Traffic Accidents Predictive Analysis of Canada

Yuxuan Liu 20090374
Apr 22, 2022

Abstract

The purpose of this study is to build a machine learning model for predicting traffic accidents, to study potential accident causes, raw data, and human behavior, to help develop safety-related solutions, and to spread safety awareness. This experiment was conducted by using the crash dataset from the Canadian government webpage and the Sci-kit learning functions RandomForest and XGBoosting to build the model. The main results found are that this dataset and this method provide a very accurate prediction and a relatively high precision score, whereas the recall score is not ideal. Also, we used a time series prophet to predict the future trends in the number of traffic accidents and present them in the form of images.

Introduction

A traffic accident refers to an event in which a vehicle causes personal injury or property damage due to accidents on the road. It is estimated that one person is killed in a traffic accident every 24 seconds. Every year, around 1.3 million people around the world end their lives as a result of road traffic accidents. Another 20 million to 50 million people suffered non-fatal injuries, many of them disabled as a result.

Road traffic injuries take a huge economic toll on individuals, families and nations. These losses include the cost of treatment of the dead and injured, as well as the lost workforce of the deceased and the disabled due to injuries. However, road traffic injuries can be controlled through reasonable analysis and measures.

This project uses Canadian road traffic accident information data as the object of study. By the use of data mining-related theory and methods, we established a road traffic accident analysis system to reveal the risk factors affecting traffic. It provides a database and theoretical basis for accidents so that we can predict the accident development trend, improve accident prevention mechanisms, and improve the safety of the road traffic system.

Problem Statement

The purpose of this project is to enhance traffic safety policy and reduce the risk of traffic accidents through the study of risk factors. In the study, I will identify the risk factors that have a greater impact and make predictions to reduce future accidents. Moreover, I will focus on specific parameters that may have a great impact on traffic accident rates and try to find their distribution.

Experiment

Overview

In a full understanding of road traffic accidents vehicle, road, environment and other risk factors distribution characteristics in the dataset, I will establish a feature selection of accident severity risk factors analysis based on the classification ideas of data mining theory and the knowledge on XGBoosting and RandomForest modeling. Moreover, the Prophet-based time series combination prediction model was proposed to achieve point-in-time prediction of future accidents. Finally, I will use accuracy, precision scores and confusion matrix to evaluate the model and visualize the results by drawing graphs.

Introduction of Dataset

The dataset contains 23 columns in total, each of which represents a different feature in an accident. I will go over several important ones.

- C_YEAR, C_MNTH, C_WDAY, C_HOUR: These are date and time features. C_YEAR is the year, C_MNTH is the month, C_WDAY stands for days of the week and C_HOUR is the time from 0:00 to 23:59, where 00 stands for Midnight to 0:59, and 01 stands for 1:00 to 1:59, etc.

- C_SEV: This is the core of the project. It describes the severity of the accident. Collision producing at least one fatality is marked as 1 and collision producing non-fatal injury is marked as 2.

- C_CONF: It describes the circumstance of the collision, say, two vehicles in motion with the same direction of travel had a rear-end collision. I was considering adding this parameter at first, but it does not fit my purpose, so I dropped it.

- C_WTHR: This describes the weather of accidents. Considering common sense, it is an environmental factor that may cause a collision.

- C_RSUR: This describes the road surface of accidents. It is also worth to be recorded as a feature of an accident.

- V_ID, V_TYPE, V_YEAR: These are features of vehicles.

- P_ID, P_SEX, P_AGE, P_PSN: These are information about people in the accidents. P_PSN stands for the position that this person is in, say, the front row of the vehicle or pedestrian.

- P_ISEV: It is for personal injury severity and includes 3-level: No Injury, Injury, and Fatality. This can be the target value when we create the prediction model.

- P_SAFE: This stands for the safety of device usage in accidents.

