Look Before You Leap

CSCI 35300/79502 Fall 2020

Final Paper

Hannah Do Kamil Sachryn

Contents

Github :

https://github.com/doguma/LookBeforeYouLeap

Project Poster :

https://github.com/doguma/LookBeforeYouLeap/blob/main/poster_LookBeforeYouLeap.pdf

3 Problem Description

Navigating quickly and efficiently around a city as large and busting as New York City is a priority for every commuter, and is facilitated by a robust navigation system. New York City has a workforce totalling over 4.8 million people, with over a million of those people taking their motor vehicles to work every day [1]. This number is spread amongst the five boroughs, but even with this separation traffic is an ever present problem in the life of every person. Being stuck in traffic has shown to be incredibly negative for the human body, with increases in blood pressure as well as a greater level of aggression found within people stuck in traffic, with the levels of these effects becoming greater when looked upon across multiple months of commuting in high traffic areas [2]. The air quality around heavily congested and high accident areas is also significantly lower and can lead to respiratory distress [3]. While traffic accidents themselves are difficult to prevent, it is possible to avoid coming across them via a robust navigation system. Yet these systems are seldom fast enough to react to the ever changing landscape of New York's streets. To help combat this issue we plan to introduce live data from the social media platform 'Twitter' to better predict traffic accidents.

Many drivers also use navigational tools during their commute in order to find the best possible route to their destination, but only recently have these tools begun to include real time data in their predictions. Such navigational tools, often referred to as a GPS, or Global Positioning System assist in finding the most efficient route to a destination based on a number of factors. Older tools which are still in wide use only take into account the most efficient route from one point to another, completely ignoring the current traffic conditions on the route. More modern software is able to take such data into account, showing the user the traffic ahead based on data gathered automatically via services installed in either the commuter's smartphone or within their vehicle itself. Yet these tools are not perfect, they rely on software being distributed amongst the populace and can thus find large holes in their predictions. Such tools are incredibly useful to commuters as

they are able to show them previously unknown routes calculated to have a faster arrival time as compared to simply showing the geographically efficient route to their destination, ignoring all other variables along the way.

To further increase the efficiency of GPS systems and route generation algorithms it is necessary to access more sources of information in real time. To accomplish this, modern social media is an incredibly useful tool, with millions of posts being generated every day even within new york city by itself. Tweets from the social media platform Twitter are made in real time by people currently at the site, as well as from companies and services which gather such data and attempt to deploy it in a timely manner. With input from social media it is possible to detect traffic even before the event, as a series of posts which describe an accident can appear long before a pileup of cars large enough to constitute a traffic jam can form. This is a massive step up over the previous systems which only tracked the current situation on the road, as it can make predictions into the future as to both the severity of a crash, as well as the overall impact on road conditions. This data would then be instantly integrated in the navigation system and be seen as a future road block, and then be avoided altogether.

In order to achieve this goal there are many issues to break through and solve, as well as to gather the necessary data for future predictions. Tweets from random people can often be incorrect or unnecessarily verbose, leading to predictions which are either highly over stated or difficult to compute automatically. Natural language recognition can be difficult to parse as people can use shorthand, slang, or even entirely misspell their posts. This forces us to create new and more complex ways to parse such data and keep track of ever evolving slang among the population. One way to help alleviate this issue is to use tweets from trusted sources, such as companies which keep track of police reports as well as requests for road clearing. Using these tweets can greatly increase the reliability of data which point towards the same accident, letting us confidently predict the future traffic jam even before traffic forms. Another major issue in gathering twitter data is the sheer volume of posts being generated every single day. One must use trusted sources to help go through such data and be able to identify relevant information from each and every post.

This data can then be combined with existing data regarding traffic accident severity in order to identify the traffic accident's effect on local traffic. Using the combined data, which not only includes the severity of each and every crash but also the weather, road, and trip conditions at the time is a great asset in predicting how the accident will affect local traffic. Using past data we are able to match the newly predicted accident with previous accidents of it's kind and use their known data in order to bolster the effectiveness of the twitter based accident predictor. There are many challenges in doing this, with Machine Learning being able to correctly identify and match such accidents to their historical counterparts.

The final goal of this project is to be able to combine previously known data regarding traffic accidents with the real time data generated by social media. Combining these two sets of data will give us a greater degree of control over how and when traffic accidents are predicted, with the ability to detect traffic before it is even formed. As social media evolves it will become easier to create these predictions, creating future growth for the algorithm as time goes on. With the power of previously created algorithms for detecting and displaying traffic congestion and the added data from social media we can greatly reduce the amount of issues caused by accidents, allowing for commuters to quickly and efficiently move away from an area where an accident is detected and avoid any traffic.

4 Member Roles

Hannah Do

Implemented collecting of datasets, feature selection from traffic accident record and retrieval of tweets through Snscape, Tweepy, and geocoding through ArcGIS. Also processed the tweet texts through various NLP processing and Vader methods (sentiment analysis) and converted the tweet dataframe into a frequency table to integrate to traffic accident records. Worked on merging the two datasets based on the distance between instances and used SMOTE and random instance generation to balance the class.

Generation of prediction scores from various ML models, cross validating predictions on data and implementing hyperparameter tuning to all ML models. Implementation of multithreaded algorithms for data and prediction processing. Development and testing of accident severity dataset. Research into State of the art models and papers and algorithm evaluation on datasets. Testing and evaluation of ML Models on different sets of predictors, and eliminating models which yielded no results or were incompatible. Implementation of the XGBoost model into the project.

