeated)	Numerical	Integer: int64
min-num-sec	Minimum number of road types for each cross-section (proxy for width or number of lanes, road types may be repeated)	Numerical	Integer: int64
avg-num-sec	Average number of road types for each cross-section (proxy for width or number of lanes, road types may be repeated)	Numerical	Float: float64
diff-num-sec	Difference between maximum and minimum number road types of cross-sections (road type may be repeated)	Numerical	Integer: int64
max-width	Sum each cross-section width, what is the maximum width	Numerical	Float: float64
min-width	Sum each cross-section width, what is the minimum width	Numerical	Float: float64
avg-width	Average width of cross-sections	Numerical	Float: float64
diff-width	Difference between maximum and minimum widths of cross-sections	Numerical	Float: float64
length	Length of the sub-block	Numerical	Float: float64

APPENDIX 2. Using logistic regression with L1 regularization (LASSO)

All the features/variables that passed the initial screen based on data quality was included. A logistic regression with L1 (LASSO) regularization was applied. The selection of the best penalty parameter (C)[13] was based on the evaluation of the logarithmic loss and the outcome of the regularization (i.e. number of features that had zero beta coefficients after regularization operation). It was found that with weaker penalty most of the variables were retained, but the log loss was minimized (C = 1). While stronger penalty will greatly reduce the dimensionality of the model, the performance of the model can be compromised at the same time. Therefore, the current approach is to balance the two aspects and find a C value that does not significantly impact the performance but can reduce the dimensionality of the model to a decent extent. Currently C = 1 is used in the logistic regression model.

Given that we have decided to focus on the accuracy of probability predictions (i.e. how accurate the predicted probabilities are compared to the truth), accuracy is no longer the primary metric that can be used to evaluate the model. Well calibrated classifiers are probabilistic classifiers for which the output of their probability predictions can be directly interpreted as a confidence level. For instance, a well calibrated (binary) classifier should classify the samples such that among the samples to which it gave a predict_proba value close to 0.8, approximately 80% actually belong to the positive class. Therefore, the brier score that was introduced above serves as a good metric in the model.

[13] In the L1 regularization, the penalty parameter indicates the "intensity" of penalty imposed on the regression coefficients. In general, a smaller penalty parameter results in stronger penalty and more coefficients close/equal to zero.

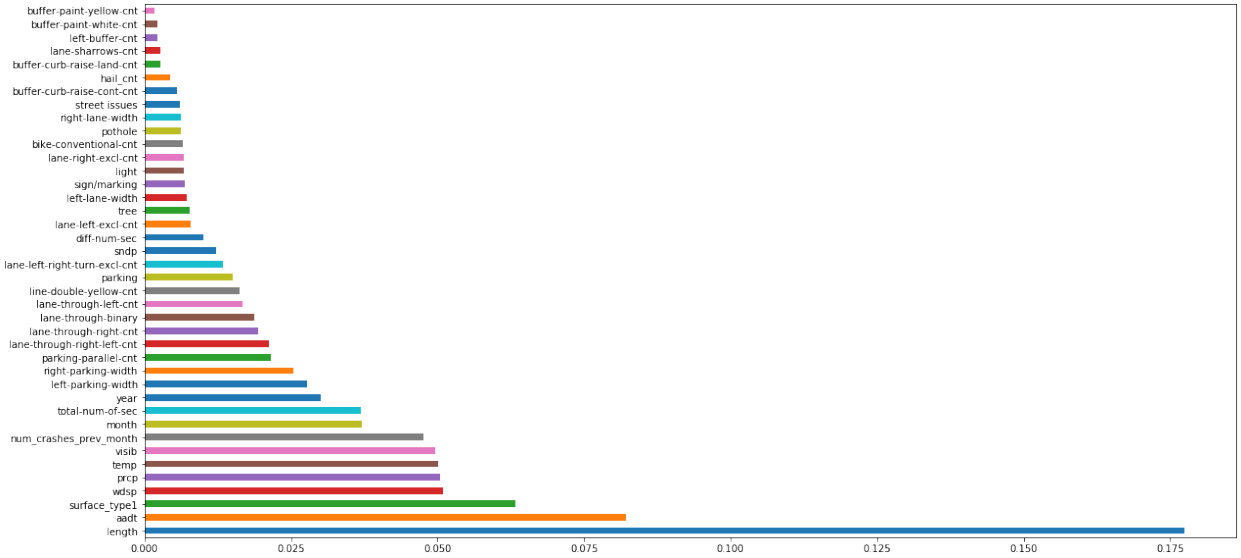
APPENDIX 3. Features included in the crash model as predictors

Variable	Description
aadt	This item identifies the annual average daily traffic. The gaps in the raw dataset are filled by the estimates from the exposure model.
bike-conventional-cnt	This item identifies the total number of conventional bike lanes.
buffer-curb-raise-cont-cnt	This item indicates the presence (yes or no) of continuously raised curbs on a buffer road type.
buffer-curb-raise-land-cnt	This item indicates the presence (yes or no) of non-continuously raised curbs on a buffer road type.
buffer-paint-white-cnt	This item identifies the total number of white lines on a buffer road type.
buffer-paint-yellow-cnt	This item identifies the total number of yellow lines on a buffer road type.
diff-num-sec	This item identifies the difference between maximum and minimum number road types of cross-sections (road type may be repeated.
hail_cnt	This item identifies the number of days that have hail in one month.
lane-left-excl-cnt	This item identifies the total number of exclusive left turn (left turn only).
lane-left-right-turn-excl-cnt	This item identifies the total number of left or right turn exclusive (a lane that leads traffic to an intersection with a left turn or right turn only decision).
lane-right-excl-cnt	This item identifies the total number of exclusive right turn (right turn only).
lane-sharrows-cnt	This item identifies the total number of sharrows.
lane-through-binary	This item indicates the presence (yes or no) of a through lane.
lane-through-left-cnt	This item identifies the total number of left through lanes.
lane-through-right-cnt	This item identifies the total number of right through lanes.

