COMP 542 Machine Learning

**Project Final Report**

Due date: by May 9 (Monday), 2022

(Cambria Font, 12 size, Single line and Single column)

**Title:**

Team Member’s Names: Hui Du, Matthew Choi

1. **Introduction** (At least 1 & ½ pages)

1.1 Problems

The focus of this project was to identify and perform qualitative analysis on the relationship between a set of features and the frequency of traffic accidents in December 2019 in North America in order to potentially assist in predicting the severity of an accident by the environment in which it occurred . A dataset restricted to December 2019 was picked due to the time restrictions of using large datasets on our machines. The project used Gradient Boosted Decision Trees to analyze the importance of various features of the data set, such as weather quality, time of accident, length (in miles) of road affected by accident, visibility as provided by the weather quality, etc. This analysis could then be used potentially to predict severity of accidents by environment data and prompt the appropriate response or preparation. For example, if our findings showed higher frequencies during typical daily commuting intervals, this would support a hypothesis that higher traffic would lead to higher amounts of traffic accidents. This would then, in combination with analyses of different sampling intervals, be useful in assisting emergency services in preparing for a higher load of accidents during a specific time frame.

There are, of course, certain results that can be expected even before analyzing the data. Due to the nature of traffic, we can assume that there will be higher frequency of traffic accidents in specific clusters in (lat, long) coordinates, given that high population locations will probably be correlated with a higher accident frequency rate. Based on this reasoning, we can also assume that the time at which an accident occurs will also have a higher importance for predicting accident frequency as we see higher traffic population during commute times. Thus it is important to note that while our project focused on finding the importance of features to assist in future prediction of accident frequency, there may be other important external influences that would have higher correlation with the feature importance than accident frequency.

1.2 Related Prior Works

1.2.1 Prior Works

Previous studies and analyses done on this dataset have generally looked at the relationship between accident frequency and specific features, or just reported the members of each feature with the highest accident frequency. One study [1.1] looked at the correlation between accident severity and the other features of the dataset, then broadened their search to include correlation between all features.

This study also used ExtraTreesRegressor and cross-checked using XGBoost to identify the feature importance similar to our project.

An additional study [1.2] used LogisticRegression to balance their dataset and then trained a random forest model to predict severity of accidents based on their features rather than looking at frequency of accidents based on features.

Overall, studies analyzing this specific dataset focused on specifically quantifying relationships between features and severity of accidents, using feature relationship with accident severity to train a prediction model, or just reporting general sentiments based on findings within the data.

1.2.2 Comparison to Implemented work

While the goal of the more popular studies on this dataset were focused on comparing both the broader correlations between feature importance and severity, as well as looking at how each available data feature indicated severity of an accident, our project instead first looked at each feature in relation to accident frequency. We first looked at each feature in silo, analyzing the relationship between the individual feature and accident frequency as described by that feature. Rather than focusing on general sentiment or focusing on how all features affect severity of accidents, we combined both approaches, analyzing individual features and removing features that would create excessive tilt towards specific conclusions to reach a result that would be more appropriate towards determining severity of an accident.

1.2.3 Approach and Result

Initially, we looked at the relationship between relevant features and accident frequency to better understand and draw hypotheses about the correlation between given features of the dataset and accident frequency. We removed irrelevant data features or features that can be easily explained by external reasoning as part of the preprocessing. Post-processing, we trained the model and ran the test data(30%) through the model to retrieve our final feature importance graph.

1. **Background** (At least 1 page)

To understand the Gradient Boosted Decision Tree (GBDT) model and subsequently the XGBoost method, it is first important to cover the differences between random forest and GBDT. First and foremost, while random forest (RF) and GBDT both use the same weak learner (decision trees), the usage of the sum of these weak learners differs between the two ensemble learning methods.