4 State of the Art

Several papers take on the challenge of creating a better and more efficient model for navigation software. Examples include using data purely from past traffic data in order to gather likely accident rates. This data relies heavily on predictors such as the use of seatbelts, the current level of light at the site of the incident, and the usage of alcohol in people involved with the accident. Other papers have looked at using Twitter by itself in order to generate predictions of traffic accidents, but those models are plagued by issues of only having an incredibly small dataset which is not only heavily biased towards sources which report purely traffic accidents, but also must rely on sporadic data from social media users by itself. Language processing of twitter data is another big factor in the development of models which attempt to process human input in real time, and causes a great impact on the overall efficiency and accuracy of the model in question. Finally, few models have been able to successfully combine the data gathered from twitter and past accident data at the same time, but those were generally run in areas with little consistent data, unlike the large amount of twitter inputs generated in a major city like New York.

The ‘High-Resolution Road Vehicle Collision Prediction for the City of Montreal’ (Antoine Hébert et al., 2019) Predicted risk of accidents with high spatiotemporal resolution over a large area and time, allowing for useful identification of significant risk factors. This extended the state of the art by dynamically combining three large datasets, doing feature engineering for imbalanced classes, and implementing BRF in Spark. Large data processing has only come into the forefront

of accident prediction in recent years, with previous studies using small amounts of data focused on a carefully selected set of roads in order to better process and predict possible accidents. As data processing techniques as well as hardware has steadily increased in power and complexity we are now able to run these techniques on the scale of an entire city over several months, if not years using sufficient data and processing capabilities. To make use of this increase in capability the paper views road collision prediction as a regression problem which shows the risk of accidents in several ways, as well as a binary classification problem where each accidentis simply predicted as to whether or not it is likely to occur in the first place.

This great increase in data was brought on by the ‘open data’ movement which has begun to disseminate open data from government agencies in several countries, including the United States and Canada. Using this hitherto untapped dataset has allowed researchers to train Machine Learning models at an unprecedented rate with a massive increase in data availability. The combination of new algorithms, new infrastructure, and new datasets has been combined to create ‘High-Resolution’ predictions and has caused the predictions to explode in both accuracy and availability. Yet, these datasets have an intrinsic drawback, being that they are incredibly imbalanced due to the majority of data being gathered from a source which only shows accidents, rather than one which shows both times where accidents happen and do not happen. To combat this issue, algorithms which balance datasets on a small number of negative cases are used, and can generate inferred data equal to or even greater than the original set of positive occurrences.

In conclusion, the paper by Hebert et al is novel in the fact that it uses a relatively very large dataset when compared to previous works of its kind, and in that it uses a comparably very large area. Via the use of predictors the proposed model is able to not only predict whether or not an accident is likely to occur in any area based on the area’s current variables, such as the level of light available, but it is able to use regression in order to determine the severity of such an accident. Though the accuracy of such a prediction only reaches a level of 85% due to unknown factors such as the driver’s driving habits, physical state, or mental state, all of which have been shown to have a major effect on one’s judgement and driving ability. In order to build upon this robust model we are able to combine such data with real time twitter data, gaining a degree of control in when predictions are made, and how they affect the overall accuracy as well as the

severity of such an accident, a factor which can have an effect on the overall traffic situation in the area.

The second paper, “From Twitter to detector: Real-time traffic incident detection using social media data” (Gu et al., 2016) focuses on using twitter data in order to supplement or even replace sensory data from other sources. The benefit of using twitter data is not only that they are available in real time, but also that they are available to the general public for no cost, making the model easily available to anyone who wishes to use it but also coming at no cost to the developer or user. The major drawback of using tweets are real time data rests in the fact that natural language processing is difficult to make perfect, and those tweets then have to be processed and scanned for several important keywords that relate them specifically to traffic prediction, or traffic incident prediction.

The daunting task of trying to process every single tweet for related keywords in order to match it as a report of a traffic incident can be lessened via using known sources of information, which is guaranteed to release accurate and consistently worded reports on accidents. Secondly, only a small amount of twitter data is actually geocodable, around five percent, which means that the majority of tweets are unusable before even being processed via natural language recognition, as there would be no way to accurately place the location of the accident. This creates further need for these reliable accounts, as they are known to have the exact location of the accident either attached to the tweet content, via a street location or an exact coordinate, or have the geocoded location already embedded within the tweet itself.

Other approaches to generating reliable and live data are either incredibly expensive to run on a large scale, or require the use of specialized software installed on the commuters smartphones in order to gather data. The popularity of twitter makes it incredibly likely that each user already has the software installed, but a tweet regarding an accident has a much higher rate of being correct than simple geodata from many people who may or may not be at the scene of an accident. Twitter data is inexpensive to use, requiring only an internet connection to function, and can be integrated into any modern computing system.

Twitter users have been shown to very quickly react and tweet about accidents they encounter on the road. This can be combined with natural language processing to create

predictions about events before anything else is capable of catching them, especially since the kinds of accidents which cause traffic are likely to be encountered by a large number of people at the exact time of the accident, which can then be immediately tweeted about or tweeted about shortly after the incident. It has been shown that Twitter is capable of spreading news of an event faster than any other form of traditional media [4]. This makes it very likely that even before other forms of accident prediction are able to catch the accident, some twitter user who witnessed the accident has already sent out a tweet about it, allowing the model to quickly cross reference it against other tweets, and form a prediction.

In conclusion, the paper by Gu et al is capable of making cheap, accurate, and easel accessible predictions about whether or not an accident has occurred. Via the use of known reliable sources of information, as well as the ability to gather random data from individual users of twitter, the model is capable of making predictions faster than traditional methods done via sensors or direct sight of appointed personnel. It is also capable of outperforming software based traffic monitoring by making it possible for a single user to report an accident, rather than using multiple stopped vehicles as an indication for a traffic accident. This also allows us to gauge the severity of an accident based on the tweet's content, which is not possible to do via passive monitoring software. Our paper improves upon this idea by including past traffic data which will be capable of either corroborating the prediction of an accident, or have the twitter data be used as a supplementary data point in accident prediction.

