MPMD 3.1: Data Science and Project Management Lab

Team 1 – Crashes – Final Report

From
Lin/Yihsuan (Shelly)/s0572048
Poboon/Pawin (Tony)/s0572079
King/Catherine/s0572057
Ladhwe/Shantanu/s0572139
Yolagel/Ender/s0572050

Department 3
Project Management and Data Science (MPMD)

Supervisor: Noa Tamir

1. Business Understanding

1.1. Project introduction

All people in Chicago have the right to utilize the road, public transportation regardless of wherever they live. However, every 3 days, someone is killed in a traffic accident and 5 people are seriously injured. In total, there were 554 people were killed in between 2010-2014 and 9,480 people were seriously injured due to a traffic accident. The major consequences of this include, but are not limited to, personal economic costs, personal trauma, taxpayer spending on emergency services and infrastructure repair, as well as an overall inequality of mobility.

With this background, Chicago Mayor Emanuel and the steering committees have committed to investigate the crash records with the hope of identifying the contributory causes of traffic-related deaths and injuries, and with the goal of reducing traffic fatalities and serious injuries by 35% by 2020 and completely eliminating them by 2026.

By participating the nonprofit Project Vision Zero, started in 1990s in Sweden, the city of Chicago is hopefully on its way to increasing its mobility safety level.

1.2. First client meeting

After meeting with the client, Levan, the project goals and requirements have been established as the following:

- One of the deliverables should include a reproduceable, reusable script which can be used on future data.
- The number of identified factors should remain small in order to give the client a manageable action plan (no more than 5 factors at first)
- One of the deliverables should include an easily understood dashboard, which can serve as educational material. The client is open to both of the following:
    - One dashboard for the stakeholders/governmental bodies and one dashboard for the general public
    - One dashboard for both stakeholders/governmental bodies and the general public

With these goals and requirements in mind, our team will use the CRISP-DM methodology to create a machine learning model which identifies the key clusters of factors and discover any patterns in order to present the client with a proposed action plan, as well as provide the general public of Chicago with educational material.

The findings of our exploration and model(s) can help to form recommendations or guidance for diverse group of stakeholders including transportation professionals, policymakers, public health officials, police and community members.

1.3. Second client meeting

We met a second time with Levan to review our intermediate findings and discuss our potential next steps. The findings we discussed can be seen in the following sections. Important to note at this point are some of the questions and feedbacks from Levan -- we are hoping this can be helpful when you meet with him moving forward:

- Levan appreciated the focus on community education and hopes to continue to see this
- Levan found the following suggestions for future data collection insightful and very practical:
    - Information on availability and use of bike lanes
    - Information on rental/city bikes
- Levan would appreciate both use of percentages as well as hard numbers in order to understand the context and significance more
- Levan wants the ability to check the data, dashboards, visualizations, and results regularly and he stressed the importance of reusability and reproducibility, regardless of the expert/data science team
    - We suggested to him that it would make sense to "refresh" monthly

2. Data Understanding & Preparation

As per the CRISP-DM methodology, we've listed a few details regarding the data quantity and quality below:

2.1. Data quantity

Criteria	Results
What is the format of the data?	The three datasets come in .csv files No target feature was provided along the data
How is data captured?	The city of Chicago has published the data publicly on their Vision Zero Chicago site
How large is the data (in numbers of rows and columns)?	Crashes (33408 rows, 49 columns) Vehicles (75673 rows, 72 columns) People (82049 rows, 30 columns)

2.2. Data quality

Criteria	Results
Does the data include characteristics relevant to the business question?	The data includes many, many features, all of which could theoretically be relevant to the business question. In order to establish which characteristics are relevant to the business question and to rate the relevancy, a further exploration into the data is required.
What data types are present?	The vast majority of features are objects. There are also a few date/time and integers. • crashes: category(18), datetime64ns(2), float64(10), int64(7), object(12) • vehicles: category(10), datetime64ns(1), float64(11), int64(2), object(48) • people: category(12), datetime64ns(1), float64(4), object(13)
Possible to prioritize specific attributes? Is there anyone to provide more insight?	Since the goal of this project is to illuminate the factors which lead to deaths and injuries in crashes, we treated all features equally before modeling. We later decided to remove "post-crash" features before modeling, as we assumed these would not be useful in indicating the factors which lead to the accident. For example: • `REPORT_TYPE` • `DATE_POLICE_NOTIFIED` • `PHOTOS_TAKEN_I` • `EMS_AGENCY` • `HOSPITAL`
Have you identified missing attributes and blank fields? If so, is there meaning behind such missing values?	There are a handful of features which are missing, as well as a good amount of features which tend to be missing together. See missingness patterns chart.
Are there spelling inconsistencies that may cause problems in later merges or transformations?	In the `MODEL` feature (which gives more information about the vehicle involved in the crash), there are significant spelling inconsistencies and possible entry errors. This feature will be one of the features which the team decides to drop for the initial baseline model, as it adds too much complexity and any helpful information which this feature would have added can also be found in another feature (`VEHICLE_TYPE`).

2.3. Missing values

Crashes **:** There are 24 out of 49 columns have any missing value and 11 columns have over 50% missing values. There are sister features which are missing together as Figure 1 shows.