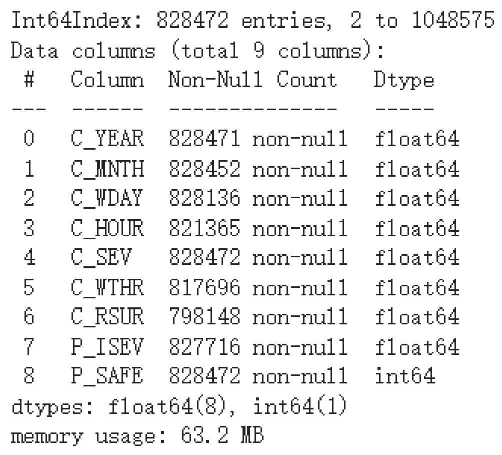
The file is too large and we can only read the first 1.04 million lines. So instead of loading all 20 years of data, we have just about 3 years.

Data Preprocessing

The purpose of this project is to study the risk factors of traffic accidents. Therefore, some features are irrelevant to my study and I will drop them in the data preprocessing step.

Since my goal is to predict and reduce the severity of the accident, C_SEV or P_ISEV (i.e. Collision/Person Injury Severity) will be set as the target. The date and time are also important factors for visualization, so C_YEAR, C_MNTH, C_WDAY and C_HOUR are needed. Though the dataset is too large that we can only read the first three years, I think that is enough for me to analysis. Then, considering the rest factors, we keep only the interesting ones: C_WTHR, C_RSUR and P_SAFE, which are the weather condition, road surface conditions and the safety device usage in accidents.

After dropping those values, I get the data that I need for further prediction. However, the data need to be cleaned up since there are non-integer values which stand for not applicable and unknown data. Thus, I will convert all the data values into numbers and replace the non-integer ones with "NaN" values. It uses a simple function to_numeric in pandas. The result is as follows.



```
Int64Index: 828472 entries, 2 to 1048575
Data columns (total 9 columns):
 #   Column  Non-Null Count   Dtype
---  ------  --------------   -----
 0   C_YEAR  828471 non-null  float64
 1   C_MNTH  828452 non-null  float64
 2   C_WDAY  828136 non-null  float64
 3   C_HOUR  821365 non-null  float64
 4   C_SEV   828472 non-null  float64
 5   C_WTHR  817696 non-null  float64
 6   C_RSUR  798148 non-null  float64
 7   P_ISEV  827716 non-null  float64
 8   P_SAFE  828472 non-null  int64
dtypes: float64(8), int64(1)
memory usage: 63.2 MB
```

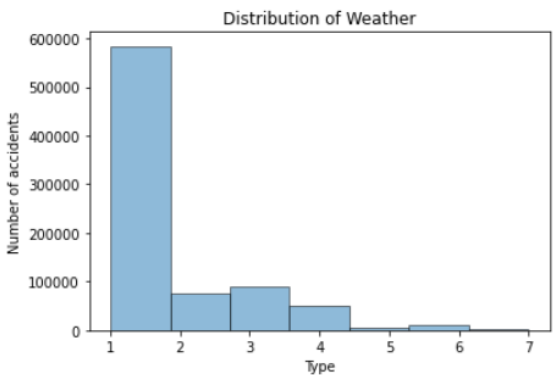
Obviously, now we can see many null values that needed to be filled in. So I start the second step of data preprocessing: filling in the missing values. There are three main ways to deal with missing data: deleting, filling in data, and ignoring it.

- Deleting: Delete the objects (tuples, records) with missing information attribute values, thus getting a complete information table.
- Data completion: Completes the information table by filling the null value with some value. Usually, the missing values are imputed with a constant value, mean, median or mode. We can also use methods like KNN to find each missing value, but it usually takes very long for the big dataset.
- Ignoring missing values: Using data mining methods that do not process the missing values, such as Bayesian networks and artificial neural networks.

We need to decide what to do with the missing value for each of the different features.

First, we notice that P_SAFE has the most missing values and it is about whether people are well equipped with safety equipment or not. Considering the three previously mentioned methods, first, we exclude the second one because it has 20% of the total data with missing values. It is well known that this method may lead to biased data when the proportion of missing data is large, especially when the missing data are not randomly distributed, and thus lead to wrong conclusions. After that, I choose between deleting and ignoring again: leaving it untreated directly would make the subsequent prediction part tricky. I decide to delete it to reduce the error it may cause.

Next, we'll deal with the remaining missing values. For weather and road conditions, we need to decide whether to delete or find a value to replace them. For this purpose, I drew histograms of these two features.