5 Method

The traffic data was collected from the NYPD collisions report, and divided into 1 month, 3 month, and 6 month timeframes. With feature engineering, unnecessary columns that resulted in huge amounts of dummy values were dropped. Twitter data was retrieved through Snscrape (links) and converted through Tweepy (texts) and stored in the according timeframe. The text processing of the tweets used NLP methods such as nltk, regular expression operations and sentiment analysis was performed through Vader Sentiment analysis. With the addition of neutral and polar

sentiment scores, camel cases, stopwords, redundant words were removed, and acronyms were replaced (Figure 1). Geocoding of the processed tweets were then performed through the ArcGIS function for retrieval of latitude and longitude values (Figure 2), and counted to get frequency for each location.

acronyms	replacement
bqe	brooklyn queens expressway
us9	united states route 9
us130	united states route 130
i87	interstate 87
i95	interstate 95
bklyn	brooklyn
bk	brooklyn
expy	expressway
rte	route
rt	route
eb	east bound
nb	north bound
nj	new jersey

Figure 1. Examples of specific acronyms replaced with phrases available for ArcGIS geocoding

	tweet	separate camel cases	replace acronyms	remove stopwords	longitude	latitude
12	Cleared: Construction on #NY30 Both directions...	Cleared Construction on NY30 Both directions...	cleared construction on new york 30 both di...	cleared construction new york 30 directions ne...	-1.045049	53.880315
13	Incident on #Q48Bus from Roosevelt Ave: Main S...	Incident on Q48Bus from Roosevelt Ave Main S...	incident on q48bus from roosevelt ave main s...	incident q48bus roosevelt ave main st roosevel...	-73.588960	40.678550
14	Incident on #US9 Both directions at County Rou...	Incident on US9 Both directions at County Rou...	incident on U.S. route 9 both directions at...	incident U S route 9 directions county route 3...	-73.917929	40.868738
16	Updated: Incident on #BM3 Both directions at 5...	Updated Incident on BM3 Both directions at 5...	updated incident on bm3 both directions at 5...	updated incident bm3 directions 57 st 2nd ave ...	-74.007140	40.714550
17	Updated: Incident on #M101Bus at Second Ave ht...	Updated Incident on M101Bus at Second Ave ht...	updated incident on m101bus at second ave ht...	updated incident m101bus second ave https co u...	-73.988656	40.727283
19	Cleared: Construction on #92Bus Both direction...	Cleared Construction on 92Bus Both direction...	cleared construction on 92bus both direction...	cleared construction 92bus directions scotland...	-73.976250	40.621120
20	Cleared: Construction on #Q32Bus at E 60th Str...	Cleared Construction on Q32Bus at E 60th Str...	cleared construction on q32bus at e 60th str...	cleared construction q32bus e 60th street lexi...	-73.970674	40.764026
21	Incident on #BM3 Both directions at 57 St and ...	Incident on BM3 Both directions at 57 St and ...	incident on bm3 both directions at 57 st and ...	incident bm3 directions 57 st 2nd ave https co...	-74.007140	40.714550
24	Incident on #Q15Bus from Roosevelt Ave: Main S...	Incident on Q15Bus from Roosevelt Ave Main S...	incident on q15bus from roosevelt ave main s...	incident q15bus roosevelt ave main st roosevel...	-73.588960	40.678550
25	Incident on #BrooklynBridge EB at Brooklyn Bou...	Incident on Brooklyn Bridge EB at Brooklyn Bo...	incident on brooklyn bridge east bound at...	incident brooklyn bridge east bound brooklyn b...	29.769650	-24.868210

Figure 2. Text processing (left to right) - separation of camel cases, acronym replacements, removal of stopwords, and location retrieval through geocoding : returning longitude and latitude values

The collected datasets were then merged based on the longitude, latitude values with a threshold of maximum *euclidean distance* between two instances - the default value used was 2000. If the distance between two instances were less than the threshold, the 'severity' column was incremented and the 'distance' column (initial value of 10000) was assigned the minimum distance between the current location and nearest accident prone area retrieved from the tweets. The sentiment of the latest instance was stored as neutral and polar scores (Figure 3).

	longitude	latitude	Frequency	polarity	neutral
11	-73.958048	40.619211	2	0.0000	1.000
10	-73.968886	40.672875	2	0.0000	1.000
9	-73.977580	40.752125	1	0.1027	0.877
15	-73.800325	40.703240	1	0.4588	0.800
14	-73.934931	40.713819	1	0.0000	1.000
13	-73.946861	40.590935	1	0.0000	1.000
12	-73.949580	40.650100	1	0.1027	0.877
0	-74.068302	40.600070	1	0.0000	1.000
1	-74.046500	40.606030	1	0.1027	0.833
7	-74.007328	40.654323	1	0.0000	1.000
6	-74.009464	40.762307	1	0.5267	0.672

Figure 3. Frequency chart of the retrieved location from the tweets. Highly redundant locations were disregarded due to possible error in geocoding.

