CAR ACCIDENT SEVERITY (WEEK 2)

Capstone Project Report

KUNAL DAS

Introduction

The economic and societal impact of traffic accidents cost U.S. citizens hundreds of billions of dollars every year. And a large part of losses is caused by a small number of serious accidents. Reducing traffic accidents, especially serious accidents, is nevertheless always an important challenge. The proactive approach, one of the two main approaches for dealing with traffic safety problems, focuses on preventing potential unsafe road conditions from occurring in the first place. For the effective implementation of this approach, accident prediction and severity prediction are critical. If we can identify the patterns of how these serious accidents happen and the key factors, we might be able to implement well-informed actions and better allocate financial and human resources.

Objectives

The first objective of this project is to recognize key factors affecting the accident severity. The second one is to develop a model that can accurately predict accident severity. To be specific, for a given accident, without any detailed information about itself, like driver attributes or vehicle type, this model is supposed to be able to predict the likelihood of this accident being a severe one. The accident could be the one that just happened and still lack of detailed information, or a potential one predicted by other models. Therefore, with the sophisticated real-time traffic accident prediction solution developed by the creators of the same dataset used in this project, this model might be able to further predict severe accidents in real-time.

Dataset Overview

This is a countrywide car accident dataset, which covers **49 states of the USA**. The accident data are collected from **February 2016 to June 2020**, using two APIs that provide streaming traffic incident (or event) data. These APIs broadcast traffic data captured by a variety of entities, such as the US and state departments of transportation, law enforcement agencies, traffic cameras, and traffic sensors within the road-networks. Currently, there are about **3.5 million** accident records in this dataset. Check here to learn more about this dataset.

Acknowledgements

Please cite the following papers if you use this dataset:

- Moosavi, Sobhan, Mohammad Hossein Samavatian, Srinivasan Parthasarathy, and Rajiv Ramnath. “A Countrywide Traffic Accident Dataset.”, 2019.
- Moosavi, Sobhan, Mohammad Hossein Samavatian, Srinivasan Parthasarathy, Radu Teodorescu, and Rajiv Ramnath. "Accident Risk Prediction based on Heterogeneous Sparse Data: New Dataset and Insights." In proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, ACM, 2019.

Content

This dataset has been collected in real-time, using multiple Traffic APIs. Currently, it contains accident data that are collected from February 2016 to June 2020 for the Contiguous United States.
Check here to learn more about this dataset.

Inspiration

US-Accidents can be used for numerous applications such as real-time car accident prediction, studying car accidents hotspot locations, casualty analysis and extracting cause and effect rules to predict car accidents, and studying the impact of precipitation or other environmental stimuli on accident

occurrence. The most recent release of the dataset can also be useful to study the impact of COVID-19 on traffic behavior and accidents.

Usage Policy and Legal Disclaimer

This dataset is being distributed only for **Research** purposes, under Creative Commons Attribution-Noncommercial-ShareAlike license (CC BY-NC-SA 4.0). By clicking on download button(s) below, you are agreeing to use this data only for non-commercial, research, or academic applications. You may need to cite the above papers if you use this dataset.

Link for Kaggle dataset: https://www.kaggle.com/sobhanmoosavi/us-accidents/

Dataset Features

We can find 5 different types of attributes in the dataset. They are Traffic Attributes, Address Attributes, Weather attributes, Point-Of-Interest (POI) attributes and Period of Day attributes. Detailed attributes are given below:

1. Traffic Attributes (12):
    1. ID: This is a unique identifier of the accident record.
    2. Source: Indicates source of the accident report (i.e. the API which reported the accident.).
    3. TMC: A traffic accident may have a Traffic Message Channel (TMC) code which provides more detailed description of the event.
    4. Severity: Shows the severity of the accident, a number between 1 and 4, where 1 indicates the least impact on traffic (i.e., short delay as a result of the accident) and 4 indicates a significant impact on traffic (i.e., long delay).
    5. Start_Time: Shows start time of the accident in local time zone.
    6. End_Time: Shows end time of the accident in local time zone.
    7. Start_Lat: Shows latitude in GPS coordinate of the start point.
    8. Start_Lng: Shows longitude in GPS coordinate of the start point.
    9. End_Lat: Shows latitude in GPS coordinate of the end point.
    10. End_Lng: Shows longitude in GPS coordinate of the end point.
    11. Distance(mi): The length of the road extent affected by the accident.
    12. Description: Shows natural language description of the accident.
2. Address Attributes (9):
    1. Number: Shows the street number in address field.
    2. Street: Shows the street name in address field.
    3. Side: Shows the relative side of the street (Right/Left) in address field.
    4. City: Shows the city in address field.

5. County: Shows the county in address field.
6. State: Shows the state in address field.
7. Zipcode: Shows the zipcode in address field.
8. Country: Shows the country in address field.
9. Timezone: Shows timezone based on the location of the accident (eastern, central, etc.).

3. Weather Attributes (11):

1. Airport_Code: Denotes an airport-based weather station which is the closest one to location of the accident.
2. Weather_Timestamp: Shows the time-stamp of weather observation record (in local time).
3. Temperature(F): Shows the temperature (in Fahrenheit).
4. Wind_Chill(F): Shows the wind chill (in Fahrenheit).
5. Humidity(%): Shows the humidity (in percentage).
6. Pressure(in): Shows the air pressure (in inches).
7. Visibility(mi): Shows visibility (in miles).
8. Wind_Direction: Shows wind direction.
9. Wind_Speed(mph): Shows wind speed (in miles per hour).
10. Precipitation(in): Shows precipitation amount in inches, if there is any.
11. Weather_Condition: Shows the weather condition (rain, snow, thunderstorm, fog, etc.).

4. POI Attributes (13):

1. Amenity: A Point-Of-Interest (POI) annotation which indicates presence of amenity in a nearby location.
2. Bump: A POI annotation which indicates presence of speed bump or hump in a nearby location.
3. Crossing: A POI annotation which indicates presence of crossing in a nearby location.
4. Give_Way: A POI annotation which indicates presence of give_way sign in a nearby location.
5. Junction: A POI annotation which indicates presence of junction in a nearby location.
6. No_Exit: A POI annotation which indicates presence of no_exit sign in a nearby location.
7. Railway: A POI annotation which indicates presence of railway in a nearby location.
8. Roundabout: A POI annotation which indicates presence of roundabout in a nearby location.
9. Station: A POI annotation which indicates presence of station (bus, train, etc.) in a nearby location.
10. Stop: A POI annotation which indicates presence of stop sign in a nearby location.
11. Traffic_Calming: A POI annotation which indicates presence of traffic_calming means in a nearby location.
12. Traffic_Signal: A POI annotation which indicates presence of traffic_signal in a nearby location.
13. Turning_Loop: A POI annotation which indicates presence of turning_loop in a nearby location.

5. Period-of-Day (4):

    1. Sunrise_Sunset: Shows the period of day (i.e. day or night) based on sunrise/sunset.

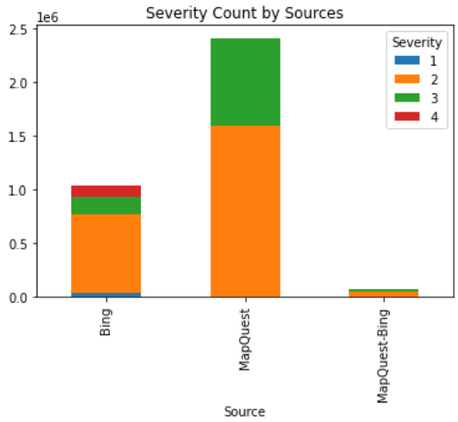
    2. Civil_Twilight: Shows the period of day (i.e. day or night) based on civil twilight.