lane-through-right-left-cnt	This item identifies the total number of right and left turn through (a lane that leads traffic to an intersection with a through, left turn or right turn decision).
left-buffer-cnt	This item identifies the total number of left buffers.
left-lane-width	This item identifies the average width of lane road type on the left-most side.
left-parking-width	This item identifies the average width of parking on the left-most side.
length	This item identifies the length of each sub-block.
light	This item identifies the number of light related 311 requests in a month.
line-double-yellow-cnt	This item identifies the total number of double yellow lines.
month	This item indicates the month of the year.
num_crashes_prev_month	This item identifies the number of crash events from the previous month.
parking	This item denotes the number of parking related 311 requests in a month.
parking-parallel-cnt	This item identifies the total number of parallel parking spaces.
pothole	This item denotes the number of 311 reports for potholes on the road.
prcp	This item indicates the total precipitation (rain and/or melted snow) of the day in inches, and average by month.
right-lane-width	This item identifies the average width of lane road type on the right-most side.
right-parking-width	This item identifies the average width of parking on the right-most side.
sign/marking	This item denotes the number of 311 reports for signs or markings.
sndp	This item identifies the average snow depth for the month in inches.
street issues	This item identifies the number of 311 requests for street issues, such as cleaning or repair.
surface_type1	This item indicates the major surface type on each road segment.

temp	This item indicates the mean temperature for the month in degrees Fahrenheit.
total-num-of-sec	This item indicates the total number of road types in each sub-block.
tree	This item identifies the number of 311 requests for tree related services, such as trimming.
visib	This item indicates the mean visibility for the month in miles.
wdsp	This item indicates the mean wind speed for the month in knots.
year	This item identifies the year.

APPENDIX 4. Feature importance chart and list



Score	Variable	Description
0.177383	length	This item identifies the length of each sub-block.
0.082182	aadt	This item identifies the annual average daily traffic. The gaps in the raw dataset are filled by the estimates from the exposure model.
0.063331	surface_type1	This item indicates the major surface type on each road segment.
0.050855	wdsp	This item indicates the mean wind speed for the month in knots.
0.050489	prcp	This item indicates the total precipitation (rain and/or melted snow) of the day in inches, and average by month.
0.050068	temp	This item indicates the mean temperature for the month in degrees Fahrenheit.
0.049628	visib	This item indicates the mean visibility for the month in miles.
0.047662	num_crahses_prev_month	This item identifies the number of crash events from the previous month.
0.037027	month	This item indicates the month of the year.
0.036901	total-num-of-sec	This item indicates the total number of road types in each sub-block.
0.030089	year	This item identifies the year.
0.027663	left-parking-width	This item identifies the average width of parking on the left-most side.
0.025424	right-parking-width	This item identifies the average width of parking on the right-most side.
0.021517	parking-parallel-cnt	This item identifies the total number of parallel parking spaces.
0.021235	lane-through-right-left-cnt	This item identifies the total number of right and left turn through (a lane that leads traffic to an intersection with a through, left turn or right turn decision).

0.019423	lane-through-right-cnt	This item identifies the total number of right through lanes.
0.018643	lane-through-binary	This item indicates the presence (yes or no) of a through lane.
0.016652	lane-through-left-cnt	This item identifies the total number of left through lanes.
0.016215	line-double-yellow-cnt	This item identifies the total number of double yellow lines.
0.015097	parking	This item denotes the number of parking related 311 requests in a month.
0.013376	lane-left-right-turn-excl-cnt	This item identifies the total number of left or right turn exclusive (a lane that leads traffic to an intersection with a left turn or right turn only decision).
0.012253	sndp	This item identifies the average snow depth for the month in inches.
0.010010	diff-num-sec	This item identifies the difference between maximum and minimum number road types of cross-sections (road type may be repeated.
0.007843	lane-left-excl-cnt	This item identifies the total number of exclusive left turn (left turn only).
0.007690	tree	This item identifies the number of 311 requests for tree related services, such as trimming.
0.007218	left-lane-width	This item identifies the average width of lane road type on the left-most side.
0.006950	sign/marking	This item denotes the number of 311 reports for signs or markings.
0.006751	light	This item identifies the number of light related 311 requests in a month.
0.006632	lane-right-excl-cnt	This item identifies the total number of exclusive right turn (right turn only).
0.006602	bike-conventional-cnt	This item identifies the total number of conventional bike lanes.
0.006204	pothole	This item denotes the number of 311 reports for potholes on the road.
0.006175	right-lane-width	This item identifies the average width of lane road type on the right-most side.
0.006085	street issues	This item identifies the number of 311 requests for street issues, such as cleaning or repair.
0.005540	buffer-curb-raise-cont-cnt	This item indicates the presence (yes or no) of continuously raised curbs on a buffer road type.
0.004319	hail_cnt	This item identifies the number of days that have hail in one month.
0.002773	buffer-curb-raise-land-cnt	This item indicates whether the highway operates as a one or two-way facility during peak hours of operation.
0.002711	lane-sharrows-cnt	This item identifies the total number of sharrows.
0.002263	left-buffer-cnt	This item identifies the total number of left buffers.
0.002201	buffer-paint-white-cnt	This item identifies the total number of white lines on a buffer road type.
0.001691	buffer-paint-yellow-cnt	This item identifies the total number of yellow lines on a buffer road type.

APPENDIX 5. ETL diagram