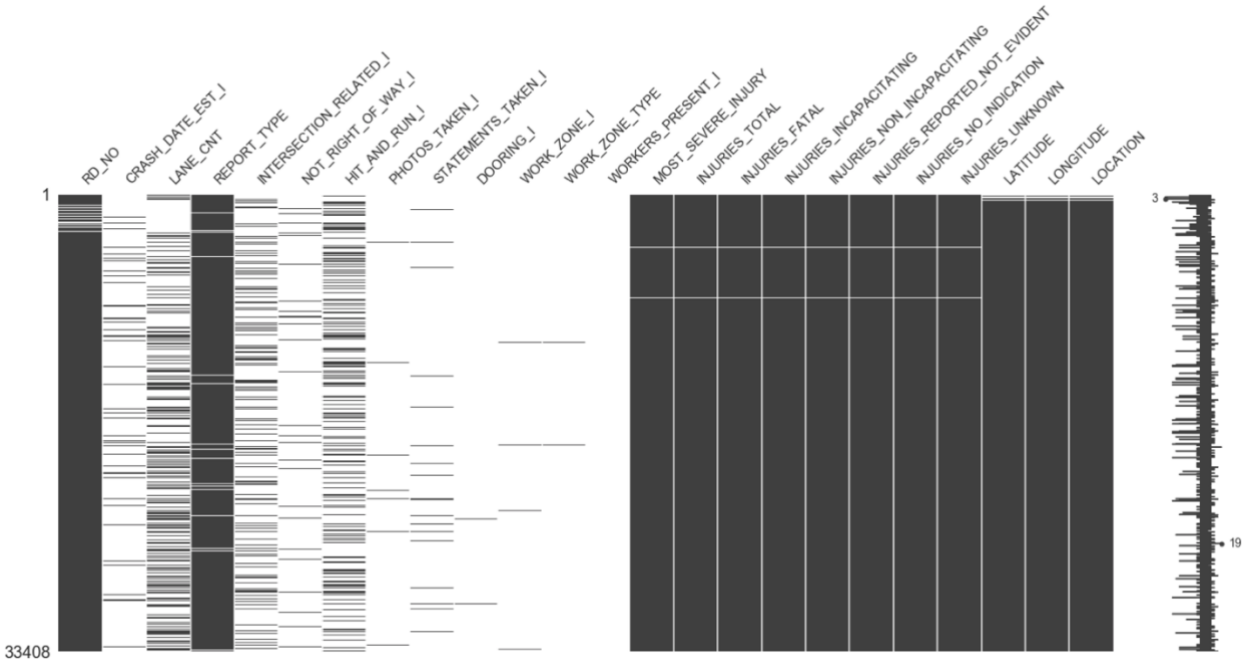


Figure 1

Vehicles**:** There are 68 out of 72 columns have any missing values and 54 columns have over 50% missing values. There are sister features which are missing together as Figure 2 shows. (for plotting purpose, we show only feature have less than 50% missing values)

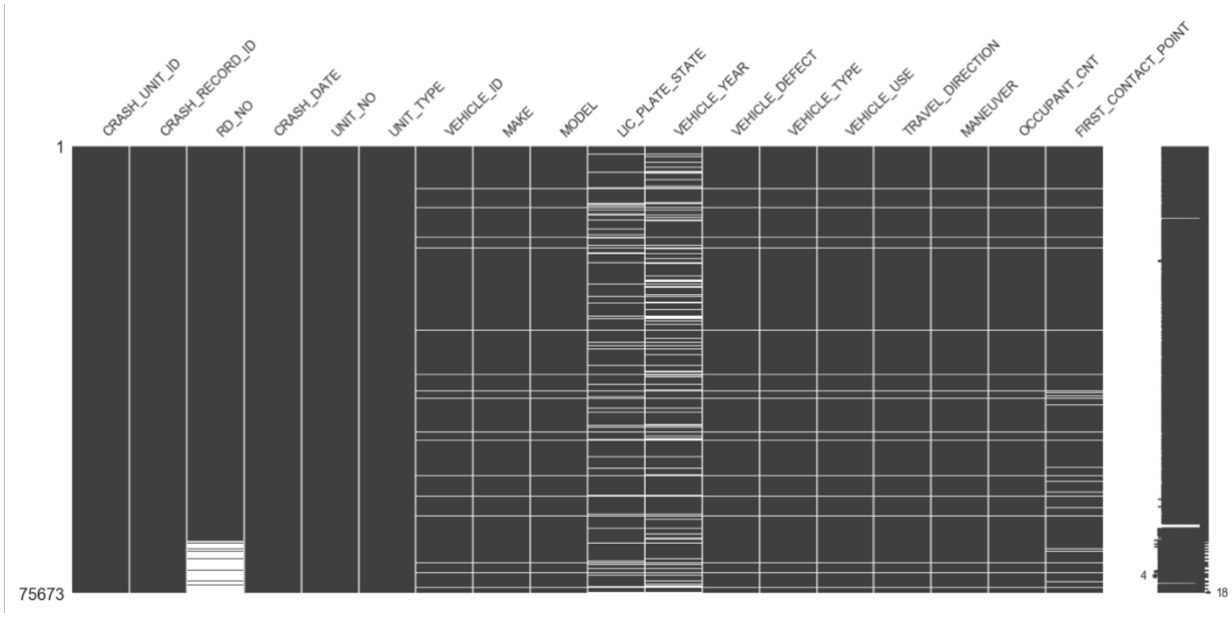


Figure 2

People**:** There are 26 out of 30 columns have any missing values and 9 columns have over 50% missing values. There are sister features which are missing together as Figure 3 shows.