    3. Nautical_Twilight: Shows the period of day (i.e. day or night) based on nautical twilight.

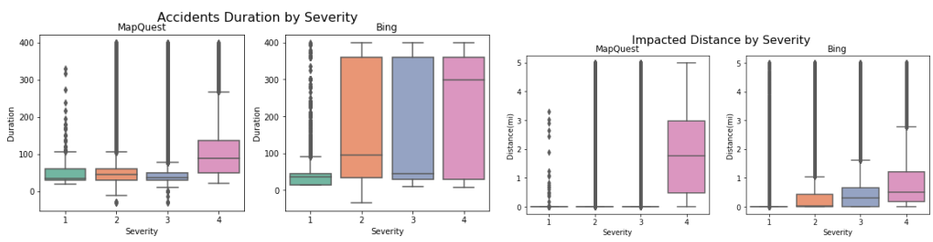
    4. Astronomical_Twilight: Shows the period of day (i.e. day or night) based on astronomical twilight.

Reporting Sources

These data came from two sources, MapQuest and Bing, both of which report severity level but in a different way. Bing has 4 levels while MapQuest has 5. And according to dataset creator, there is no way to do a 1:1 mapping between them. Since severity is what we really care about in this project, I think it is crucial to figure out the difference.



The stacked bar chart shows that two data providers reported totally different proportions of accidents of each level. MapQuest reported so rare accidents with severity level 4 which cannot even be seen in the plot, whereas Bing reported almost the same number of level 4 accidents as level 2. Meanwhile, MapQuest reported much more level 3 accidents than Bing in terms of proportion. These differences may be due to the different kinds of accidents they tend to collect or the different definitions of severity level, or the combination of them. If the latter is the case, I do not think we can use the data from both at the same time. To check it out, we can examine the distribution of accidents with different severity levels across two main measures, Impacted Distance and Duration.



Two differences are obvious in the above plots. The first is that the overall duration and impacted distance of accidents reported by Bing are much longer than those by MapQuest. Second, same severity level holds different meanings for MapQuest and Bing. MapQuest seems to have a clear and strict threshold for severity level 4, cases of which nevertheless only account for a tiny part of the whole dataset. Bing, on the other hand, does not seem to have a clear-cut threshold, especially regards duration, but the data is more balanced.

It is hard to choose one and we cannot use both. I decided to select MapQuest because serious accidents are, we really care about and the sparse data of such accidents is the reality we must confront.

Finally, drop data reported from Bing and 'Source' column.

Useless Features

Features 'ID' doesn't provide any useful information about accidents themselves. 'TMC', 'Distance(mi)', 'End_Time' (we have start time), 'Duration', 'End_Lat', and 'End_Lng'(we have start location) can be collected only after the

accident has already happened and hence cannot be predictors for serious accident prediction. For 'Description', the POI features have already been extracted from it by dataset creators. Let's get rid of these features first.

```
Unique count of categorical features:
Side 3
Country 1
Timezone 5
Amenity 2
Bump 2
Crossing 2
Give_Way 2
Junction 2
No_Exit 2
Railway 2
Roundabout 2
Station 2
Stop 2
Traffic_Calming 2
Traffic_Signal 2
Turning_Loop 1
Sunrise_Sunset 3
Civil_Twilight 3
Nautical_Twilight 3
Astronomical_Twilight 3
```

Drop 'Country' and 'Turning_Loop' for they have only one class.

Clean Up Categorical Features

If we look at categorical features closely, we will find some chaos in 'Wind_Direction' and 'Weather_Condition'. It is necessary to clean them up first.

Wind_Direction is modified by dropping 'Country' and 'Turning_Loop' for they have only one class.
Weather-related vehicle accidents kill more people annually than large-scale weather disasters(source: weather.com). According to Road Weather Management Program, most weather-related crashes happen on wet-pavement and during rainfall. Winter-condition and fog are another two main reasons for weather-related accidents. To extract these three weather conditions, we first look at what we have in 'Weather_Condition' Feature.
Weather_condition is modified using the most common ones.
Since the 'Weather_Timestamp' is almost as same as 'Start_Time', we can just keep 'Start_Time'. Then map 'Start_Time' to 'Year', 'Month', 'Weekday', 'Day' (in a year), 'Hour', and 'Minute' (in a day).
Date and Time Format is fixed.

Drop NaN

The counts of missing values in some features are much smaller compared to the total sample. It is convenient to drop rows with missing values in these columns.

Drop NAs by these features:

'City'
'Zipcode'
'Airport_Code'
'Sunrise_Sunset'
'Civil_Twilight'
'Nautical_Twilight'
'Astronomical_Twilight'
Continuous Weather Data

Continuous weather features with missing values:
Temperature(F)
Humidity(%)
Pressure(in)
Visibility(mi)
Wind_Speed(mph)

Before imputation, weather features will be grouped by location and time first, to which weather is naturally related. 'Airport_Code' is selected as location feature because the sources of weather data are airport-based weather stations. Then the data will be grouped by 'Start_Month' rather than 'Start_Hour' because using the

former is computationally cheaper and remains less missing values. Finally, missing values will be replaced by median value of each group.

```
The number of remaining missing values:
Temperature(F) : 4899
Humidity(%) : 4924
Pressure(in) : 4865
Visibility(mi) : 11084
Wind_Speed(mph) : 11257
```

There still are some missing values but much less. Just dropna by these features for the sake of simplicity.

Categorical Weather Features

For categorical weather features, majority rather than median will be used to replace missing values.

```
Count of missing values that will be dropped:
Wind_Direction : 7999
Clear : 10507
Cloud : 11276
Rain : 9490
Heavy_Rain : 8939
Snow : 8963
Heavy_Snow : 8932
Fog : 8955
```

EXPLORATION & ENGINEERING

Resampling

Based on the exploration we did in 1.2, the accidents with severity level 4 are much more serious than accidents of other levels, between which the division is far from clear-cut. Therefore, I decided to focus on level 4 accidents and regroup the levels of severity into level 4 versus other levels.

```
df['Severity4'] = 0
df.loc[df['Severity'] == 4, 'Severity4'] = 1
df.Severity4.value_counts()
```

```
0    2377735
1       6615
Name: Severity4, dtype: int64
```

As seen from above, the data is so unbalanced that we can hardly do exploratory analysis. To address this issue, the combination of over- and under-sampling will be used since the dataset is large enough. level 4 will be randomly oversampled to 100000 and other levels will be randomly undersampled to 100000.

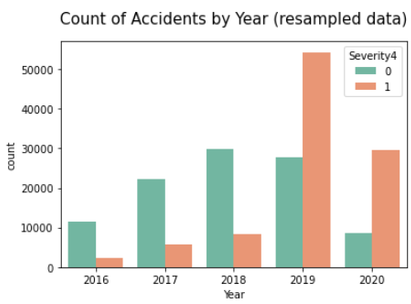
```
df = df.drop(['Severity'], axis = 1)
df_bl = pd.concat([df[df['Severity4']==1].sample(100000, replace = True),
                   df[df['Severity4']==0].sample(100000)], axis=0)
print('resampled data:', df_bl.Severity4.value_counts())
```

```
resampled data: 1    100000
0    100000
Name: Severity4, dtype: int64
```

Then we can do some exploratoty analysis on resampled data.

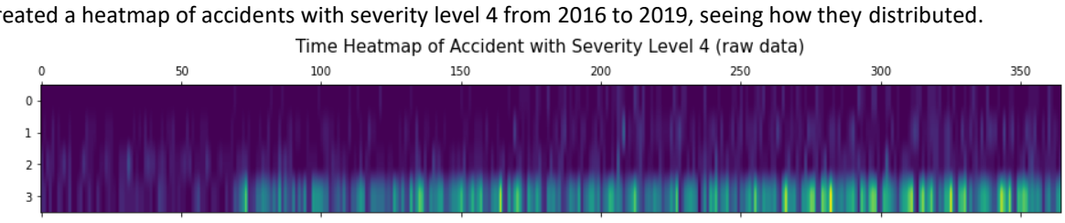
Time Features

Year



There must be something wrong. It is impossible that the number of accidents with severity level 4 in 2019 is more than 5 times the number in 2018 while the number of other levels accidents is less. Let us back to raw data to have a look.