COLLISION_ID	VEHICLE TYPE CODE 1	VEHICLE TYPE CODE 2	distance	severity	polar sentiment	neutral sentiment	target
4352699	Station Wagon/Sport Utility Vehicle	NaN	151	79	0.0951927	0.890857	1
4364644	Sedan	NaN	122	83.0	0.09519267207963791	0.8908573205572614	1
4349813	Station Wagon/Sport Utility Vehicle	Station Wagon/Sport Utility Vehicle	205	87	0.0189853	0.975063	1
4352706	Sedan	NaN	3	25	0.280911	0.76806	1
4329138	Station Wagon/Sport Utility Vehicle	NaN	396	90.0	0.07370066027867632	0.9090990115569674	1
...	...	...	...	...	...	...	...
4335034	Sedan	Sedan	207	12	0.0165304	0.970694	1
4358043	Station Wagon/Sport Utility Vehicle	Sedan	270	58.0	0.05872441746946828	0.9301630808235241	1

Figure 4. Distance, severity, neutral sentiments added as additional features to the existing traffic record data. Non-accident instances (target value of 0s) were appended to the bottom on the dataset

The full feature set including accident record and twitter features consists of a single class, in which accidents occur. Since this causes class imbalance, we added negative samples of non-accident cases by getting random values within a specific range for each feature in order to prevent biased or obvious differences, and these instances were appended on the bottom of the existing dataset. We initially put in 30 instances for 1 month data, 60 for 3 months, and 90 for 6 months. Using SMOTE function allowed this still unbalanced data to have an equal number of target values for each class.

We chose SMOTE, or Synthetic Minority Oversampling Technique since the procedure is different from typical oversampling. Instead of duplicating the class with less data, leading to less variance of the data distribution, SMOTE works with k-nearest neighbor algorithms to build synthetic data. Initially, random data from minority classes are chosen, and k-nearest neighbors from the data are set, then the synthetic data would be made between the random data and randomly selected neighbors. This is repeated until the minority class has the same amount of data as the majority class. We have utilized SMOTE in order to increase the non-accident instances (Figure 5a), however, having equal number of accident and non-accident cases are still unrealistic, so we have also added more proportion to non-accident instances (target value of 0) by manually creating random instances (Figure 5b).

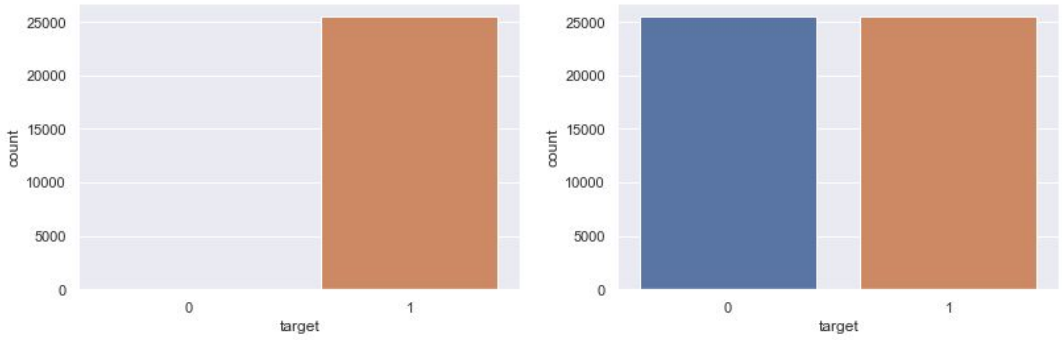

Figure 5a. Before and after SMOTE oversampling of the minority class : the result (right) shows equal number of target 0s and 1s after oversampling

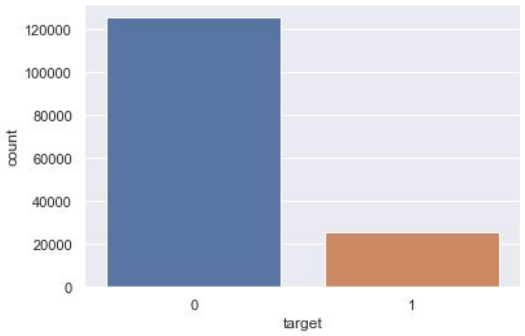

Figure 5b. Larger portion of negative instances (target 0) added to SMOTE data

The synthetic data with different proportions of class were then run by different types of ML models with cross validation in 1 month, 3 months, and 6 months timeframes. The 6 months data was quite a bulk, and did not have the best running time, so we chose random sampling of the instances to reduce the size of the total data. We used kNN, SVC-rbf, Gaussian Naive Bayes, Decision Tree, Random Forest, and Logistic Regression. We tried to choose diverse models since we did have any information whether the features would be correlated to each other, or if the algorithms would require linear, polynomial, or multi-dimensional functions. The following diagram is a summary of the text and data processing procedure to produce a pre-processed and merged dataset (Figure 6).