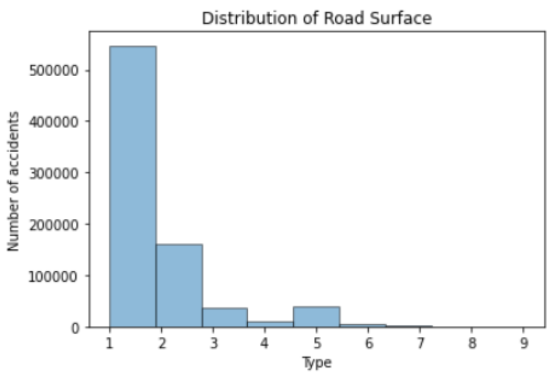


Fig 1. Distribution of Weather and Road Surface

As can be seen from the graph, there is one value in particular that stands out in both features, accounting for more than 90% of the overall. They represent a sunny day and a dry and normal road surface, respectively. I think this is a good reference value to fill in the missing values, and we take the most possible value to fill in the missing attribute values in a simple and easy way.

The next step is to compare P_ISVE and C_SEV. I would choose the latter because the latter tells the binary form of severity and directly distinguishes between minor or no injuries (1) and serious injuries and deaths (2). This does a good job of merging the two casualty types for us. Better yet, it has no NaN value. Therefore, I dropped P_ISEV.

Finally, we have to deal with the year, month, day and hour. I found only about 0.03% of the whole with NaN values for month and day, so I can just drop that row. For hours, instead, I filled it with its median value because it has a lot of missing values than previous ones. And, I convert them to integers for further usage.

Now that all the missing values are filled in, our data preprocessing comes to an end.

In the next step, I will separate the training set from the test set. This is easy to do with the help of the train_test_split function. The target value is C_SEV. I decided to set the training/testing ratio to 80/20.

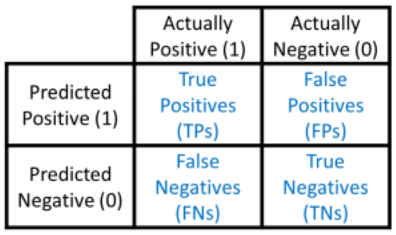
Modeling and Evaluation

Now the goal is to build a model to make predictions. First I tried the XGBoost model. When choosing the feature values, I was going to put all of them in, but I thought the year and month would have little effect on the number of crashes. So I added the rest of the features to the pipeline to build the classifier, and then I set some parameters of XGB and used grid search to finish the prediction of the model. To make it more flexible, the estimators were increased from 100 to 500. To make it less flexible, the max-depth needs to be larger, so that it could be less generalized. I also got the best estimator using grid_search_xgb.best_estimator_ to change and improve the parameters of XGBoosting.

However, it takes about 2 hours to wait for the result. So then I decide to apply XGBoosting directly without feature selection and grid search, which saves a lot of time. It turns out that the accuracy is very close. Hence, I just kept the latter one with the best estimator that I got previously for convenience.

Next, I tried Random Forest, a supervised machine learning algorithm widely used for classification and regression problems. It builds decision trees on different samples, uses bagging and feature randomization in classification, and tries to create a forest of unrelated trees whose predictions are more accurate than those of any individual tree. I mainly modified three of the hyperparameters: n_estimators, max_depth, min_samples_split. The first two parameters were modified for the same reasons as XGBoost, and the last parameter was The minimum number of samples required to split an internal node. Similarly, after finding the best estimator, I decide to apply RandomForestClassifier directly to improve efficiency since the evaluation result is close.

For evaluation, I chose to use the accuracy, precision score and recall score because it helps us understand which model is looking at all the different predictions and which model is actually more effective. These three numbers are calculated based on the confusion matrix. A confusion matrix is a summary of prediction results on a classification problem. The number of correct and incorrect predictions are summarized with count values and broken down by each class.



	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)

Fig 2. Confusion Matrix

The formula for calculating the accuracy is shown below.

- Accuracy = (TP+TN)/ TP+TN+FP+FN = (TP+TN) / all data
  Accuracy represents the proportion of samples (TP and TN) that are correctly predicted out of all samples (all data). When the data set is unbalanced, the accuracy rate will not be a good representation of the performance of the model. There may be cases where the accuracy rate is high and a few classes of samples are scored wrong, and other model evaluation metrics should be selected.