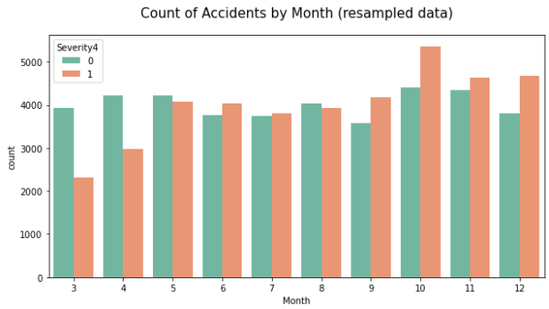
I created a heatmap of accidents with severity level 4 from 2016 to 2019, seeing how they distributed.



The heatmap indicates that something changed after Feb 2019. Maybe it is the way that MapQuest defines severity or the way they collect data. Anyway, we must narrow down our data again. Since the data after Feb 2019 is less imbalanced and the data in the future is more likely to look like this, dropping the data before Mar 2019 may be the best choice.

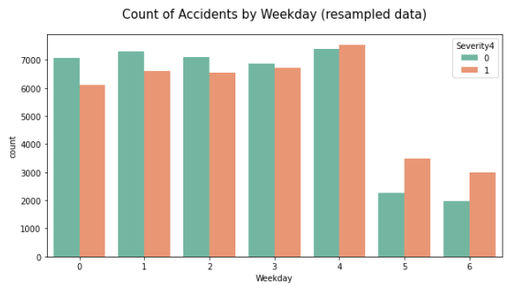
Month

It is quite interesting that the count of other levels accidents is mostly consistent from March to December, whereas the number of level 4 accidents rapidly increased from March to May and remained stable until September then increased again from October.



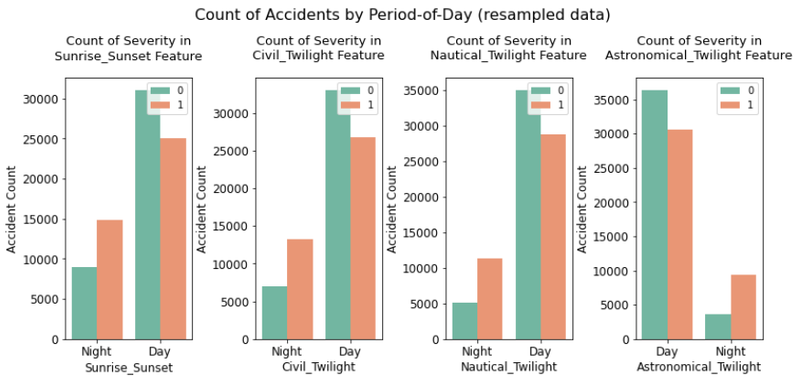
Weekday

The number of accidents was much less on weekends while the proportion of level 4 accidents was higher.



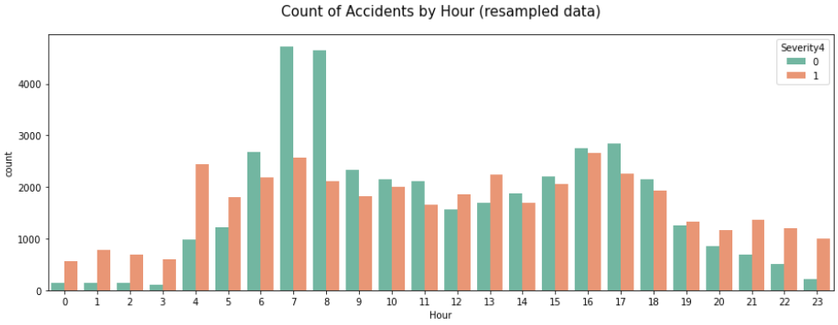
Period-of-Day

Accidents were less during the night but were more likely to be serious.



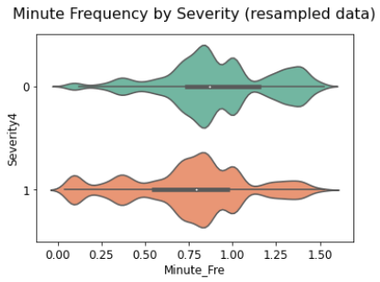
Hour

Most accidents happened during the daytime, especially AM peak and PM peak. When it comes to night, accidents were far less but more likely to be serious.



Frequence Encoding (Minute)

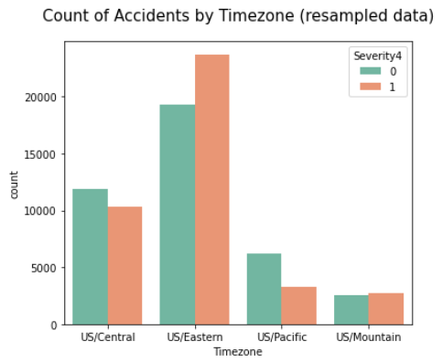
As seen in the plot of 'Hour', 'Minute' may also be an important predictor. But directly using it would produce an overabundance of dummy variables. Therefore, the frequency of 'Minute' was utilized as labels, rather than 'Minute' itself. To normalize the distribution, the frequency was also transformed by log.



Address Features

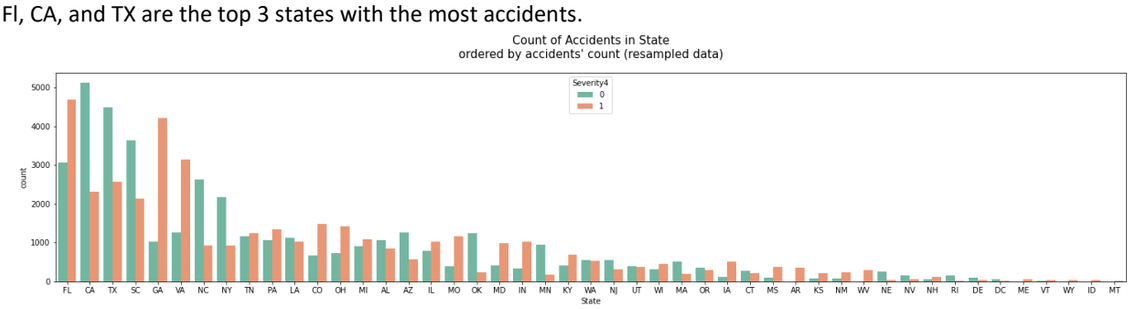
Timezone

Eastern time zone is the most dangerous one.

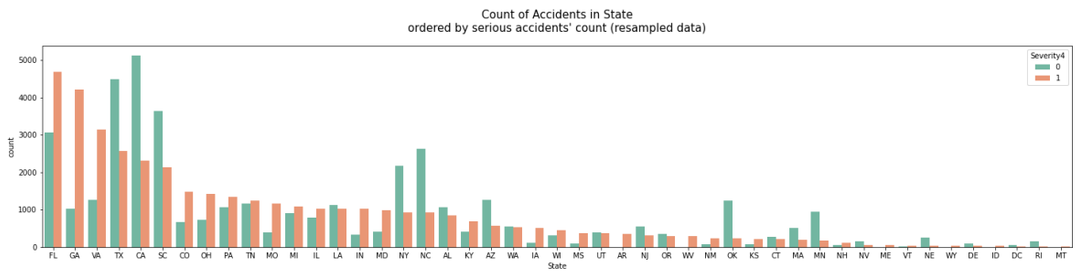


State

Fl, CA, and TX are the top 3 states with the most accidents.



It is a different story if we order the plot by the count of accidents with severity of level 4. FL is still the top one but the next two are GA and VA.



County

There are too many counties that we cannot visualize them as we did for states. But we do can incorporate census data for them.