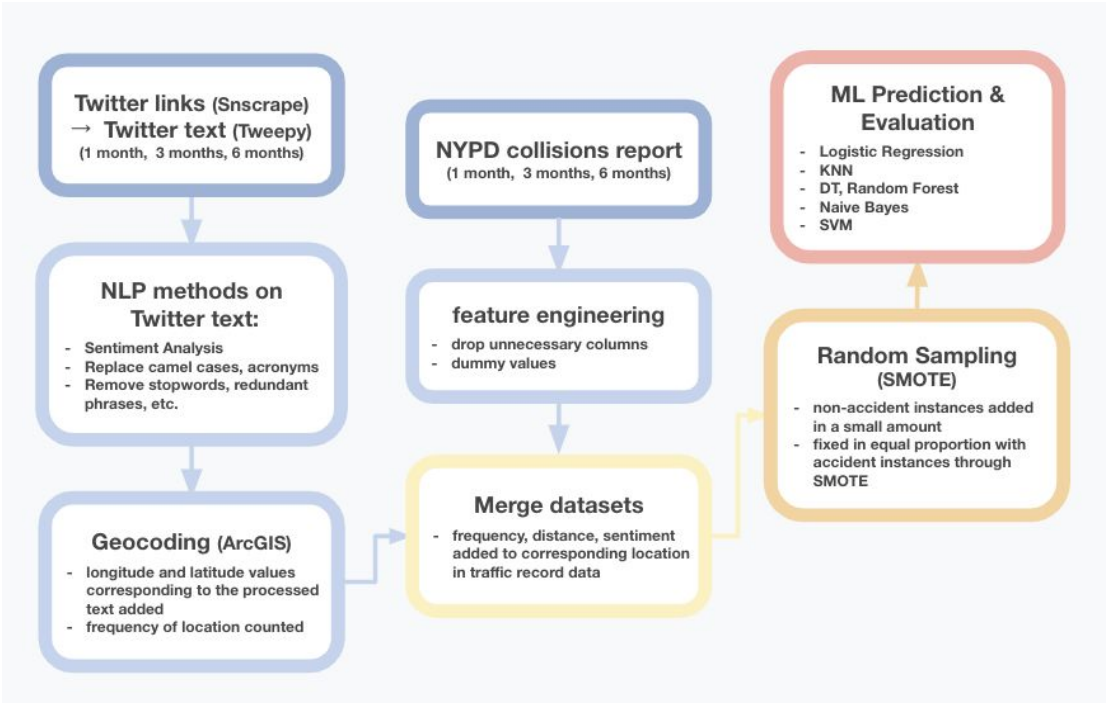

Figure 6. Diagram of data processing (accident record and twitter data), feature selection, prediction and evaluation procedure

6 Results and Evaluation

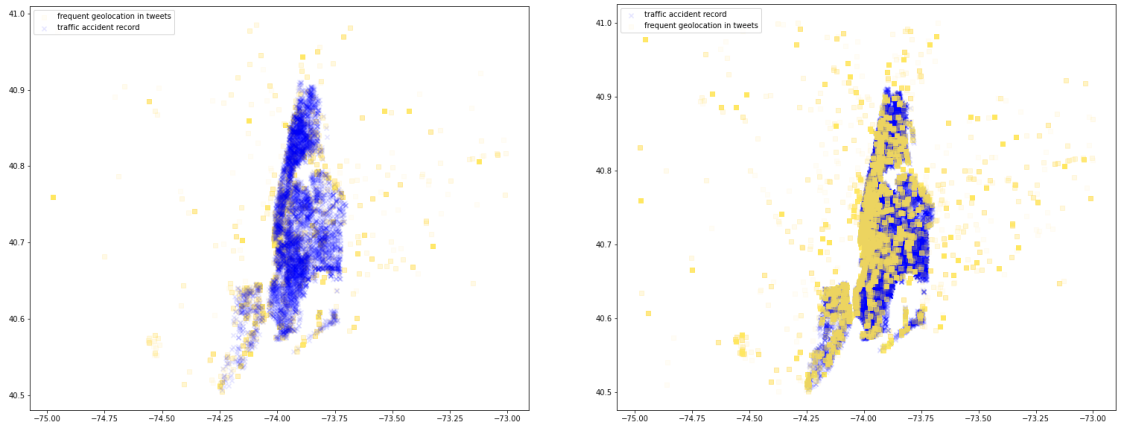

Figure 7. Scatterplot of traffic accident record (blue) and twitter geocoded locations (yellow)

As observed in Figure 7, the locations from the geocoded tweets and accident record were compared on a scatter plot, and twitter geocoded locations in yellow returned a wider distribution as the 511 tweets covered new york state, including new york city. It also showed higher density near the intersection between the city and Queens, or upper area of Brooklyn. Data distribution in Staten Island returned a similar result for both datasets, however the traffic record shows higher concentration near Bronx and Brooklyn.

	1 month			3 months			6 months		
ML models	no twitter 1:1	SMOTE 1:1	negative boost 4:1	no twitter 1:1	SMOTE 1:1	negative boost 4:1	no twitter 1:1	SMOTE 1:1	negative boost 4:1
kNN	1	1	1	1	1	1	0.96774	0.99905	0.97166
SVC	1	1	1	1	1	1	0.86500	0.89918	0.95895
Gaussian NB	1	0.99810	0.99345	1	0.99966	0.99984	0.90769	0.98816	0.99294
Random Forest	1	1	1	1	1	1	0.95759	0.99974	0.98890
Logistic Regression	0.54782	0.89225	0.90457	0.58102	0.95711	0.92540	0.48449	0.97301	0.941
XGBoost	1	1	1	1	1	1	0.95116	0.99928	0.99866

Table 1. Accuracy of prediction through different machine learning models performed on extracted data : 1 month, 3 months, and 6 months record of accident record, twitter dataset with smote method and negative instance boosting (4:1 proportion)

Many of the prediction models returned a redundant value of 1 as accuracy scores, which we assumed that it may be due to lack of data for 1 month or 3 months period. From 6 months data, we could observe that most of the models had high accuracy of classifying the target values, even without the Twitter features. However there was room for improvement, especially for Logistic Regression, and it was observed that adding the Twitter features and balancing the dataset with SMOTE and negative boosting returned higher accuracy values. The SMOTE dataset had better prediction scores than negative boosting, however the negative boosting still showed significant increase in performance compared to the features without twitter analysis (Table 1).

	3 months			6 months		
	no twitter 1:1	smote 1:1	negative boost 4:1	no twitter 1:1	smote 1:1	negative boost 4:1
precision	0.502156	0.933666	0.751797	0.676999	0.947914	0.787041
recall	1.000000	0.986144	0.852816	0.653652	0.979288	0.885943
f1-score	0.668580	0.959188	0.799127	0.665121	0.963346	0.833569

Table 2. Evaluation of Logistic Regression model on different types of datasets, precision, recall, and f1-scores were measured

To further examine the prediction scores, we retrieved precision, recall and f1-score for each dataset. It was shown that the longer the period, the more stable precision-recall tradeoff values were. And smote and negative boost datasets that included Twitter analysis had overall higher scores in precision, recall, and f-1 score values compared to the accident record only dataset (Table 2).