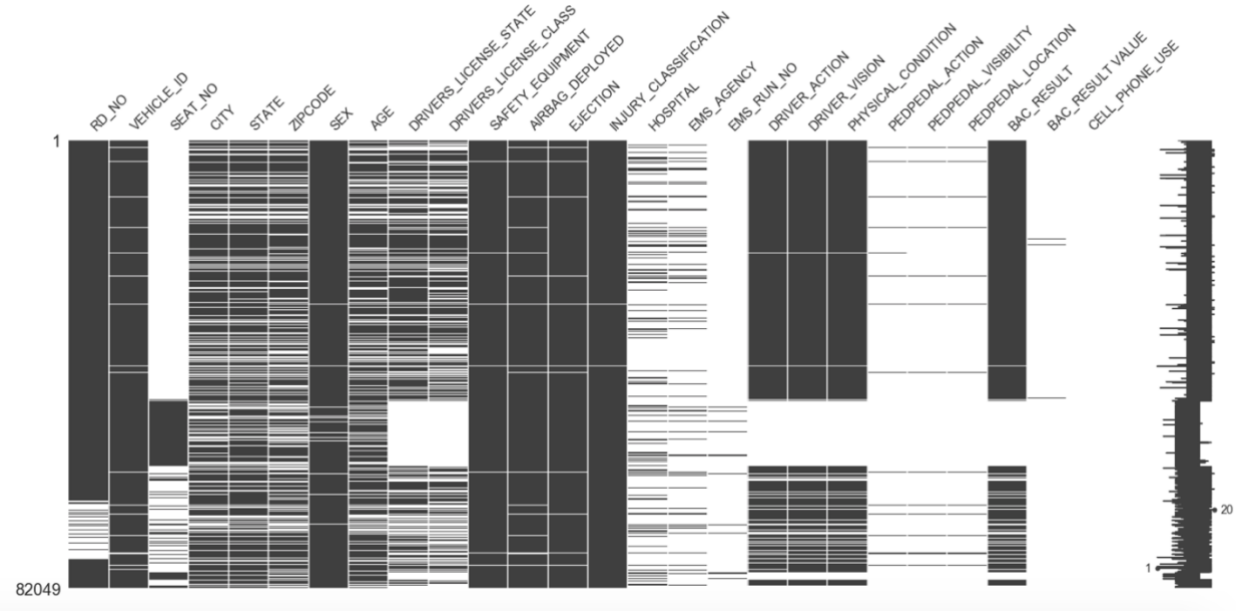


Figure 3

2.4. EDA findings, brief summary

2.4.1. Geographical analysis:

Overview of the crashes distribution:

- Based on the `LONGITUDE` and `LATITUDE` data, we plotted the data into Chicago map and we found that most of the crashes happened in downtown area as below graph shown (red and yellow parts in figure 4)

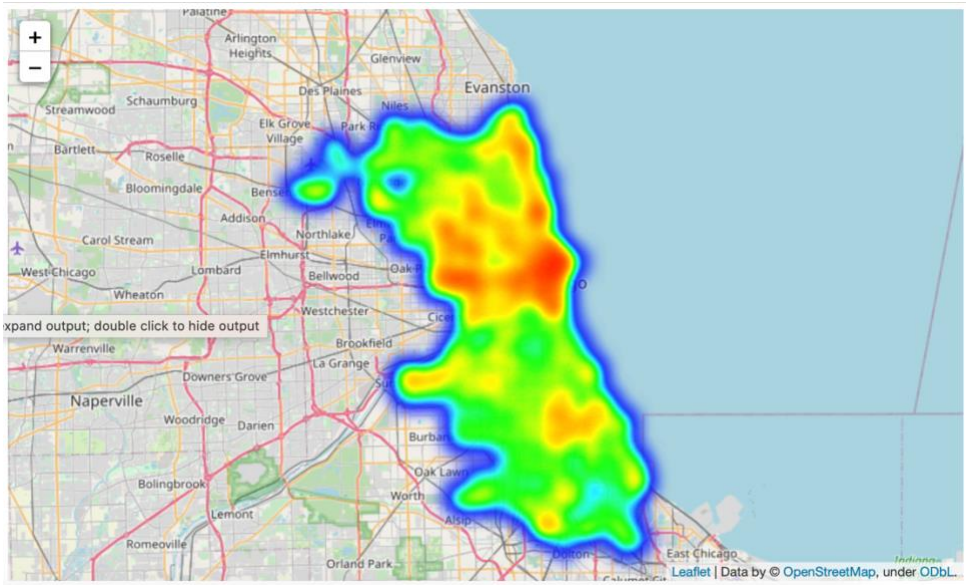


Figure 4

- Analysis the crashes distribution based on hours; there are more crashes happened in between 8am to 8pm.

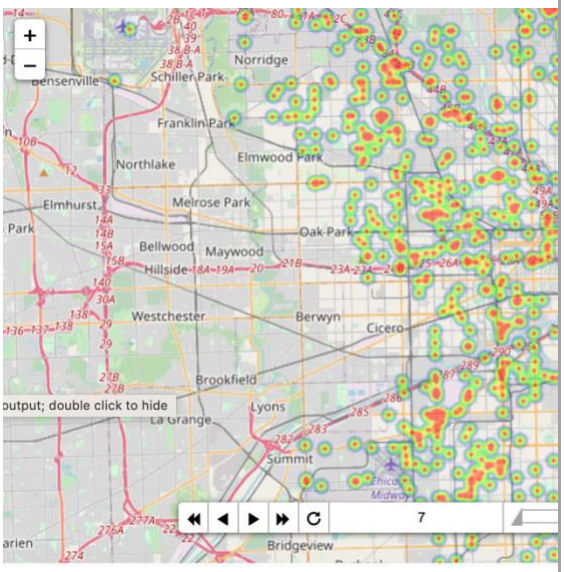
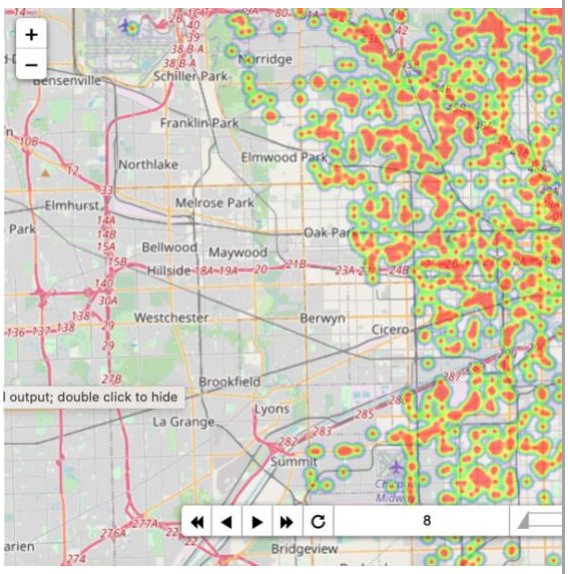
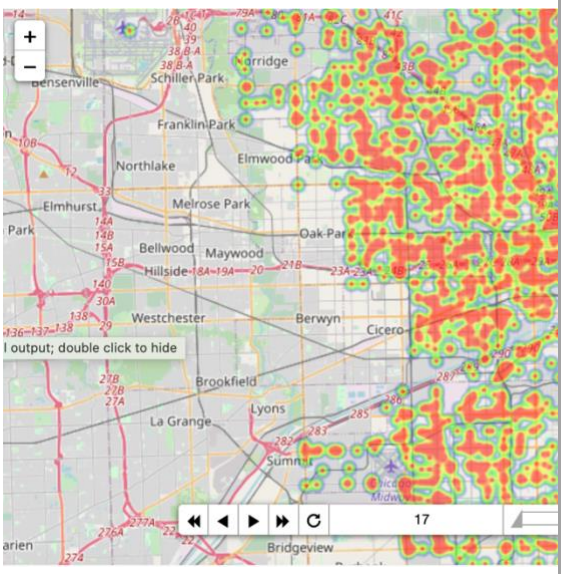
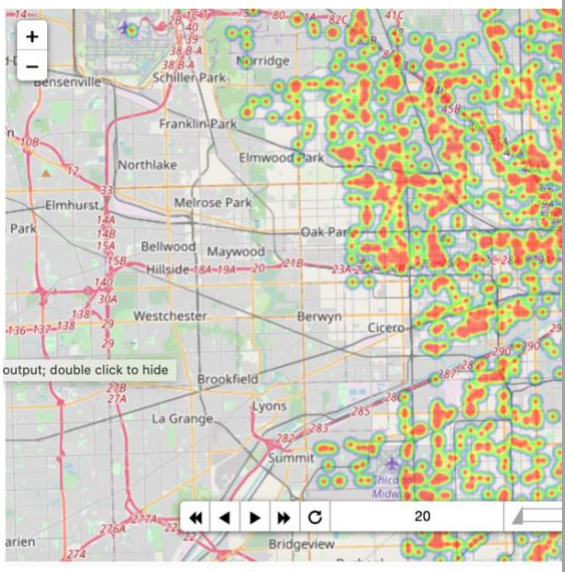
7am	8am
	