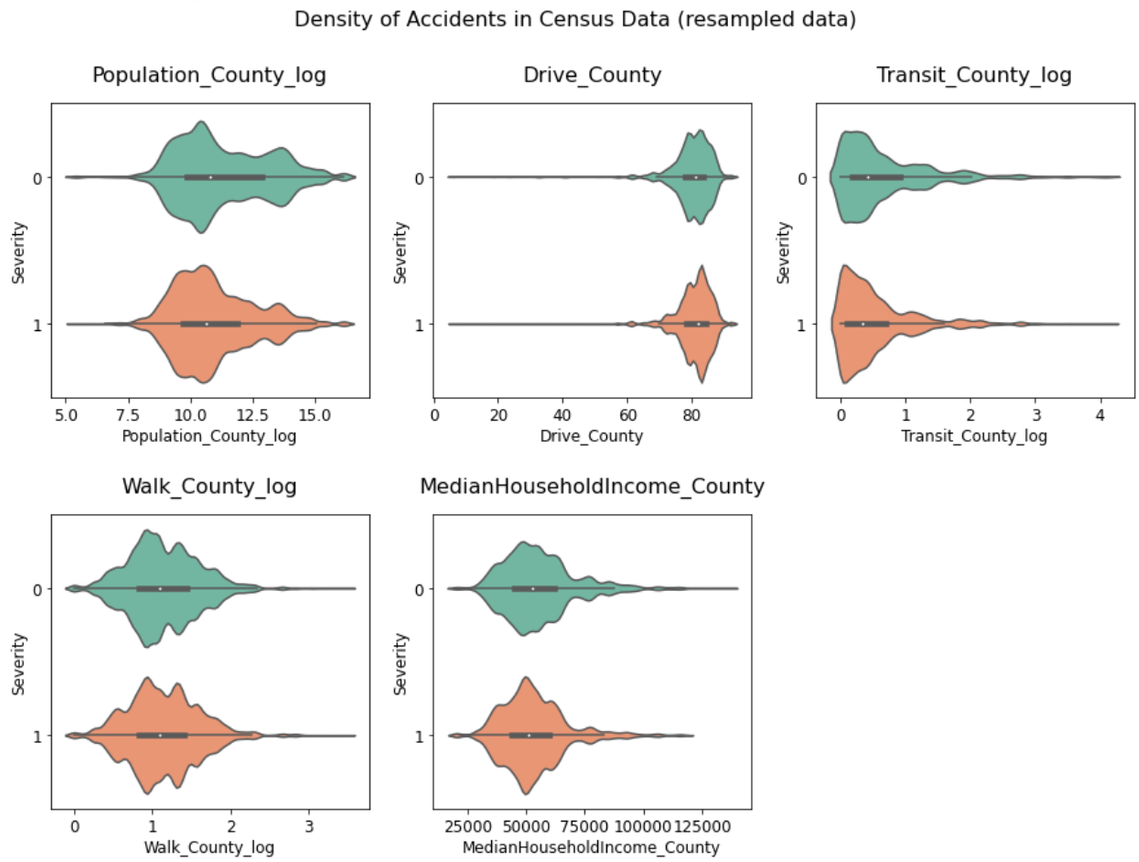
Several basic variables, like total population, percent of commuters who drive, take transit or walk to work, and median household income, for all counties were downloaded from ACS 5-year estimates 2018. Then, counties' names were isolated.

	index	Population_County	Drive_County	Transit_County	Walk_County	MedianHouseholdIncome_County	County_y
0	Washington County, Mississippi: Summary level:...	47086	86.4	0.0	1.3	30834	washington
1	Perry County, Mississippi: Summary level: 050,...	12028	85.8	0.0	1.8	39007	perry
2	Choctaw County, Mississippi: Summary level: 05...	8321	85.6	0.3	1.1	37203	choctaw
3	Itawamba County, Mississippi: Summary level: 0...	23480	82.4	0.2	0.7	40510	itawamba
4	Carroll County, Mississippi: Summary level: 05...	10129	90.0	0.0	1.4	43060	carroll

Counties' names turned out to be very tricky. Converting all of them into lowercase is not enough. Some counties name in USA-accidents omit "city" or "parish", and hence cannot be matched with names in census data. We need to manually put them back and re-join them.

```
Count of missing values before:  index                        17553
Population_County               17553
Drive_County                    17553
Transit_County                  17553
Walk_County                     17553
MedianHouseholdIncome_County    17553
dtype: int64
Count of missing values after:  index                         5134
Population_County                5134
Drive_County                     5134
Transit_County                   5134
Walk_County                      5134
MedianHouseholdIncome_County     5134
dtype: int64
```

Drop na and use Logit transformation on some variables having extremely skewed distribution.

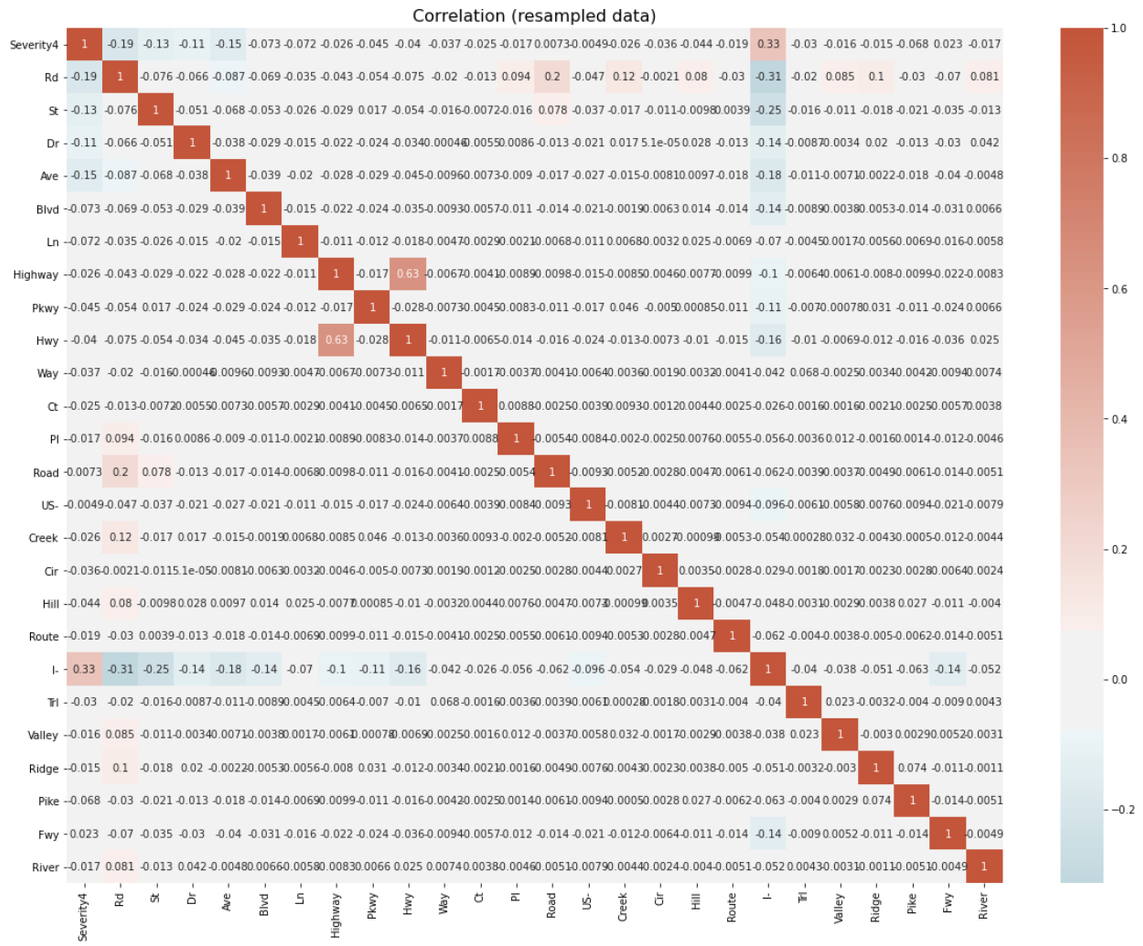


Percent of people taking transit to commute seems to related to severity. Level 4 accidents happened more frequently in those counties with a lower usage rate of transit.