Random forest method uses a process called bagging, in which individual decision trees are fit to subsamples and used in parallel with other individual trees. GBDT uses the boosting process, in which weak learners (decision trees) are fitted in series, with each output of a tree becoming the input of the subsequent tree to create a strong learner (the entire series). Whereas in the random forest model, each tree is fit to a subsample of the entire dataset, GBDT trees work off of the sequential results of the previous trees, working to remove error from the end result. Thus our project used the XGBoost model due to its flexibility of loss functions and ability to converge towards a more accurate result than that of the RF model, which either takes a majority vote from trees within the forest, or a mean value of all predictions given by trees. GBDT models are also much more sensitive to overfitting and outliers, requiring but also allowing for much more sensitive fine-tuning of parameters.

1. **Data and Model** (At least 2 pages)
   1. Dataset

In this paper, the dataset comes from kaggle US Traffic[3.1]. There’s 51 features and 2.9 million data in this dataset. Considering our problem of hardware, in this paper, only the year 2019 will be used for visualization and prediction (953630).

Some features are useless for the prediction. As a result, there’s 26 total features[3.2] used for training after preprocessing in the final.

* 1. Learning Models

3.2.1 Boosting Tree[3.3]

Boosting trees are boosting methods that use classification trees and regression trees as the basic classifiers. Boosting tree is considered to be one of the best performance methods in statistical learning, and the Boosting method actually uses essentially an additive model (linear combination of basis functions) with a forward stagewise algorithm. Boosting methods that use decision trees as basis functions are called Boosting trees. For classification problems the decision tree is a binomial classification tree and for regression problems the decision tree is a binomial regression tree.

Boosting tree can be expressed by the additive model[3.4] of decision tree:

Where means decision tree; means decision parameter; means the number of tree.

Boosting tree using forward stagewise algorithm[3.5]:

initial state

, when step = m

For each step, the :

Since the linear combination of trees can fit the training data well, even if the relationship between the inputs and outputs in the data is complex, boosting trees is a highly useful learning algorithm.

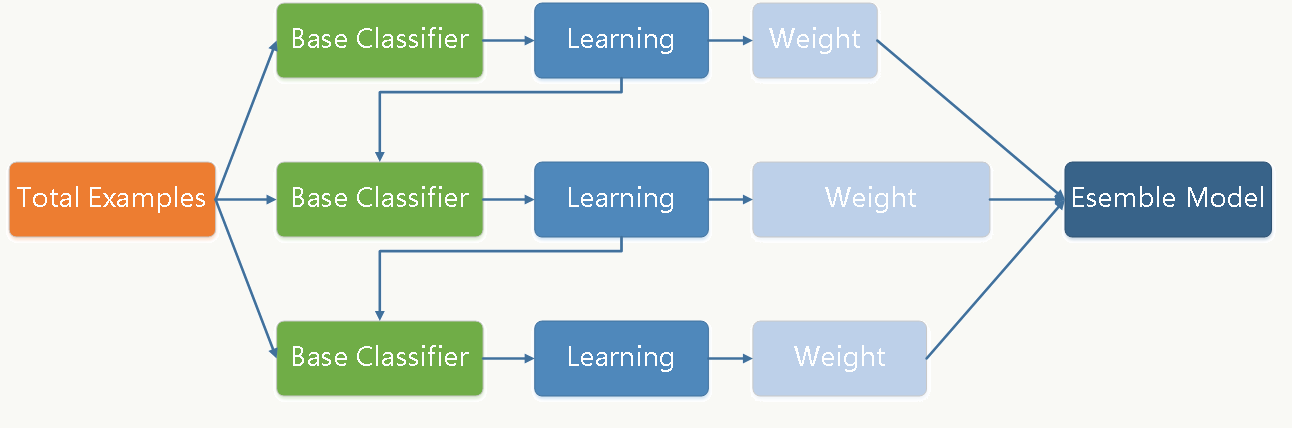
Boosting tree for different problems are discussed below, with the main difference being the different loss functions used. This includes regression problems with squared error loss functions, classification problems with exponential loss functions, and general decision problems with general loss functions.

3.2.2 GBDT

The boosting tree utilizes an additive model with a forward stagewise algorithm to implement the optimization process of learning. When the loss function is squared error loss function and exponential loss function, each step of optimization is simple. However, for general loss functions, it is often not so easy to optimize each step.