- Precision score = TP/(TP+FP) = TP predicted as positive samples

The precision rate of positive class represents the proportion of samples predicted as positive in the true class of samples predicted as positive

- Recall score = TP/(TP+FN) = TP true for POSITIVE samples(3)

The recall rate of positive class represents the proportion of samples successfully predicted by the model in the true class of samples predicted as positive. The recall rate of positive class is only related to the samples with true positive and not to the samples with true negative; while the precision rate is affected by both classes of samples.

Time Series

While linear regression studies the effect of features on the corresponding variables at a point in time, time series analysis studies the effect of historical data on the future. Since the dataset contains the year and month of data, I can create a time series analysis. The method is based on the theory of stochastic processes and mathematical statistics methods to study the statistical laws followed by random data series for use in solving practical problems.

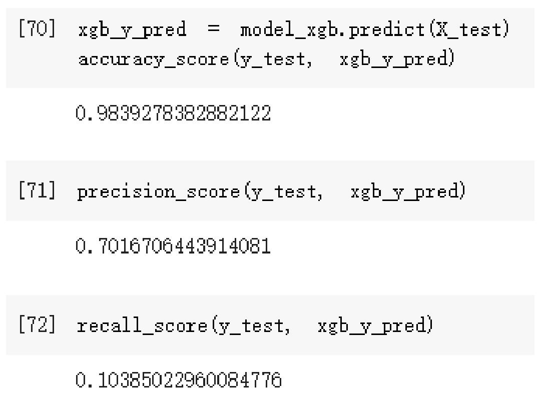
What I want to predict is the number of traffic accidents in the next ten years. Since the original database only gives us the year, month and day of the week, we can't know the exact date. My idea is to count all the crash times from the first to the seventh day of each month, and then model and predict them by month. I am aware that there will be a relatively large error if I do this, since I only have 32 months of data, but it should be possible to see some patterns.

It is important to note that I will use multiplicative seasonality. Multiplicative trend means that the trend is not linear and multiplicative seasonality means that the width or height of the seasonal period varies over time.

Results and Analysis

After applying these values to the two classifiers, I found that both models give me a pretty high accuracy. However, when the data set is not balanced, the accuracy rate will not be a good representation of the model performance. There may be cases where the accuracy is high and a few classes of samples are scored wrong. So I then compare the precision score and recall score.

The followings are the output of the XGBoosing model.



```
[70] xgb_y_pred = model_xgb.predict(X_test)
     accuracy_score(y_test, xgb_y_pred)

     0.9839278382882122

[71] precision_score(y_test, xgb_y_pred)

     0.7016706443914081

[72] recall_score(y_test, xgb_y_pred)

     0.10385022960084776
```

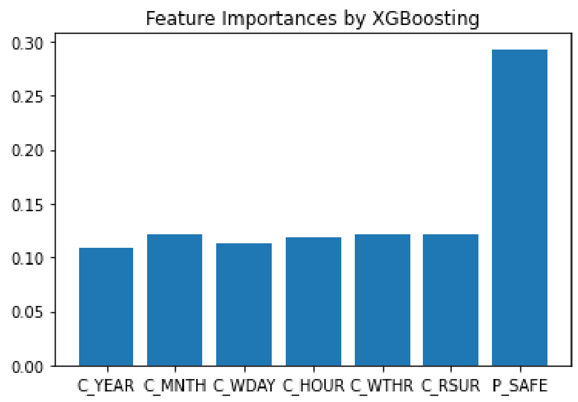
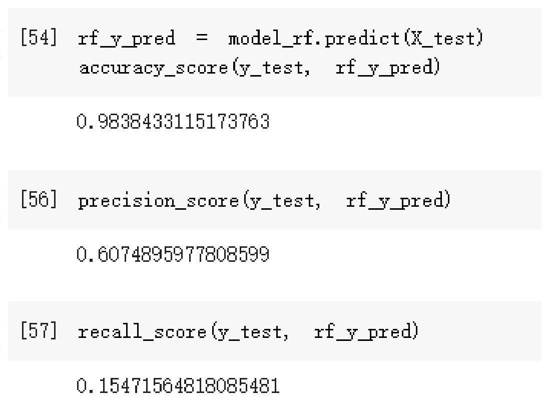