More accidents in downtown area	*The area of accidents got spread out a bit with focus in downtown area*
5pm	**8pm**
	
Accidents are widely spread and are more than with the focus still in downtown area	*Number of Accidents slowly decrease from 5pm with a focus on downtown area*

Figure 5

- Analysis of crashes based on Months of (2017 - 2020)

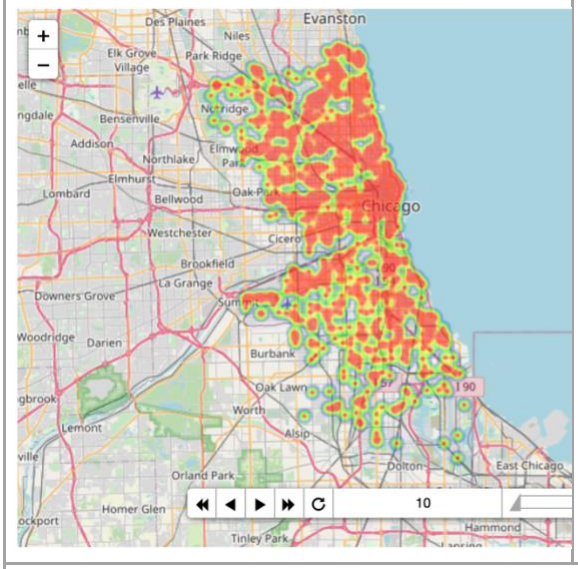
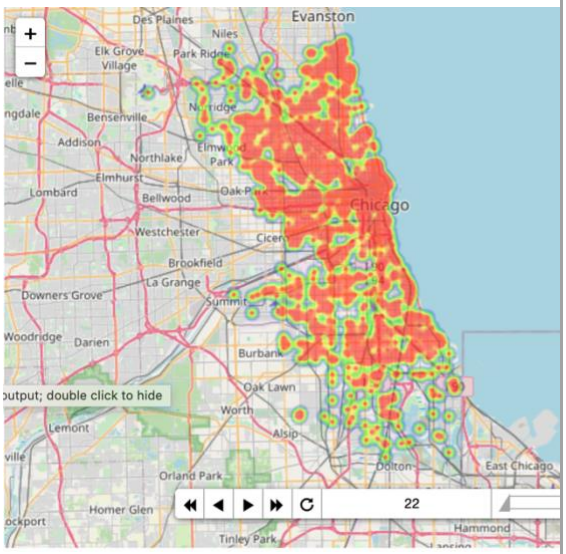
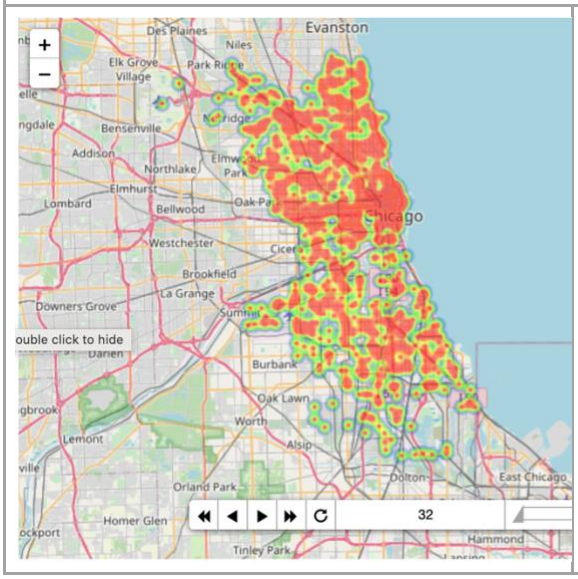
2017 October	2018 October
	
2019 October	
	

Figure 6

- Focus area (high risk areas)

According to the location information, combined with INJURY and FATAL crash information, we discovered that there are 8 community areas and collectively they collectively account for 65% of fatal crashes. These should be the focus area (figure 7).