Street

There are more and more studies found that higher speed limits were associated with an increased likelihood of crashes and deaths. (https://www.cga.ct.gov/2013/rpt/2013-R-0074.htm) And speed limits are highly related to street type. Street type hence can be a good predictor of serious accidents. There is no feature about street type in the original dataset though, we can extract it from the street name.

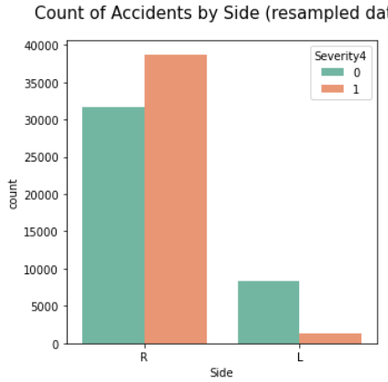
The top 40 most common words in street names were selected. This list contains not only street types but also some common words widely used in street names.



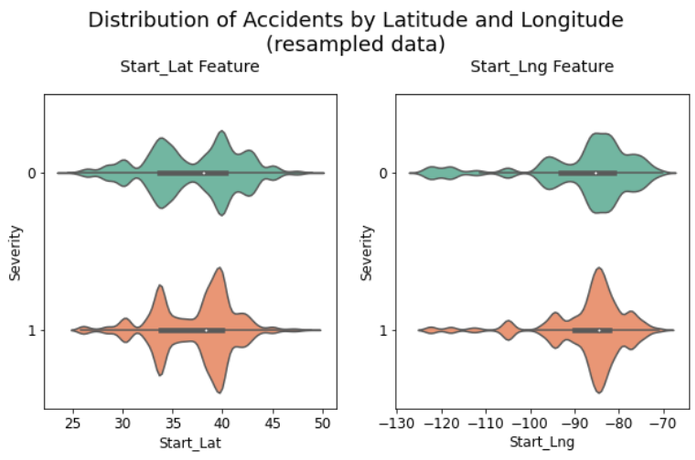
Interstate Highway turns out to be the most dangerous street. Other roads like basic road, street, drive, and avenue are relatively safe. Let us just keep these five features.

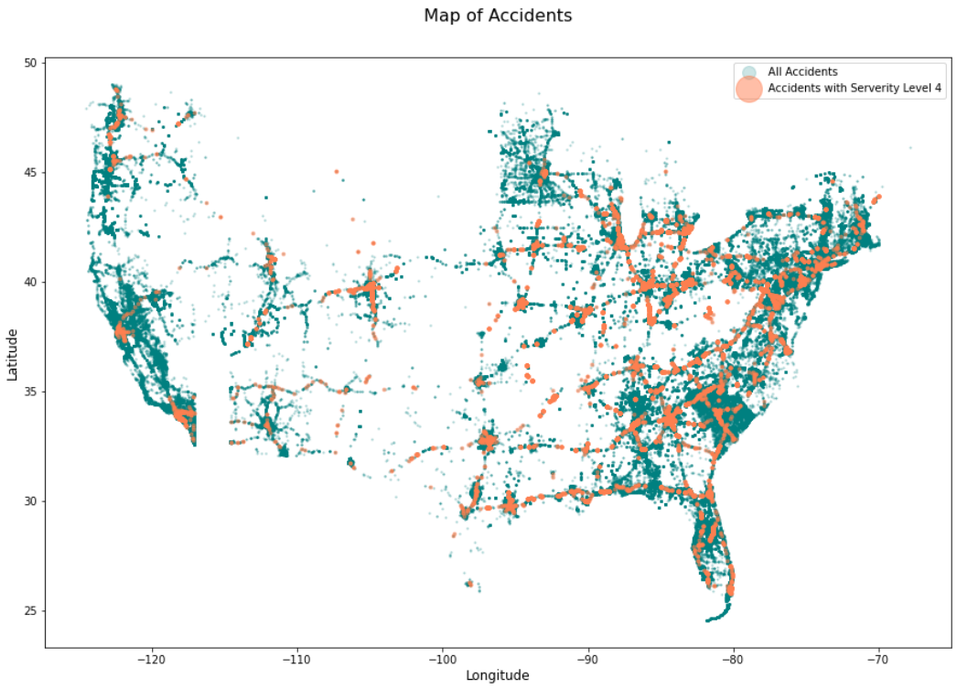
Side

Right side of the line is much more dangerous than left side.



Latitude and Longitude

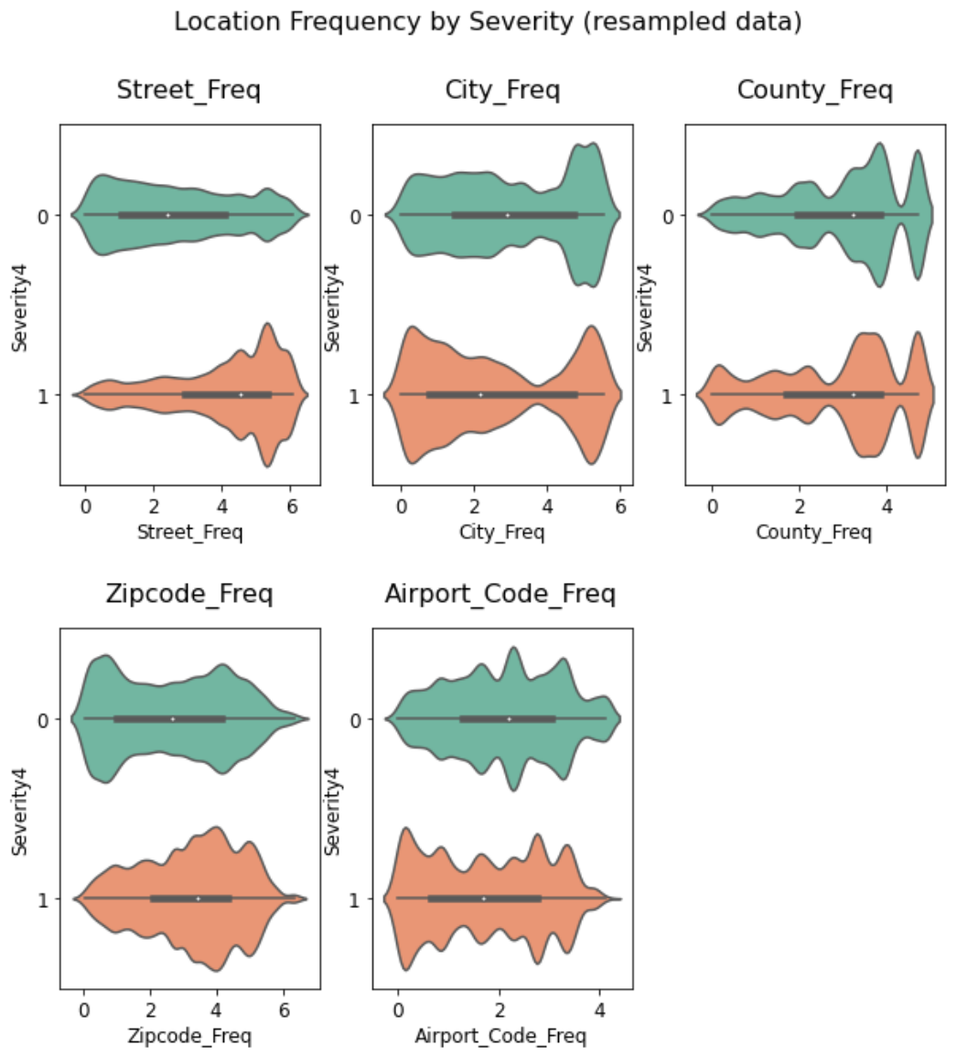




Frequency Encoding

Like 'Minute', some location features like 'City' and 'Zipcode' that have too many unique values can be labelled by their frequency. Frequency encoding and log-transform:

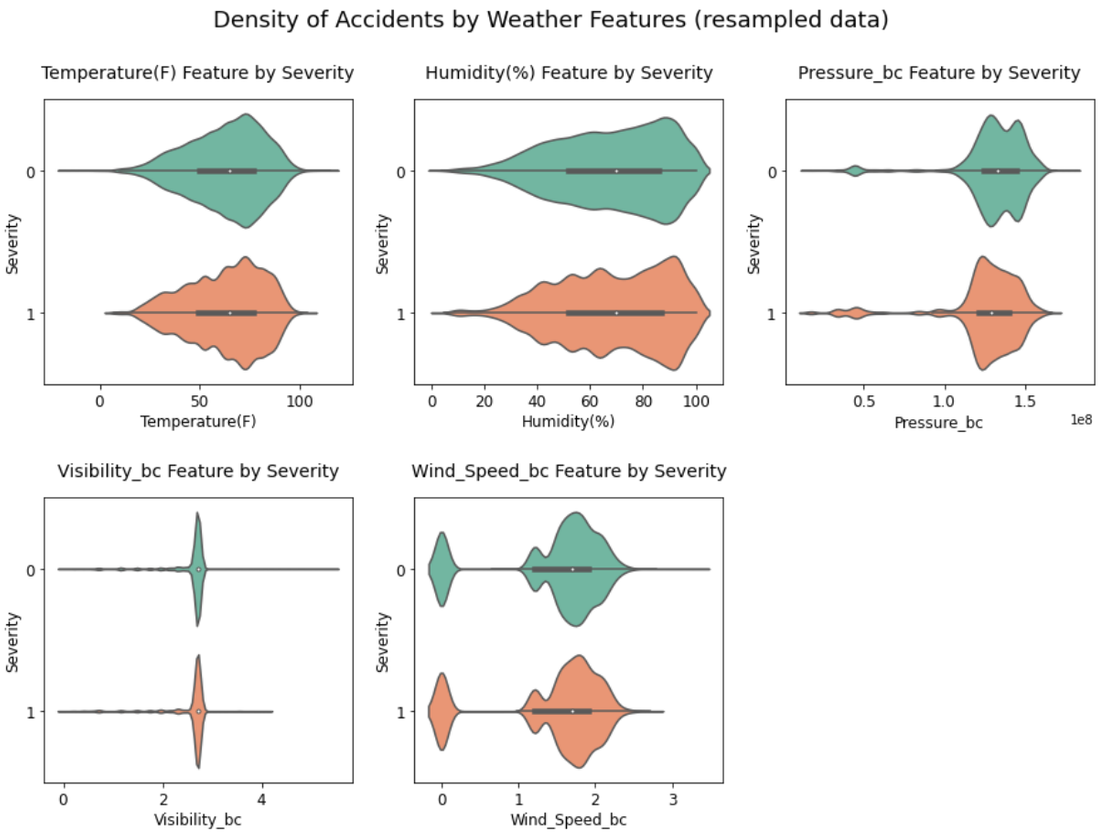
'Street'
'City'
'County'
'Zipcode'
'Airport_Code'



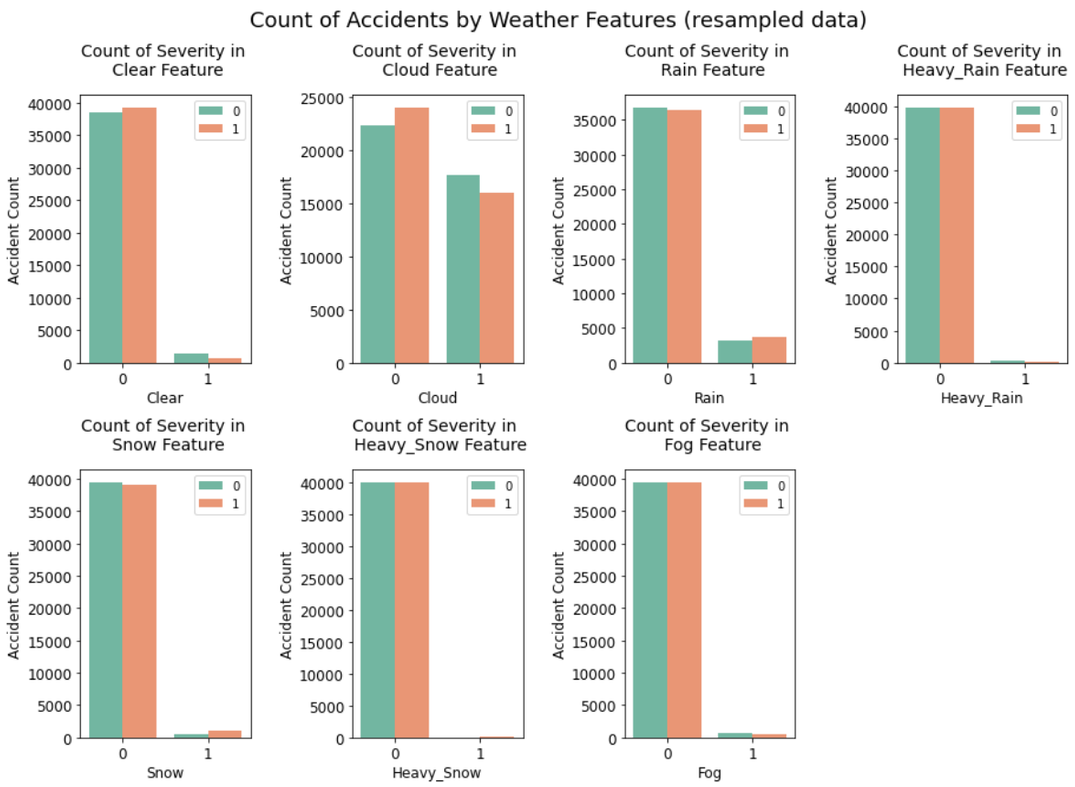
Two opposite patterns can be identified in these plots. For 'Street' and 'Zipcode', higher frequency means higher likelihood of being a serious accident. In contrast with these smaller regions, for 'City' and 'Airport_Code' instead, higher frequency means less likelihood of being a serious accident. Get rid of features we do not need anymore.

Weather Features

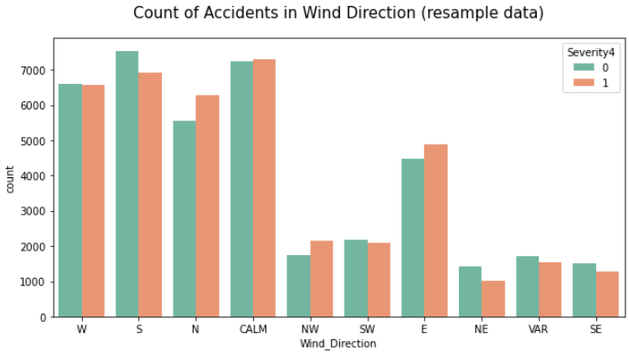
Continuous Weather Features
Normalize features with extremely skewed distribution first.