Fig 3. Result of evaluation for XGBoosting

The followings are the output of the RandomForest model.



```
[54] rf_y_pred = model_rf.predict(X_test)
     accuracy_score(y_test, rf_y_pred)

     0.9838433115173763

[56] precision_score(y_test, rf_y_pred)

     0.6074895977808599

[57] recall_score(y_test, rf_y_pred)

     0.15471564818085481
```

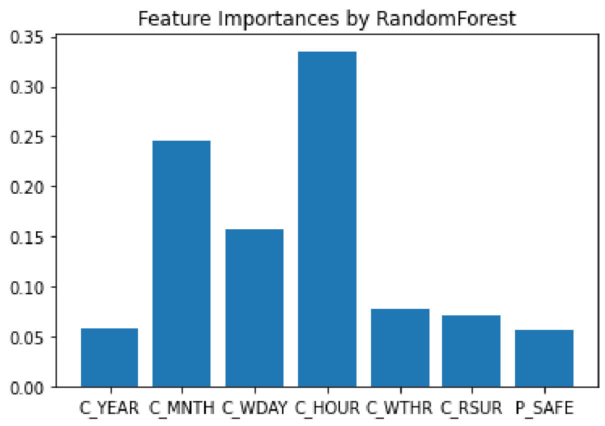


Fig 4. Result of evaluation for RandomForest

As we can see, both models give high accuracy scores, but the two other evaluation values differ from each other. A perfect precision score of 1 means that every result searched is relevant (but it is not stated whether all relevant documents are retrieved), while a perfect recall score of 1 means that all relevant documents are retrieved (but says nothing about how many irrelevant documents were also retrieved). XGBoosting has a high precision score and lower recall score compared to the RandomForest. And they also have different feature importance.

Next, we come to the time series. I know this forecast is not very accurate since the data in the dataset do not have an accurate date, but it shows a certain trend. First I tried to predict the trend of traffic accidents in the coming year when yearly seasonality is not specified.

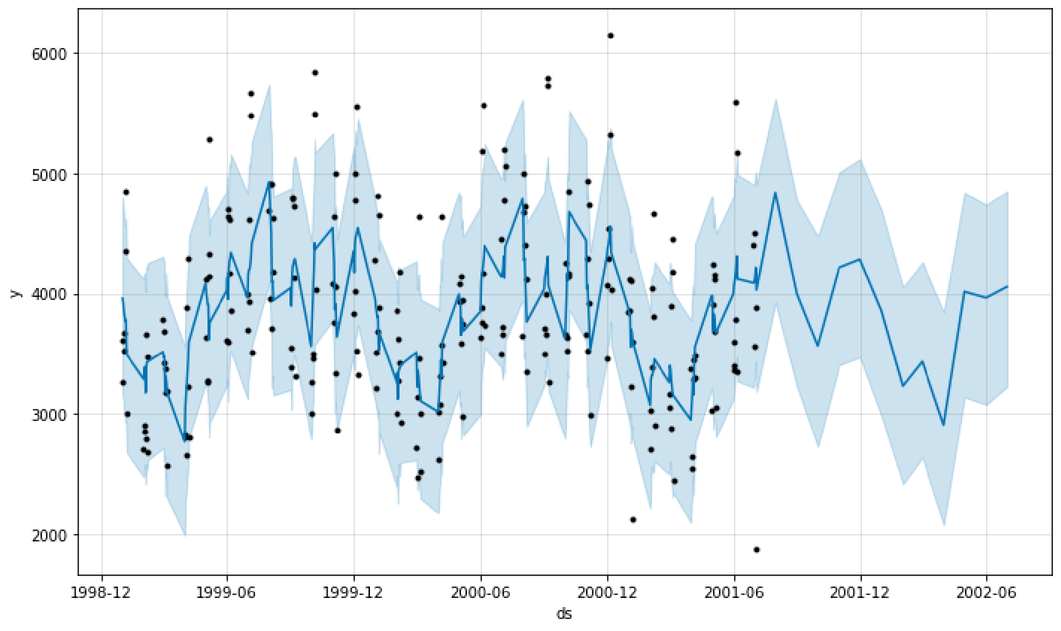


Fig 5. Prediction without yearly seasonality for future 1 year

Then I predicted the trend of traffic accidents in the next ten years when the yearly seasonality was specified.