To address this problem, Freidman proposed the gradient boosting algorithm[3.6]. Gradient Boosting is a large class of algorithms in Boosting, which borrows its idea from gradient descent, and its basic principle is to train the newly added weak classifiers based on the negative gradient information of the current model loss function, and then the trained The weak classifiers are then combined into the existing model in an accumulative form.

The Gradient Boosting algorithm that uses decision trees as weak classifiers is called GBDT[3.7], sometimes called MART (Multiple Additive Regression Tree), and the decision tree used in GBDT is usually CART[3.8].



In gradient descent[3.9], for the optimal , which obtained the initial value by T times iterations.

Assume , thus for

where denotes the first order derivative of the Taylor expansion[3.10] of at .

The boosting step if same in a function.

In each iteration, the Gradient Boosting algorithm first calculates the negative gradient of the current model on all samples, and then trains a new weak classifier with this value as the target to fit and calculate the weights of this weak classifier, and finally realizes the update of the model.

In this paper, due to the page length, each step of GBDT will not be listed.

3.2.3 Xgboosting:

XGBoost and GBDT are both boosting methods, except for some differences in engineering implementation and problem solving, the biggest difference is the definition of the objective function.

The objective function[3.11] can be define:

to calculate the objective, using Taylor expansion:

where

is square loss function, for example

is for regularization, including L1 and L2

is one of the trees.

Xgboosting not only supports decision trees, but also can be implemented in the linear model. In this paper, only the tree model will be discussed.

For the tree :

Where is one dimension vector, meaning leaf weight, is the structure of the tree.

As a result, the complexity of tree can be defined:

where is the number of leaves, is L2 norm of leaf scores.

For the final tree model, the objective becomes:

Where , put all samples belonging to the first leaf node into a sample set of leaf nodes

is the first order partial derivatives sum of node leaf and is the second order partial derivatives sum of node leaf

In practical training, when building the -th trees, a very critical problem is how to find the optimal points of the leaf nodes, and XGBoost supports two methods for splitting nodes[3.12] - greedy algorithm and approximate algorithm. This paper will not discuss these algorithms.

As a result, this several advantages for this method:

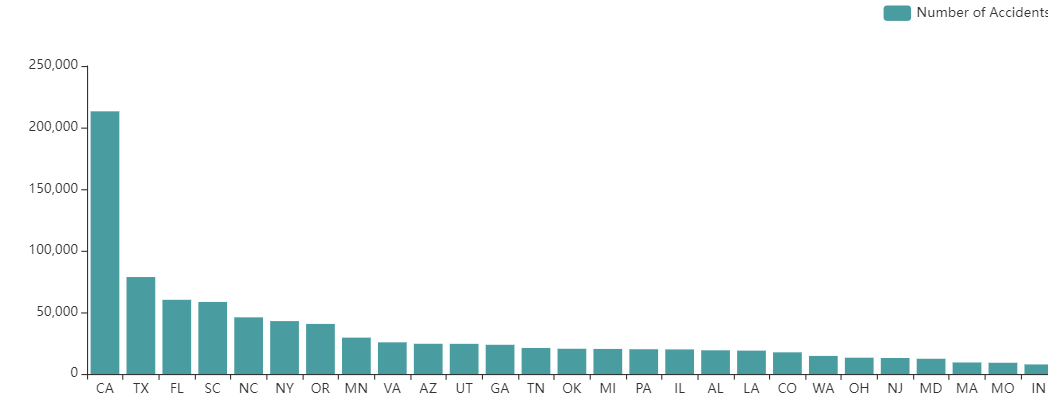
* High accuracy: Compared to GBDT, the XGBoost uses second-order Taylor expansion that is performed on the loss function, instead of the first-order. It highly improves the accuracy of loss function.
* Flexibility: Compared to GBDT, XGBoost not only supports the CART classifier, but also can be used in linear model. XGBoost using a linear classifier is equivalent to logistic regression (classification problem) or linear regression (regression problem) with L1 and L2 regularization terms. In addition, the XGBoost tool supports custom loss functions, as long as the functions support first- and second-order derivatives.
* Regularization: Compared to GBDT, XGBoost adds a regular term to the objective function to control the complexity of the model. The regular term reduces the variance of the model, which makes the learned model simpler and helps prevent overfitting.
* Shrinkage: Equivalent to the learning rate, XGBoost multiplies the weights of the leaf nodes by this factor after one iteration, mainly to weaken the influence of each tree and to allow more learning space later.
* Column Sampling: Compared to GBDT, XGBoost borrows from random forests and supports column sampling, which not only reduces overfitting, but also reduces computation.
* Missing Value: For samples with missing values of features, the sparse perception algorithm used by XGBoost can automatically learn its split direction.
* Concurrency: Compared to decision tree, instead of taking a long time to sort the value, XGBoost pre-sorts the data before training, and then saves it as a block structure, which is repeatedly used in later iterations to greatly reduce the computational effort. This block structure also makes parallelism possible. When splitting nodes, the gain of each feature is calculated, and the feature with the largest gain is finally selected for splitting, then the gain of each feature can be calculated in multiple threads.
* Approximation: When tree nodes are split, we need to calculate the gain corresponding to each split point for each feature, i.e., enumerate all possible split points using the greedy method. The greedy algorithm becomes inefficient when the data cannot be loaded into memory at once or in a distributed situation, so XGBoost also proposes a parallelizable approximation algorithm for efficiently generating candidate split points.