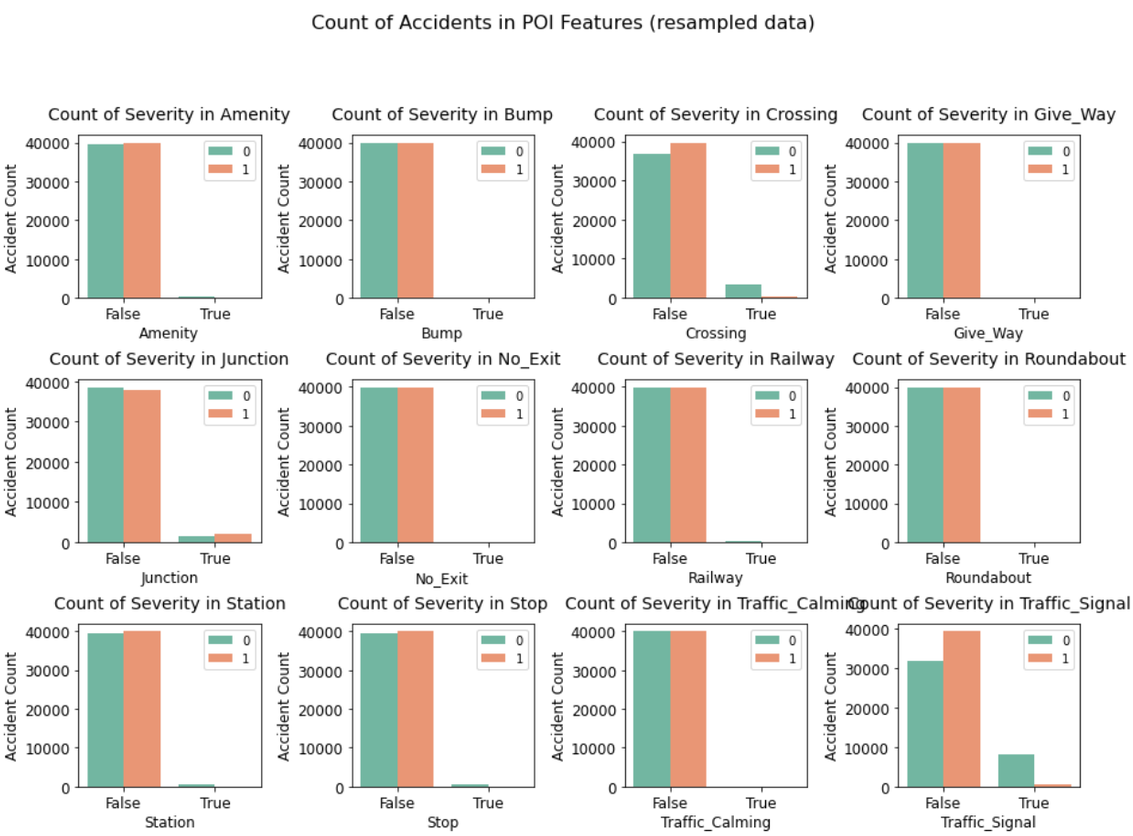
Weather Conditions



As seen from above, accidents are little more likely to be serious during rain or snow while less likely on a cloudy day.



POI Features



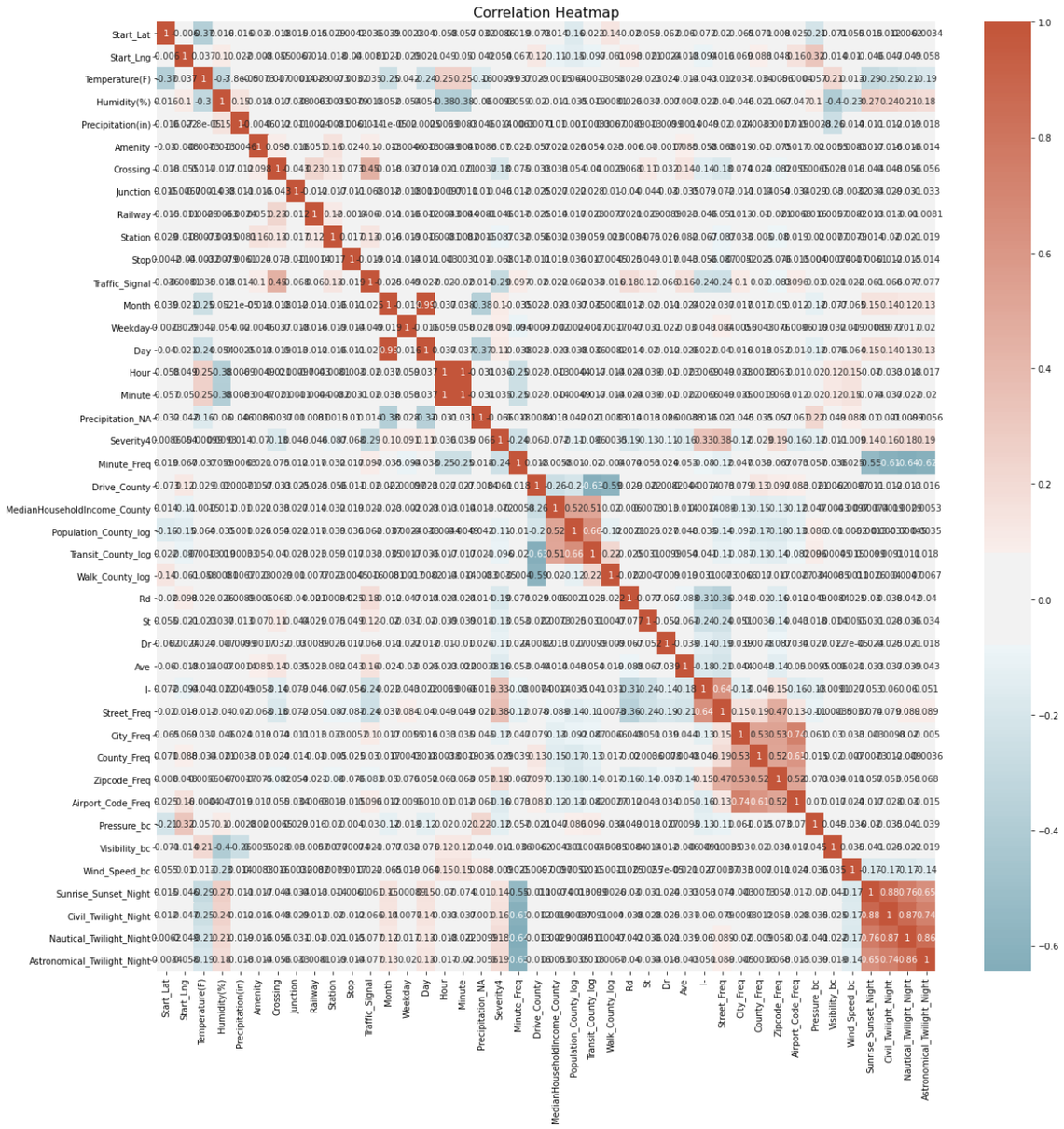
Accidents near traffic signal and crossing are much less likely to be serious accidents while little more likely to be serious if they are near the junction. Maybe it is because people usually slow down in front of crossing and traffic signal, but junction and severity are highly related to speed. Other POI features are so unbalanced that it is hard to tell their relationship with severity from plots.

Drop some features:

'Bump'
'Give_Way'
'No_Exit'
'Roundabout'
'Traffic_Calming'

Correlation

Features are not highly correlated with each other. The top 3 highest correlations between severity and features are -0.17 (traffic_signal), 0.15 (Start_Lng), 0.12 (Start_Lat).



The above figure shows strong positive correlations between 'Severity4' (level 4 severity) with 'I-' (Interstate highway) as well as 'Street_Freq', and strong negative correlation between it with 'Traffic_signal' and 'Minute_Freq'.

We can also identify several highly colinear features, such as 'Day'-'Month', 'Minute'-'Hour', 'Transit_County_log'-'Population_County_log', 'Airport_Code_Freq'-'City_Freq', and four period-of-day features. Let's drop some of them.

drop features:

'Day'
'Minute'
'Population_County_log'
'City_Freq'
'Civil_Twilight_Night'
'Nautical_Twilight_Night'

Model

Train Test Split

Resample data and split it into X and y.
Standardize features based on unit variance.
Split data into X_train, X_test, y_train, and y_test. The size of training data is about 64000 and the test is about 16000.

Logistic Regression

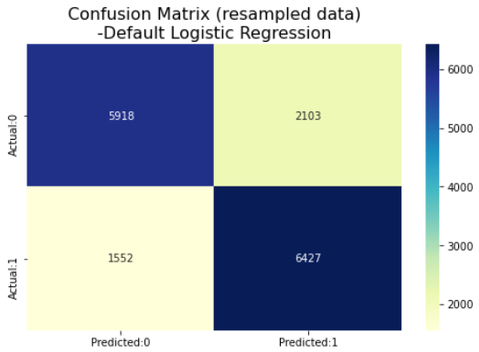
Logistic regression was emploied as a baseline to perform binary classification task.

```
# Logistic regression with default setting.
from sklearn.linear_model import LogisticRegression

clf = LogisticRegression(max_iter=10000,random_state=42)
clf.fit(X_train, y_train)

accuracy_train = clf.score(X_train, y_train)
print("Train Accuracy: %.1f%%"% (accuracy_train*100))
accuracy_test = clf.score(X_test,y_test)
print("Test Accuracy: %.1f%%"% (accuracy_test*100))
```

```
Train Accuracy: 77.3%
Test Accuracy: 77.2%
```




```
Train Accuracy: 77.3%
Test Accuracy: 77.1%
Wall time: 1.3 s
```

Even with the best parameter setting, logistic regression yielded very poor results for both training data and test data.