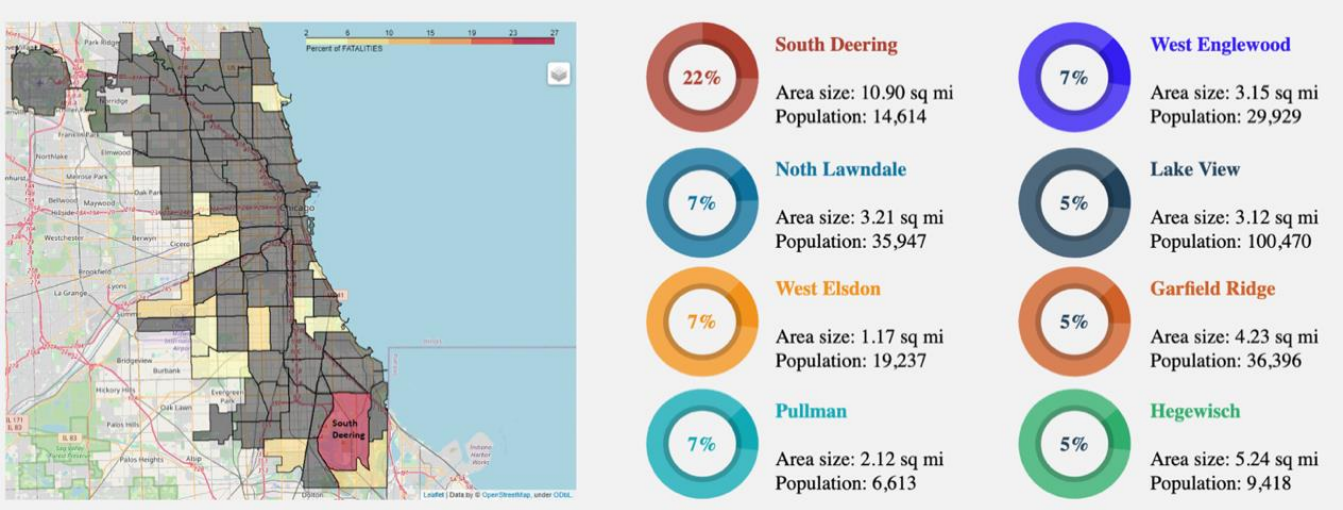


Figure 7

2.4.2. Analyses of `VEHICLE_YEAR` (age of car)

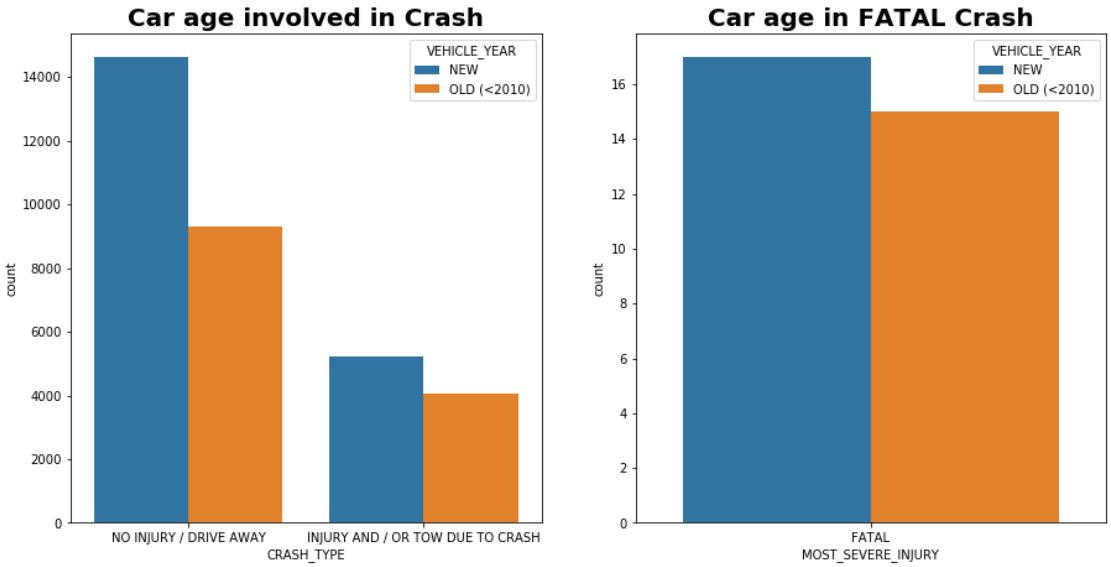


Figure 8

The cars with an age less than 10 years are more likely to be involved in Crashes and are more likely to be involved in Fatal crashes too.

2.4.3. Analysis of `AIRBAG_DEPLOYED`

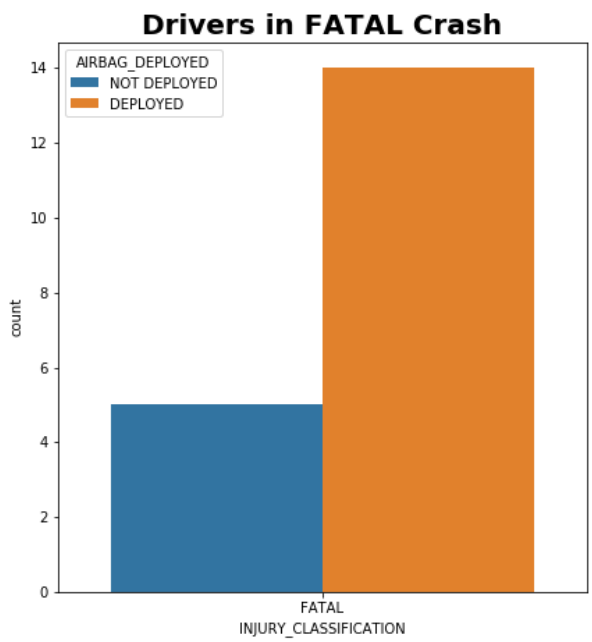


Figure 9

Also, the Fatal accidents were so severe that the Airbags deployment inside car proved to be ineffective and hence leading to fatalities.

2.5. Data merge

Before initial data splitting, our goal was to successfully merge all (3) datasets to preserve as much information as possible without too much redundancy for modeling.

According to the data portal,

- The `people` set has **many-to-one** relationship with `vehicles` set but a pedestrian in `vehicles` set has **one-to-one** relationship with `people` set.
- Passengers data from `people` dataset don't have a trajectory separate from the vehicle unit they belong to and they don't actively contribute to the accidents with their existence.

We aimed to include the information in `people` set only belong to vehicles (bicycle, pedestrian, driver) but not passengers to prevent data redundancy. For this reason, we took these steps:

- Filtered out passengers from `people` (shape: 82049, 30) and created `non_passengers` (shape: 64835, 30).

PERSON_ID	PERSON_TYPE	...
O229153	BICYCLE	...
P211417	PASSENGER	...
O925931	PEDESTRIAN	...
P211625	PASSENGER	...
O925562	DRIVER	...
...	...	...

>

PERSON_ID	PERSON_TYPE	...
O229153	BICYCLE	...
O925931	PEDESTRIAN	...
O925562	DRIVER	...
...	...	...