**Experiments and Analysis of Results** (At least 2 pages)

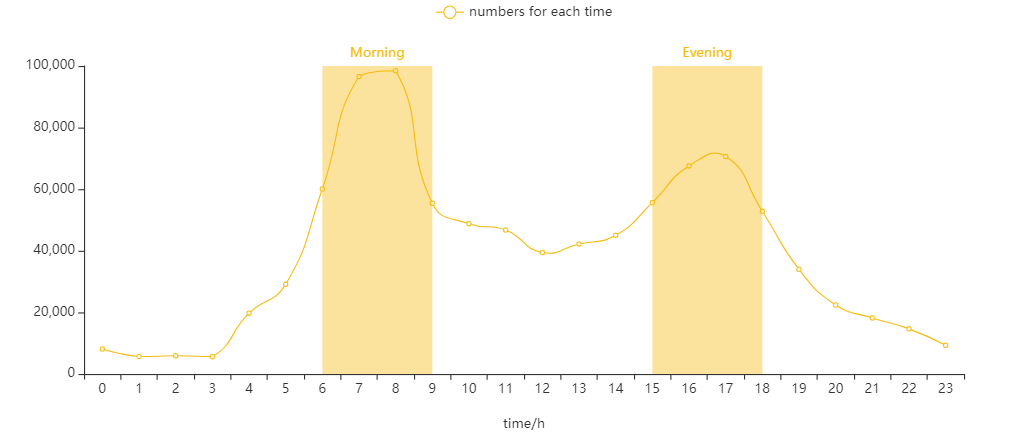
3.3 Describe in detail how you implemented and what results you have obtained from this project.

Based on the dataset itself, the procedure for analysis includes: feature selection, visualization, preprocessing, and modeling. Due to the time and result, in this paper, the result is good enough, the parameter adjustment is not included (if more times left then will be ).