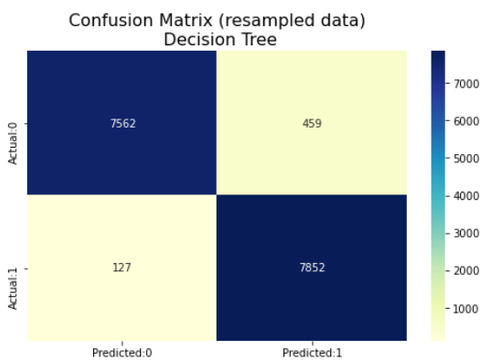
Decision Tree

Then, decision tree classifier was employied. The grid search was performed over choices of 'min_samples_split': {5,10, 20, 30, 40}, 'max_features': {None, 'log2', 'sqrt'}

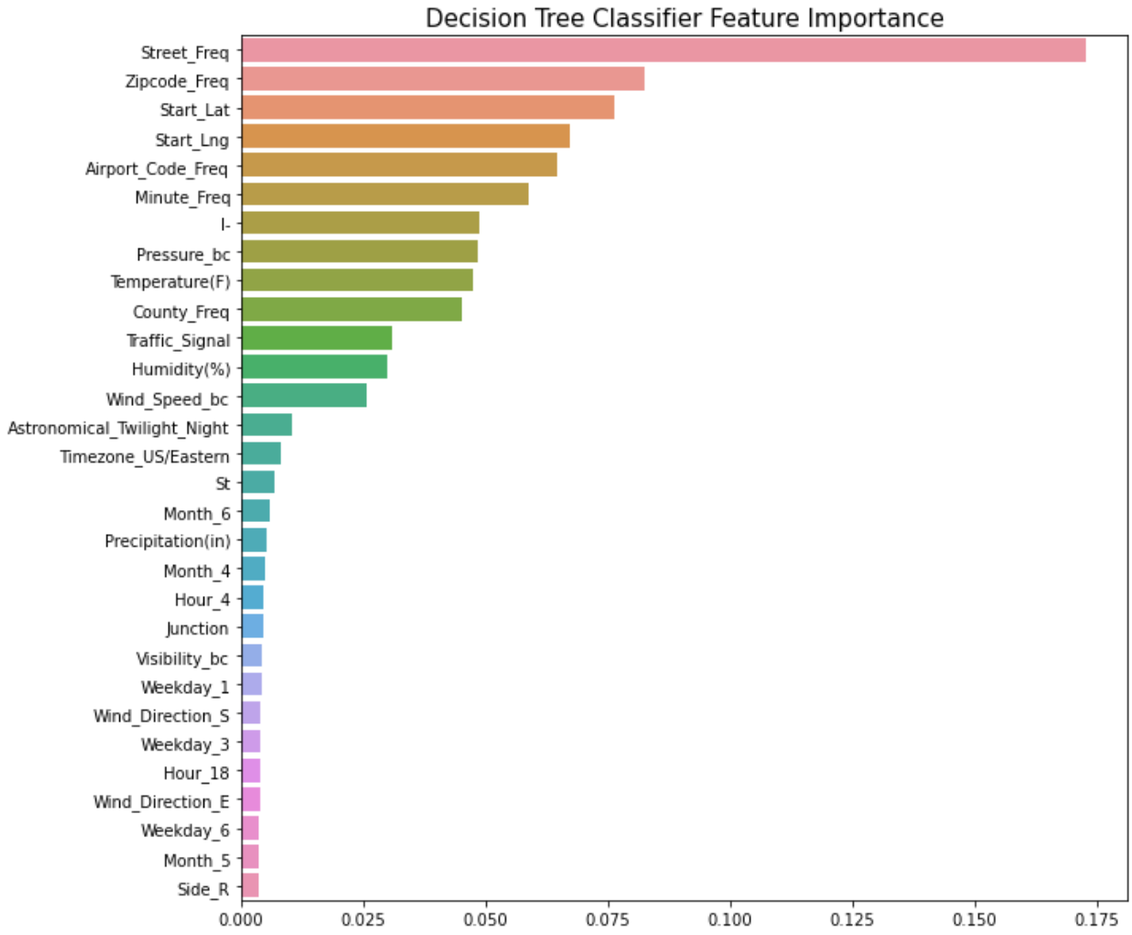
```
%%time
from sklearn import tree
# Training step, on X_train with y_train
tree_clf = tree.DecisionTreeClassifier(min_samples_split = 5)
tree_clf = tree_clf.fit(X_train,y_train)

tree_accuracy_train = tree_clf.score(X_train, y_train)
print("Train Accuracy: %.1f%%"% (tree_accuracy_train*100))
tree_accuracy_test = tree_clf.score(X_test,y_test)
print("Test Accuracy: %.1f%%"% (tree_accuracy_test*100))
```

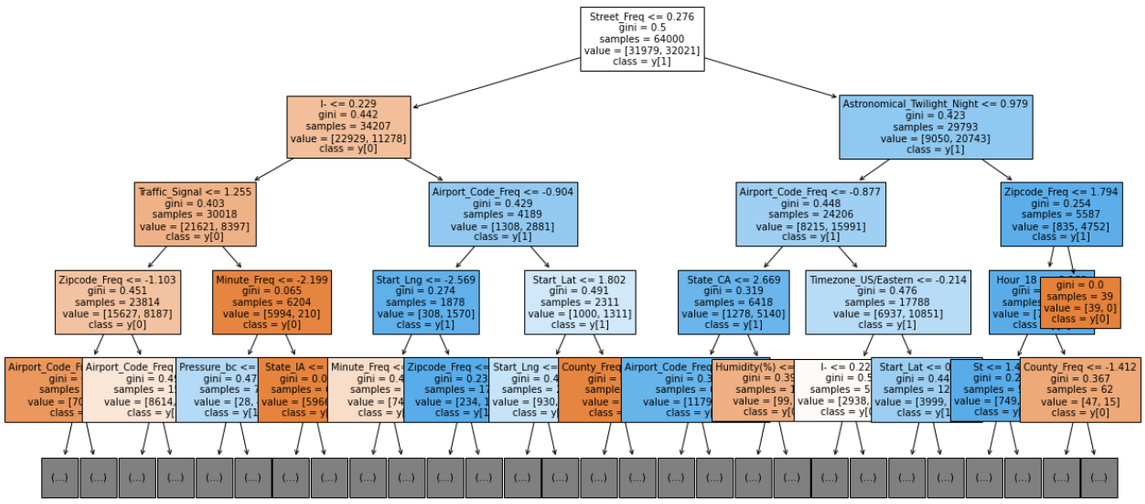
```
Train Accuracy: 99.7%
Test Accuracy: 96.3%
Wall time: 3.12 s
```



By using decision tree classifier, the training accuracy improved to 99.8% and test accuracy to 96.0%. The result is almost perfect.



The feature importance plot shows that high-resolution spatio-temporal patterns of accidents are the most useful features to predict severity. Among them, street frequency is far more important than any other feature. In addition to these spatio-temporal features, weather features like pressure, temperature, humidity, and wind speed are also very important. Some other features like interstate highway('I-'), traffic signal are important as well.



Future Work

Random Forest can be applied to check how well the model is trained.
Incorporate this model in a real-time accident risk prediction model or develop a new real-time severe accident risk prediction on grid cells.
Detailed relations between some key factors and accident severity can be further studied.
Policy implications of this project can be explored