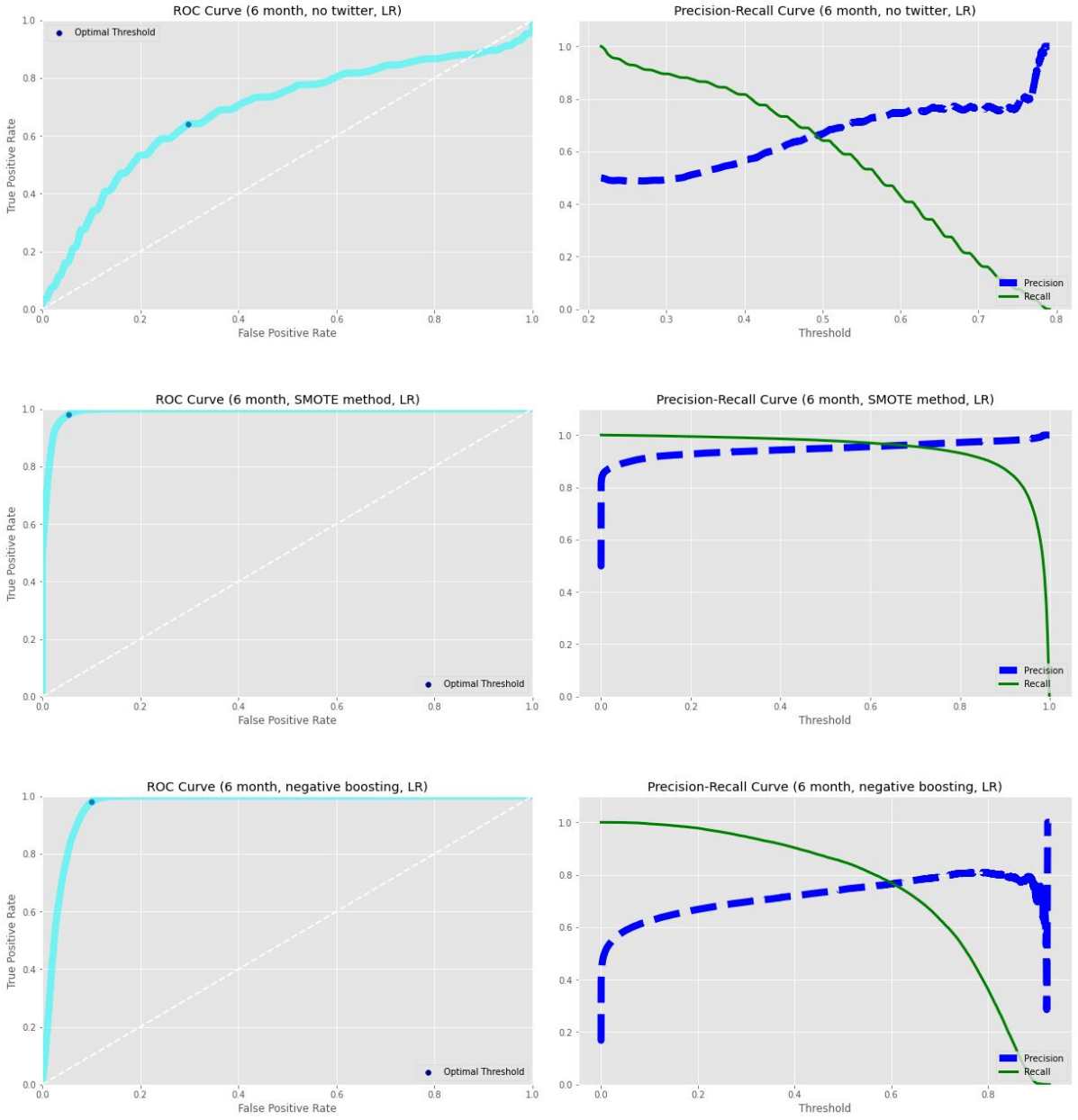

Figure 8. ROC Curve and Precision-Recall Curves for according datasets : no twitter, smote, negative boosting / 6 months

Through feature scoring function provided by Random Forest Classifier, each feature was measured by the amount, or importance of how much it affected the classification of the target value (Figure 9). The scale shows that the distance feature had the highest percentage, polar and neutral sentiment the next, and so on. As the top three features are from Twitter analysis, this suggests the accuracy of the prediction increased mainly because of the additional feature sets.

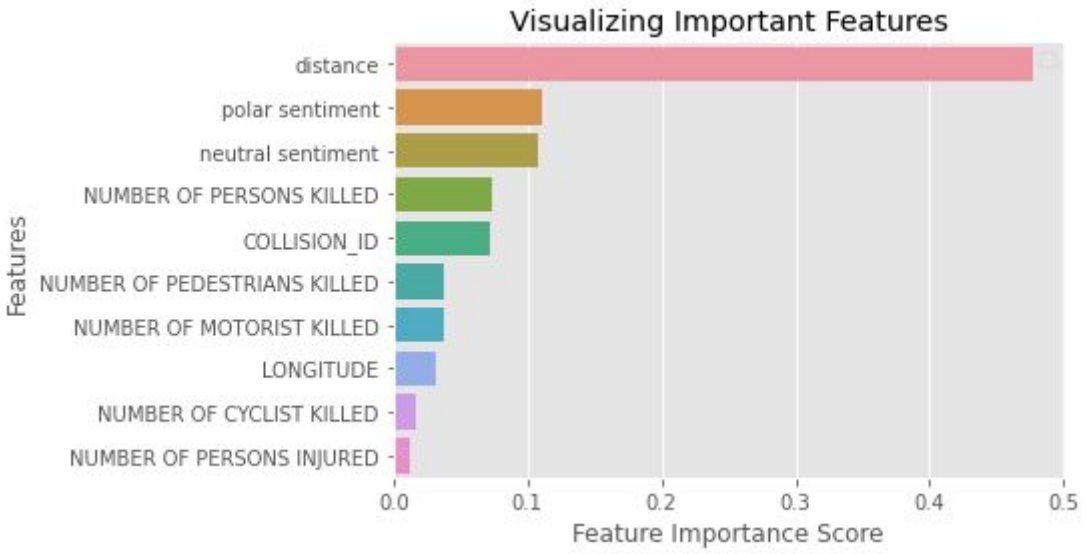

Figure 9. Feature importance scores through Random Forest Classifier on negative boost dataset, 6 months