Figure 10

- Left-joined non_passengers (shape: 64835, 30) to vehicles (shape: 75673, 72) on CRASH_RECORD_ID, RD_NO, VEHICLE_ID, CRASH_DATE and created vehicles_with_people (shape: 75838, 98) that preserve redundant information.

CRASH_RECORD_ID	RD_NO	VEHICLE_ID	CRASH_DATE	...
70a18f80d33a3e2bdde9f21c7c0cafaa8d133285c2a395...	JD310165	877509	26.07.2020 01:50	...

- And lastly, inner joined vehicles_with_people (shape: 75838, 98) to crashes (shape: 33408, 49) on CRASH_RECORD_ID, RD_NO, CRASH_DATE and created data (shape: 68503, 144)

CRASH_RECORD_ID	RD_NO	CRASH_DATE	CRASH_UNIT_ID	UNIT_NO	...
0211e1f766f3940dfa87375661d25b716655e908c320cc	JC301403	11.06.2019 08:40	667550	1	...
0211e1f766f3940dfa87375661d25b716655e908c320cc	JC301403	11.06.2019 08:40	667551	2	...
..	...	...	...	...	...

This method increased crashes records only **per active parties** involved, while doing so passengers are still involved in statistics through features starts with "INJURY" if they got harmed. With this method, we preserved full information across datasets.

2.6. Missing values:

After merging the datasets, we investigated again the missing values. We found that 74 out of the 144 features have over 50% missing values. For the initial baseline model, we decided to take the following steps as our first trial:

- Missing values on these features are replaced "N" (for "no"), as we assume the absence of a value indicates a non-yes.

Features	Imputation Values
INTERSECTION_RELATED_I	"N"
HIT_AND_RUN_I	"N"
WORK_ZONE_I	"N"
EXCEED_SPEED_LIMIT_I	"N"

- Missing values on these features are replaced with the values listed below:

Features	Imputation Values
LIC_PLACE_STATE	"IL"
SEX	"X"
SAFETY_EQUIPMENT	"USAGE UNKNOWN"
AIRBAG_DEPLOYED	"DEPLOYMENT UNKNOWN"

- Missing values on these features are replaced with the median or the mode of the feature:

Features	Imputation Values
LANE_CNT	MEDIAN, MODE (both values are same)
OCCUPAN_CNT	MEDIAN
VEHICLE_YEAR	MODE
AGE	kNN

- We dropped all rows where these features were missing:

Features	Number of dropped rows
LOCATION	249
MOST_SEVERE_INJURY	115

After these actions, there are still 42 features containing missing values. In order to save time and limit the complexity of our first model, the following features were dropped for the simplicity of the baseline model:

"RD_NO"	"CRASH_DATE_EST_I",	"TRAFFIC_CONTROL_DEVICE",
"DEVICE_CONDITION	FIRST_CRASH_TYPE",	'ALIGNMENT'
REPORT_TYPE"	'DAMAGE'	'DATE_POLICE_NOTIFIED'
'SEC_CONTRIBUTORY_CAUSE'	'STREET_NO'	'STREET_DIRECTION'
'STREET_NAME'	'BEAT_OF_OCCURRENCE'	'PHOTOS_TAKEN_I
'STATEMENTS_TAKEN_I'	'DOORING_I'	'WORK_ZONE_TYPE'
'WORKERS_PRESENT_I'	'INJURIES_REPORTED_NOT_EVIDENT'	'INJURIES_NO_INDICATION'
'INJURIES_UNKNOWN'	'LATITUDE'	'LONGITUDE'
'LOCATION'	'CRASH_UNIT_ID'	'UNIT_NO'
'UNIT_TYPE'	'VEHICLE_ID'	'MAKE'
'MODEL'	'VEHICLE_DEFECT'	'VEHICLE_USE',
'TRAVEL_DIRECTION'	'TOWED_I'	'TOWED_BY'
'TOWED_TO'	'PERSON_ID'	'PERSON_TYPE'
'SEAT_NO'	'CITY'	'STATE'
'ZIPCODE'	'DRIVERS_LICENSE_CLASS'	'AIRBAG_DEPLOYED'
'EJECTION'	'INJURY_CLASSIFICATION'	'HOSPITAL'
'EMS_AGENCY'	'EMS_RUN_NO'	'DRIVER_ACTION'
'DRIVER_VISION'	'PHYSICAL_CONDITION'	'PEDPEDAL_ACTION'
'PEDPEDAL_VISIBILITY'	'PEDPEDAL_LOCATION'	'BAC_RESULT'
'BAC_RESULT VALUE'	'MANEUVER'	'FIRST_CONTACT_POINT'
'VEHICLE_TYPE'		

**Features are listed in table format solely for ease of reading.

This left us with 33 features in our baseline model:

CRASH_RECORD_ID	NOT_RIGHT_OF_WAY_I	CRASH_MONTH
CRASH_DATE	HIT_AND_RUN_I	NUM_PASSENGERS
POSTED_SPEED_LIMIT	WORK_ZONE_I	LIC_PLATE_STATE
WEATHER_CONDITION	NUM_UNITS	VEHICLE_YEAR
LIGHTING_CONDITION	MOST_SEVERE_INJURY	OCCUPANT_CNT
TRAFFICWAY_TYPE	INJURIES_TOTAL	EXCEED_SPEED_LIMIT_I
LANE_CNT	INJURIES_FATAL	SEX
ROADWAY_SURFACE_COND	INJURIES_INCAPACITATING	AGE
ROAD_DEFECT	INJURIES_NON_INCAPACITATING	DRIVERS_LICENSE_STATE
CRASH_TYPE	CRASH_HOUR	SAFETY_EQUIPMENT
INTERSECTION_RELATED_I	CRASH_DAY_OF_WEEK	CELL_PHONE_USE

**Features are listed in table format solely for ease of reading.

2.7. Outliers

As most of the features were categorical, there were very few outliers to be treated in numerical data. Without more context, we assume that categorical outliers do not exist, at least for the simplicity of our baseline model. We treated the numerical outliers for our baseline model using our research in the subject matter (motor vehicles, Chicago infrastructure, Chicago laws and regulations).