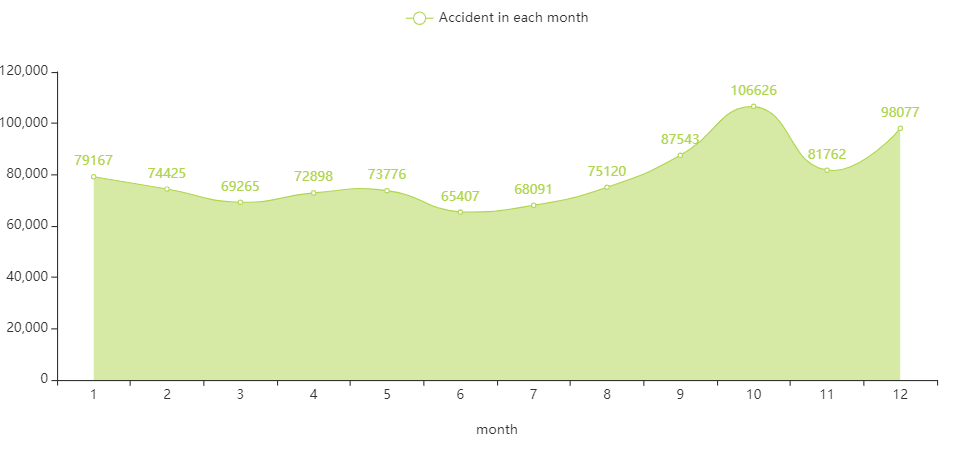
1. More than a half of accidents happened in CA, TX, FL, SC, NC. It is highly possible that more accidents occured in more developed areas.



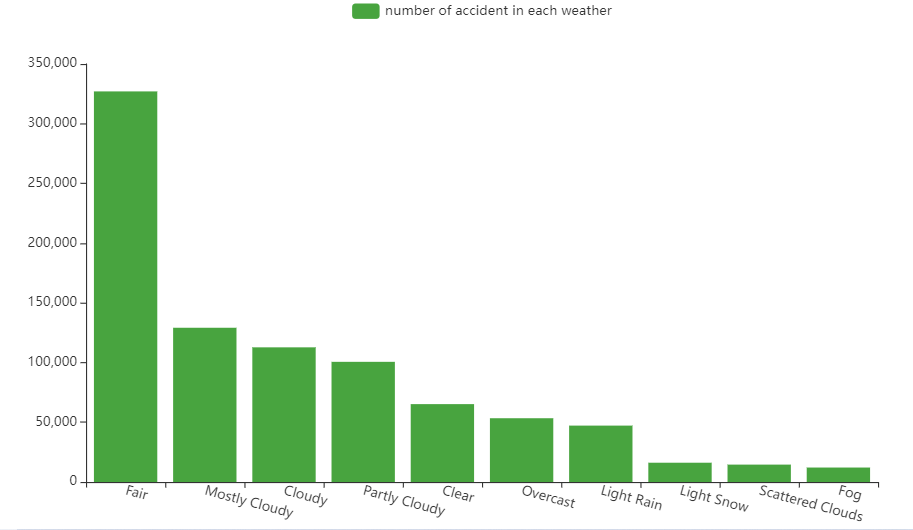
1. It is clear that in the morning and evening, the amount of accidents are very high. Thus, the dangerous time is the time when anyone go to work or back home.



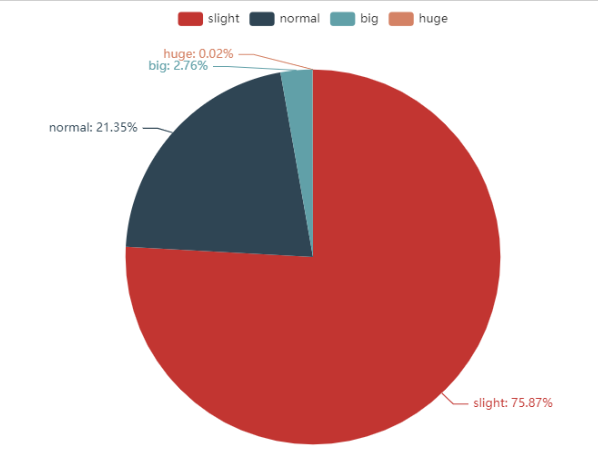
1. The number of accidents in the second half of the year is significantly higher than that in the first half of the year, perhaps because there are more holidays and less workload in the second half of the year.