7 Discussion

a. Dataset collection

Our initial choice of traffic accident record was the Countrywide Traffic Accident Dataset provided on Kaggle by Mossavi et al. [5]. However, a big portion of the dataset was missing for Queens, which made us choose the NYC Open Data instead. Similar with the Twitter data, we began with multiple public Twitter accounts rather than private accounts to avoid biased data. However, TotalTrafficNYC, NYPD Traffic and NYC DOT still contained many advertisements and information unrelated to traffic accidents, in which we have limited ourselves on the collection of using 511 NY tweets. Plotting the distribution on the geolcations retrieved showed slight disparity between the actual accident record and the geolocation, this leaves room for improvement in collecting fair distribution of Twitter locations via further processing and understanding of the text for geocoding.

b. Class imbalance

Having instances of one class for binary classification, we had to create random negative instances in a given range. For 1 month data, 30 instances for each date was created since SMOTE required a minimum amount of negative instances to run

oversampling. Apart from SMOTE data, additional negative instance boosting was performed since actual traffic accident instances do not have the same probability as non-accident instances. The odds of getting into a car accident is way apparently lower than ½, however we assumed adding on too many random negative instances may result in the same effect of adding noise to the dataset, so a proportion of 1:4 was temporarily assigned for accident and non-accident instances. The latter dataset did not have as high of an accuracy as SMOTE (1:1) dataset, but still showed significance in improving prediction scores compared to the accident dataset without any Twitter features.

c. Difference in performance of machine learning models :

i. Models with initial high accuracy scores (kNN, Random Forest, XGBoost)

The random forest and XGBoost algorithms are both tree-based learners in ensemble learning by collecting votes of different trees to improve prediction accuracy. They both train trees in which random forest focuses on different subsets from the data, while XGBoost trains sequential trees with prioritizing misclassified instances for each iteration. kNN algorithm can be seen in a similar context in identifying k nearest neighbors to obtain the majority vote. Then unlabeled data is linked with the labeled data, branching out to other instances.

These ML models work best when the features are correlated to each other, and running our dataset with seaborn pairplot shows that features such as ‘severity,’ ‘number of motorists injured’ or ‘number of persons injured’ seem to be in correlation with each other (Figure 10), which has significance to the ML models as kNN examines the similarity in neighbors while tree based algorithms can de-correlate trees to run its algorithms.

Figure 10. Correlation comparison graph of the full feature set - seaborn pairplot : negative boost dataset / 6 months

ii. Models with initial average accuracy scores (SVC, Gaussian NB)

For Support Vector Classifier, since the number of features are less than the number of instances in the dataset, a polynomial kernel was used with hypertuning the gamma and C values. And Gaussian NB was naively run as it uses classification techniques based on Bayes' theorem for multiplication of the probability scores. The scores from both models were adequate to pass as great performance, but SVC may not have performed as well due to the size of the data being large and Naive Bayes having strong assumptions on feature independence while the dataset tells us otherwise.

iii. Model with greatest difference in accuracy scores (Logistic Regression)

Logistic Regression separates the inputs into two regions with a linear boundary, and is often used in multi-variable or binary classification as it is easy to regularize. However it is

easy to be outperformed by other complex classifiers as the performance depends highly on the feature set. This may be the reason why the classifier may not have performed well before the addition of the new feature sets since the existing accident record may not have had enough features to be identified as important variables. And since traditional logistic regression is based on a linear form and it is extended to a higher-order polynomial equation, the approach itself is very different from the previous ML models. Therefore creating a full feature set with Twitter analysis helped the model to utilize the given variables to produce better equations and prediction scores.

8 Barriers and Future work

Through the project we came across several bottlenecks, sources of missing data, hardware limitations, and time limitations. Through working with this project we were able to break through several issues we did not foresee. Data sets were often lacking in certain data, and even gathering the data itself could often be a challenge unto itself. In the end, the major constraint upon the project was the amount of time we had in order to compile every part, with several parts only being doable once a predecessor is satisfied.

One of the major issues faced during this project was the limitation on the amount of data we could retrieve at any one time, regardless of our ability to process it. The twitter API which we used to retrieve live data from Twitter servers had a limitation placed upon it, this caused a major bottleneck in our data retrieval rate, causing significant delays on every other part of the project which relied on gathering this data. In order to circumvent this issue we had to create multiple developer accounts capable of retrieving data and use them to supplement the slow trickle of data we were receiving. In order to properly continue work on this model we would require an unlimited access to Twitter's data, giving us real time access in perpetuity.

Processing natural language was a difficult process, and required many manual adjustments to properly choose and process the correct strings of data. Twitter is a platform where each posted message has a set limit on it's length, and has thus evolved a language of its own, full of shorthands and acronyms made to convey a message longer than the hard limit would imply. To combat this, we created a list of conversions from shorthand words to their full equivalents, but this

list is inevitably incomplete. To create a full list of acronyms and shorthands, and link them to machine learn-able equivalents would require a large amount of manpower to pour through thousands of tweets and identify repeating patterns. In the future it would be required to attempt such an effort, each new acronym would unlock a slew of tweets which could yield even more data. The issue of twitter data parsing leads into another issue where it is difficult to use the tweets effectively, and to categorize them. Each tweet is unique, unless it comes from a trusted source which only reports upon traffic incidents and clearances. Tweets that come from individual persons have little structure, and even sometimes break the conventions of the language they are written in. Analyzing this data is very difficult and time consuming, required fine tuning of our model to accept different words and combinations thereof. To combat this we would require far more processing power or time, in order to create a learning model which is capable of categorizing the data itself, or to create a system of adjustments and classification to later feed into our model.