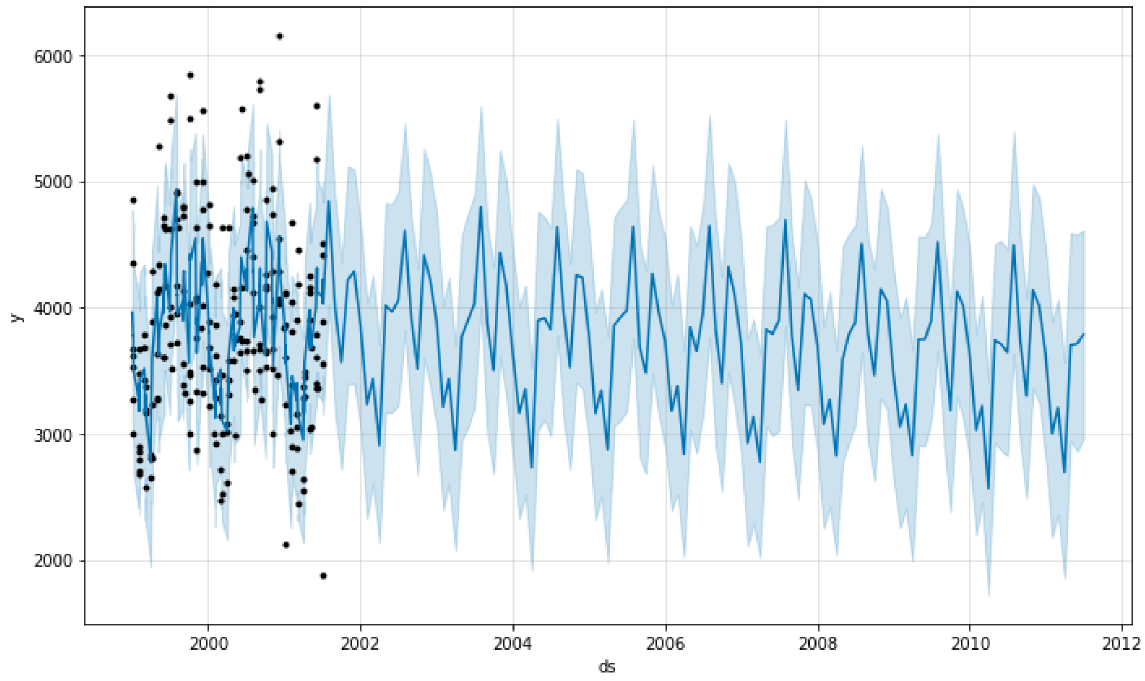


Fig 6. Prediction with yearly seasonality for future 10 years

From the graph, we know that the occurrence of traffic accidents decreases year by year.

I also made a point of counting the effect of each feature on traffic accidents, such as whether the number of collisions would be less on sunny days than on rainy days, or the number of collisions on dry roads than on slippery roads. But when I looked at the histogram, I found that it was actually the sunny and dry roads that had the most collisions. In other words, traffic accidents will happen every normal day, we always need to pay attention to traffic safety.

Conclusion

Using RandomForest, XGBoost and Prophet modeled separately, plus drawing some histograms, we have been able to obtain a lot of performance. The main results found are that this dataset and this method provide a very accurate prediction and a relatively high precision score, whereas the recall score is not ideal. By time series prophet, we predict that the future trend in the number of traffic accidents is decreasing and we present them in the form of images.

The study builds machine learning models for predicting traffic accidents, studying potential accident causes, raw data, and human behavior, to help develop safety-related solutions, and to spread safety awareness.

References

maximeg1. (2019, May 7). Predict severity of accidents. Kaggle. Retrieved April 22, 2022, from https://www.kaggle.com/code/maximeg1/predict-severity-of-accidents

National Collision Database. Open Government Portal. (n.d.). Retrieved April 22, 2022, from https://open.canada.ca/data/en/dataset/1eb9eba7-71d1-4b30-9fb1-30cbdab7e63a

Wikimedia Foundation. (2022, March 31). Precision and recall. Wikipedia. Retrieved April 22, 2022, from https://en.wikipedia.org/wiki/Precision_and_recall