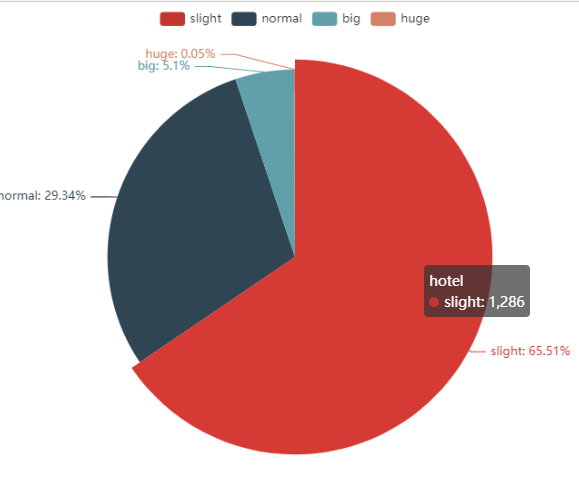
1. Accidents mostly happened in fair weather, but the ratio of weather is different; for example, there's 365 days in a year and 300 days are ‘Fair’ day, so more accidents happened in ‘Fair’ day.



1. To test the assumption, plot the ratio of different weather: in this case, the Fair and Snow presented. It is the same as the assumption before, but the relationship between the “bad weather” and more ”huge” accidents is not always the case. For example, the Clear and light snow is not like our assumption.



Accident Type Distribution for Fair Weather

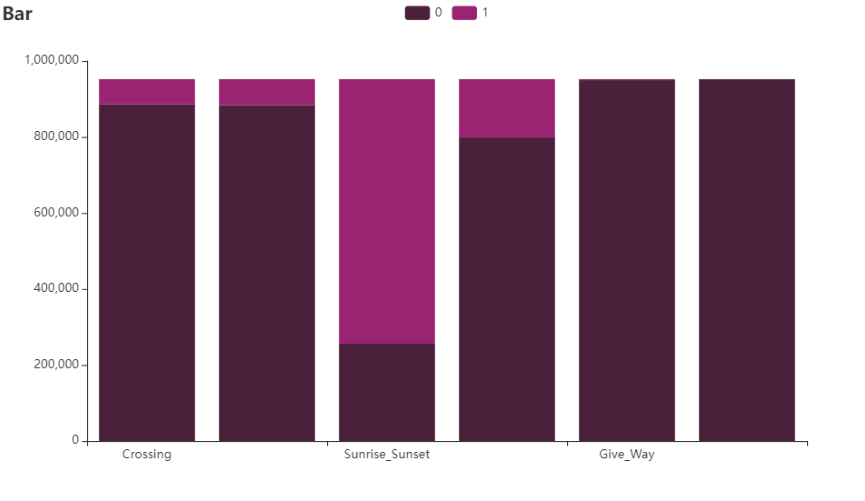


Accident Type Distribution for Snowy Weather

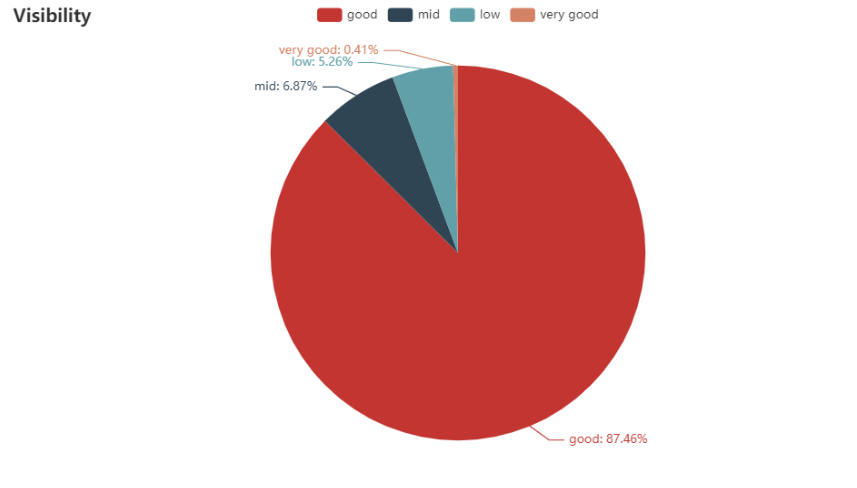
1. Most of the accidents occurred during the day, not at the intersection,

no precipitation, no signal lights, no yield signs, no speed bumps.