For LANE_CNT, which refers to the number of street lanes excluding turning lanes, all outliers were between 9 – 99 lanes. As per our research, the highest number of lanes on a Chicago street should be 6. Because the number of outliers was insignificant (<20) compared to the size of the data and because all outliers were above 6, we decided to replace all outliers with 6.

2.8. Transformations/normalizations

For the initial baseline model we did not perform and transformations/normalizations.
However, it was performed on numerical features in one of the iterations to check the impact on the model performance.

2.9. Feature engineering

2.9.1. Target Feature

The first feature we engineered is the target feature. Since our goal with this project is to help Vision Zero identify possible causes which lead to deaths/injuries in accidents, we decided to create a simple, categorical target feature for our baseline model - INJURY. If an accident resulted in a fatal, non-incapacitating injury, incapacitating injury which basically mean death or any type of injury, this is encoded as 1 with the label : INJURED. If not, 0 with the label NOT INJURED. With this feature engineered, the balance of injured to not-injured is 11% (5,892 rows) to 89% (48,496 rows).

2.9.2. Other features

2.9.3. Level reduction

Many features in our datasets have multiple levels. In order to reduce the level to reduce the complexity, we group some of them for better insight extraction.

Feature name	Reduce Method
FIRST_CONTACT_POINT	Reduce it to focus on only FRONT, SIDE, REAR and OTHER
MANEUVER	Merge same type (LANE","OVER","ENTER) into TURN
SAFETY_EQUIPMENT	If USED or HELMET is in the string, group it as USED SAFETY EQUIP. If "NOT", "IMPROPER","NONE PRESENT" is involved, then group it as DID NOT USE SAFETY EQUIP

AIRBAG_DEPLOYED	If NOT DEPLOYED, NOT UNKNOWN are in the level name, group it as NOT DEPLOYED, otherwise, as DEPLOYED
CRASH_HOUR	Binned the hour into 4 levels; • 2-8am : Early Morning • 8~12pm : Morning • 12~18pm : Afternoon • 19pm ~ 2am : Night
TRAFFIC_CONTROL_DEVICE	Group levels have NO CONTROLS, UNKNOWN, OTHER as NO_SIGN, Others as SIGN
TRAFFICWAY_TYPE	Group NOT, ONE-WAY as NOT_DIVIDED, Others as DIVIDED
LOCATION	Based on the LATITUDE (41.84 <= LATITUDE <= 41.9100064) & LONGITUDE (-87.7421459 <= LONGITUDE <= -87.50) marked as Downtown, Others as Not Downtown Results as Not Downtown: 39902 rows of records Downtown: 14486 rows of record
AGE_GROUP	Based on AGE column, group into 4 levels; • 0-18 : below 18 • 19-30 : between 19 and 30 • 31-60 : Middle Age 31 and 60 • Else : Older than 60

2.9.4. Additional features

Feature Name	**Method**
VEHICLE_AGE	CRASH_DATE - VEHICLE_YEAR
AGE_SEX_GROUP	Combine AGE and SEX column

2.10. Missing values treatment for our final model

Categorical Features treatment I: Many categorical features have UNKNOWN/NA which we decided to change these rows into missing values and impute them. Since we knew that we would create dummy features of below imputed features before modeling, if any of the class such as UNABLE_TO_DETERMINE or OTHER were supposed to contribute to the model then we would dive deeper into this category and investigate more.

Features	**Imputation Values**
LIGHTING_CONDITION	"DAYLIGHT"
TRAFFICWAY_TYPE	“NOT DIVIDED”
AIRBAG_DEPLOYED	"UNABLE TO DETERMINE"
TRAFFIC_CONTROL_DEVICE	“NO CONTROLS”
DEVICE_CONDITION	“NO CONTROLS”
WEATHER_CONDITION	“CLEAR”
ROADWAY_SURFACE_COND	“NO DEFECTS”
ROAD_DEFECT	“CLEAR”
VEHICLE_DEFECT	“UNABLE TO DETERMINE”
VEHICLE_TYPE	“OTHER”
TRAVEL_DIRECTION	“N”

MANEUVER	“OTHER”
SAFETY_EQUIPMENT	“UNABLE TO DETERMINE”
AIRBAG_DEPLOYED	“UNABLE TO DETERMINE”
EJECTION	“UNABLE TO DETERMINE”
DRIVER_ACTION	“OTHER”
DRIVER_VISION	“OTHER”
PHYSICAL_CONDITION	“UNABLE TO DETERMINE”
PEDPEDAL_ACTION	“UNABLE TO DETERMINE”

Categorical Features treatment II:

Features	**Imputation Values**
INTERSECTION_RELATED_I	“N”
NOT_RIGHT_OF_WAY_I	“N”
HIT_AND_RUN_I	“N”
DOORING_I	“N”
WORK_ZONE_I	“N”
LIC_PLATE_STATE	“IL”
EXCEED_SPEED_LIMIT_I	“N”
FIRST_CONTACT_POINT	“OTHER”
PERSON_TYPE	“UNABLE TO DETERMINE”
CITY	“OTHER”
SEX	“UNABLE TO DETERMINE”
CELL_PHONE_USE	“UNABLE TO DETERMINE”

Numerical features treatment I: VEHICLE_YEAR, NUM_UNITS, POSTED_SPEED_LIMIT, AGE are treated using kNN imputation method by setting n_neighbors = 5.

Numerical feature treatment II:

Features	**Imputation Values**
LANE_CNT	2
NUM_PASSENGERS	0
OCCUPANT_CNT	0
BAC_RESULT VALUE	0

3. Evaluation Method

3.1. Data Split

We split the data as 80% train and 20% test sets in order to keep at least 10,000 observations at test set. And during the split, we shuffled the data to reflect the randomness that would be expected on yet-to-be-seen data and checked if the classes were proportionally distributed.

3.2. Metric

Our thought process regarding the main evaluation metric was to prioritize the tolerence levels in favor for either false positives or false negatives. We decided that in this case, false alarms (FPs) can be tolerated, but false negatives (FNs) would not be tolerable when identifying which crashes will result in an injury or fatality.