Another of the major hurdles we encountered was one of missing data within the dataset. Each instance of data had to be confirmed to make sure it was not missing any important fields, and removed if it was. Furthermore, we found that we were missing a large amount of data from certain geographical regions with little reason as to why it was missing. To combat this issue we had to source a new dataset, one which used very different variables from our previous dataset, and had to adjust our model accordingly. While this was not a fatal setback, it created issues of its own, the least of which was having to redesign our model around this new set of features, and rerun all previous tests and tuning for the relevant machine learning algorithms.

The last major hurdle was the process of generating geocodes from individual tweets. Only about five percent of the individual tweets have geocodes embedded in them when retrieved from random individual sources. To further complicate matters, geocoding individual tweets took a large amount of time to process in bulk. While individual geocodes once identified could be processed in real time, using them to train the model took a considerable amount of time. Along with this issue certain tweets could not be processed, and would cause algorithms to fail at random unless scouted beforehand. Finally, many tweets would mention a bus stop nearby instead of giving a proper geocodable location. Further work would involve adding the new york bus system to the model, allowing for mentions of bus lines to result in a geocodable value.

9 Lesson learned on ML topics

From gathering data to the execution of Machine Learning algorithms on processed data we were able to learn far more than by using data which was known to provide expected results. Being able to freely choose the topic and how to go about forming it into a coherent whole was a lesson in itself. Choosing a suitable set of data for our project was a challenge, and required repetitive trial and errors. Datasets can be incomplete or missing certain variables which completely changed the way a model is run. And using imbalanced datasets brought on its own advantages and disadvantages to the machine learning model as a whole, and the act of balancing strengths and weaknesses of each dataset was a lesson in itself.

Through repeated testing of each algorithm we were able to identify their strengths and weaknesses, but doing so was incredibly time expensive. Each model had its own running time, Support Vector Classifier took a comparatively long amount of time with hyperparameter tuning and running the model, while other ML models had faster performance. Through research, tuning, and creating algorithms of our own to validate data we were able to expand our own knowledge of the intricate parts of machine learning.

Processing large amounts of data, as well as trying to combine sources of data ourselves was a valuable lesson on the importance of having a strong and well thought out process during the entire project. Each step of the project took very large amounts of time to complete, and often had to be done sequentially, with each successive step requiring the data processed by the previous step. With direction of what parts of the project had to be done in which order we optimized the development of the project as a whole. This prevented instances where an issue arose partway through development, caused by an earlier step, in which the lengthy algorithms would have to be run again. These experiments were a valuable lesson in creating more robust algorithms, as well as creating ways to test the project as a whole on a smaller scale which could be run in faster time.

Finally, we learned a great deal about how each type of algorithm reacts to certain sets of data. Ensemble learning models such as Random Forest and XGBoost showed great performance when run on correlated features. The tree-based learners could de-correlate the features to collect

votes from different subsets or instances to improve prediction accuracy, and the kNN algorithm identified k nearest neighbors to obtain the majority vote. These algorithms had great performance even before the addition of our Twitter data. The Gaussian Naive Bayes, on the other hand, relied on assumptions that were untrue on the dataset, yielding not as significant scores as the other models, at the same time being one of the most expensive models to run. The Support Vector Machine showed differences in scores in having enough feature sets for the model as the performance increased with extension of the feature sets. Lastly, the Logistic Regression model stood out as it had a drastic increase in performance before and after the additional features. Having a characteristic of using linear equations seemed to require having core variables that strongly affect the performance of the model. The approach was very different from previous ML models and therefore did not perform as well initially, however, adding the Twitter features showed increased stability and performance by creating an efficient linear boundary. As our data is highly correlated, many attributes were dependent on each other, and the classification of instances were highly nonlinear. By experimenting on different ML models, we learned the mechanism of the algorithms, how they differentiated from each other, and the specific strengths and weaknesses of the models.

10 Acknowledgement

This project was supervised by Professor Anita Raja, Machine Learning (CSCI 79502) at Hunter College. And the project title ‘Look Before You Leap’ provided by Isaac Lapides.

11 References

[1] U.S. Census \Sex Of Workers By Means Of Transportation To Work For Workplace Geography\, https://data.census.gov/cedsci/table?q=ACSDT1Y2017.B08406&g=1600000US3651000&tid=ACSDT1Y2017.B08406

[2] Stokols, D., Novaco, R. W., Stokols, J., & Campbell, J. (1978). Traffic congestion, Type A behavior, and stress. *Journal of Applied Psychology, 63*(4), 467–480. https://doi.org/10.1037/0021-9010.63.4.467

[3] Zhang, K., & Batterman, S. (2013). Air pollution and health risks due to vehicle traffic. *The Science of the total environment*, *450-451*, 307–316. https://doi.org/10.1016/j.scitotenv.2013.01.074

[4] Sakaki, Takeshi & Okazaki, Makoto & Matsuo, Yutaka. (2010). Earthquake Shakes Twitter Users: Real-Time Event Detection by Social Sensors. Proceedings of the 19th International Conference on World Wide Web, WWW '10. 851-860. 10.1145/1772690.1772777.

[5] Moosavi, Sobhan, Mohammad Hossein Samavatian, Srinivasan Parthasarathy, and Rajiv Ramnath. (2019) "A Countrywide Traffic Accident Dataset."