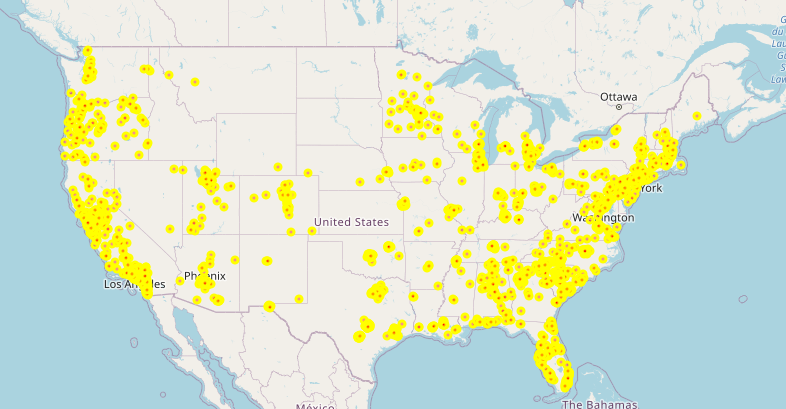
In this case, more traffic signs might be helpful to reduce the frequency of accidents.



1. Most accidents happened under good visibility, it seems like there’s no relationship between the visibility and accidents. However, same like the weather, perhaps the ratio of good visibility is more than the ratio of bad.



1. There’s the average distribution of traffic accidents in the US. When people in the area with more yellow dots, be careful.



3.4 Analyze the experiment result and why you obtained such results.

1. preprocessing:

Delete the useless columns, so at last there 25 [3.2] features left.

Using mean, common and nearest values full up with missing.

Using OneHotEncoder code with the non-numerical value.

Using PCA reduces the dimension of the final preprocessing dataset.

Standardization eliminates the heterogeneity between different attributes or quadrants, and reduces the variance between different attributes in the same quadrant or the same attribute in different quadrants.

1. parameters

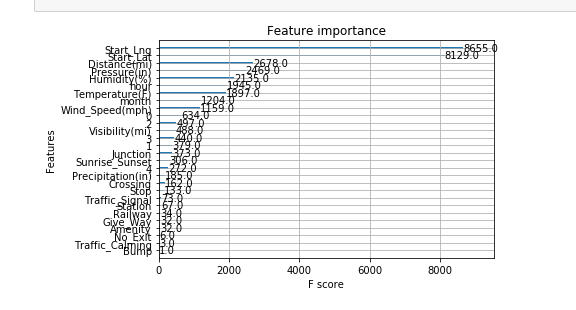
| booster | gbtree |
| --- | --- |
| objective | multi:softmax |
| num\_class | 3 |
| gamma | 0.1 |
| max\_depth | max\_depth |
| lambda | 2 |
| subsample | 0.7 |
| colsample\_bytree | 0.7 |
| silent | 1 |
| eta | 0.007 |
| seed | 1000 |
| nthread | 4 |

In the final the testing accuracy is 77.82%, it is within the acceptable accuracy range of the proposal.

1. Evaluation

The F score which in the feature importance can reflect the feature importance of the training model.

The top five are: Start Lng, Start\_lat, Distance, Pressure, Hour and Temperature.



1. **Conclusion** (Free page)

1. Be careful when driving in developed areas.

2. The morning and evening commuting time dangerous time in one day.

3. If there’s more holiday and less work, the accident rate might be high.

4. When the weather is bad, the probability of huge accident might be high.

5. Traffic sign missing may highly increase the rate of accident.

6. Visibility may not be a cause of traffic accidents.

7. The top 5 features which can strongly influence the severity of accident is Start Lng, Start\_lat, Distance, Pressure, Hour and Temperature.

Reference:

[1.1]:<https://www.kaggle.com/code/deepakdeepu8978/how-severity-the-accidents-is/notebook>

[1.2]:<https://www.kaggle.com/code/jingzongwang/usa-car-accidents-severity-prediction/notebook>

[3.1]:<https://www.kaggle.com/code/aaronds/us-traffic-accidents-analysis-and-prediction/data>

[3.3][3.4][3.5]: Hastie, T.; Tibshirani, R.; Friedman, J. H. (2009). ["10. Boosting and Additive Trees"](https://web.archive.org/web/20091110212529/http://www-stat.stanford.edu/~tibs/ElemStatLearn/). The Elements of Statistical Learning (2nd ed.). New York: Springer.

pp. 337–384. [ISBN](https://en.wikipedia.org/wiki/ISBN_(identifier)) [978-0-387-84857-0](https://en.wikipedia.org/wiki/Special:BookSources/978-0-387-84857-0). Archived from [the original](http://www-stat.stanford.edu/~tibs/ElemStatLearn/) on 2009-11-10.

[3.6]: Friedman, J. H. (February 1999). ["Greedy Function Approximation: A Gradient Boosting Machine"](https://statweb.stanford.edu/~jhf/ftp/trebst.pdf)

[3.7]: Tianqi Chen. [Introduction to Boosted Trees](http://homes.cs.washington.edu/~tqchen/pdf/BoostedTree.pdf)

[3.8]:<https://en.wikipedia.org/wiki/Predictive_analytics#Classification_and_regression_trees_.28CART.29>

[3.9]: Courant, Richard (January 1943). ["Variational methods for the solution of problems of equilibrium and vibrations"](https://projecteuclid.org/journals/bulletin-of-the-american-mathematical-society-new-series/volume-49/issue-1/Variational-methods-for-the-solution-of-problems-of-equilibrium-and/bams/1183504922.full)

[3.10]: [Abramowitz, Milton](https://en.wikipedia.org/wiki/Milton_Abramowitz); [Stegun, Irene A.](https://en.wikipedia.org/wiki/Irene_Stegun) (1970), [Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables](https://en.wikipedia.org/wiki/Abramowitz_and_Stegun), New York: [Dover Publications](https://en.wikipedia.org/wiki/Dover_Publications), Ninth printing

[3.11]:Chen, Tianqi; Guestrin, Carlos (2016). "XGBoost: A Scalable Tree Boosting System". In Krishnapuram, Balaji; Shah, Mohak; Smola, Alexander J.; Aggarwal, Charu C.; Shen, Dou; Rastogi, Rajeev (eds.). Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016. ACM. pp. 785–794. [arXiv](https://en.wikipedia.org/wiki/ArXiv_(identifier)):[1603.02754](https://arxiv.org/abs/1603.02754)

[3.12]: Sagi, Omer; Rokach, Lior (2021). "Approximating XGBoost with an interpretable decision tree". Information Sciences.

Additional remarks:

[3.2]:

Severity 951955 non-null int64

Start\_Lat 951955 non-null float64

Start\_Lng 951955 non-null float64

Distance(mi) 951955 non-null float64

Temperature(F) 951955 non-null float64

Humidity(%) 951955 non-null float64

Pressure(in) 951955 non-null float64

Visibility(mi) 951955 non-null float64

Wind\_Speed(mph) 951955 non-null float64

Precipitation(in) 951955 non-null float64

Weather\_Condition 935128 non-null object

Amenity 951955 non-null int64

Bump 951955 non-null int64

Crossing 951955 non-null int64

Give\_Way 951955 non-null int64

Junction 951955 non-null int64

No\_Exit 951955 non-null int64

Railway 951955 non-null int64

Roundabout 951955 non-null int64

Station 951955 non-null int64

Stop 951955 non-null int64

Traffic\_Calming 951955 non-null int64

Traffic\_Signal 951955 non-null int64

Turning\_Loop 951955 non-null int64

Sunrise\_Sunset 951955 non-null int64

month 951955 non-null int64

hour 951955 non-null int64