For this reason, we chose Sensitivity / Recall over Precision. By evaluating sensitivity scores, we wanted to measure how sure we were that our model were not missing any injuries or fatalities.

3.3. Bootstrapped confidence interval

In order to make sure how stable the model is, we designed a rigorous procedure to add a bootstrapped confidence interval to the model's estimates.

After preprocessing train and test sets separately with the same procedure, including same missing value imputer object, except sampling which is only implemented on train set, we would test the model against 1000 boostrapped samples each for train and test sets, and store them in lists for X and y values separately. Those lists are `bootstrap_X_train`, `bootstrap_y_train`, `bootstrap_X_test`, `bootstrap_y_test` each with 1000 indicies.

Each index represents an iteration. For each iteration i, we

- fit the model classifier with tuned hyperparameters to the i-th bootstrapped sample of the lists `bootstrap_X_train` and `bootstrap_y_train`,
- predict i-th bootstrapped test sample of the list `bootstrap_X_test`,
- calculate sensitivity score by comparing the predictions with the actual bootstrapped y values of i-th sample of the list `bootstrap_y_test`.
- store i-th sensitivity score in a list `recall_scores` at the end of each iteration

4. Modelling

4.1. Baseline model

Post data preparation, all the pre-crash features were selected to create a baseline model to predict 'Injury' with two classes as '`INJURED`' and '`NOT_INJURED`'. A Random Forest Classifier model from scikit learn machine learning library was trained on the pre-crash features with default parameter n_estimators=100 to predict '`INJURED`' as a positive class and later evaluated using a cross validation generator cross_val_score from scikit learn model selection library with StratifiedKFold as 5 and obtaining Recall, Precision and F1 score on training dataset by setting scoring parameter as 'recall', 'precision' and 'f1'.

Following were the average results obtained for the same (baseline model):

Experiment base model - Picked all pre-crash features (Random Forest)	
Recall	0.21
Precision	0.77
F1 Score	0.32

As the recall score was very low (0.21), which signifies that the model can only correctly classify/predict 2 INJURY out of 10 cases. In order to get more reliable model with the sight of making use of the model to identify most valuable feature which the clients can act on and get valuable insights contributing to their cause, we hand-picked 10-15 most important

features obtained through the feature importance score through Random Forest model and Permutation importance through scikit learn inspection library.

With multiple experiments having different sets of input features, we were able to narrow down our input features to following 13 features:

```
"AGE","LANE_CNT","AIRBAG_DEPLOYED","PRIM_CONTRIBUTORY_CAUSE","POSTED_SPEED
_LIMIT","NUM_UNITS","TRAFFICWAY_TYPE","SEC_CONTRIBUTORY_CAUSE","VEHICLE_AG
E","FIRST_CRASH_TYPE","INJURY","LIGHTING_CONDITION","SEX"
```

Performed one-hot encoding using get_dummies from pandas on the categorical features and then we treated the missing values of these 13 features using KNNImputer from sklearn.impute library and set n_neighbors = 5 as its default value and round up the values then reset its index. With the same Random Forest parameter and cross_val_score we obtained average scores as below:

Experimental result based sub-selected features	
Recall	0.38
Precision	0.73
F1 Score	0.50

Further we trained our model with other classifier algorithms like Logistic Regression and XGBoost classifier. The results obtained from them as table below shown were no better than the above table.

	Metric	Accuracy	Precision	Recall	F1 Score	AUC
0	LogisticRegression	0.909	0.655	0.337	0.445	0.658
0	RandomForestClassifier	0.915	0.699	0.380	0.492	0.680
0	XGBClassifier	0.915	0.693	0.388	0.497	0.683

Figure 11

Since the classes we were trying to predict in Injury were imbalanced i.e. INJURY class contributed 5,892 instances (11%) compared to `NOT_INJURED` class with 48,496 instances (89%), we implemented and experimented with following balancing techniques with different weightage and compare with different classifier algorithms. We then trained Support Vector Machine Classifier, Logistic Regression Classifier and XGBoost Classifier with resampled train dataset.

4.2. Resampling method

4.2.1. Under-sampling using RandomUnderSampling:

With RandomUnderSampling from imblearn package we were able to produce following scores through cross_val_score and StratifiedKFold as 5 with mentioned algorithms. The RandomUnderSampling was tried with two sampling strategy:

- sampling_strategy='majority': where the majority class 'NOT_INJURED' was randomly under-sampled to number of instances of 'INJURED' class.
- sampling_strategy=0.50: where the majority class 'NOT_INJURED' instances were under-sampled to twice as of 'INJURED' class instances.

As table shown below, Random forest with sampling strategy as 'majority' provides better Precision-Recall trade-off than others.

	Metric	Accuracy	Precision	Recall	F1 Score	AUC
0	LogisticRegression majoritymajoritymajoritymaj...	0.824	0.343	0.682	0.456	0.761
0	LogisticRegression 50.0%	0.890	0.492	0.565	0.526	0.747
0	SVC majoritymajoritymajoritymajoritymajorityma...	0.746	0.248	0.663	0.361	0.710
0	SVC 50.0%	0.895	0.603	0.081	0.143	0.537
0	RandomForestClassifier majoritymajoritymajorit...	0.792	0.308	0.742	0.436	0.770
0	RandomForestClassifier 50.0%	0.878	0.453	0.618	0.523	0.764
0	XGBClassifier majoritymajoritymajoritymajority...	0.804	0.319	0.715	0.441	0.765
0	XGBClassifier 50.0%	0.881	0.462	0.603	0.523	0.759

Figure 12

4.2.2. Oversampling using RandomOverSampling and SMOTE

We also tried Oversampling techniques like RandomOverSampling and SMOTE. But the results were not better than Undersampling.

Oversampling using RandomOverSampling

	Metric	Accuracy	Precision	Recall	F1 Score	AUC
0	LogisticRegression minorityminorityminoritymin...	0.826	0.346	0.685	0.460	0.764
0	LogisticRegression 50.0%	0.890	0.495	0.563	0.527	0.747
0	RandomForestClassifier minorityminorityminorit...	0.907	0.591	0.469	0.523	0.715
0	RandomForestClassifier 50.0%	0.910	0.609	0.470	0.530	0.716
0	XGBClassifier minorityminorityminorityminority...	0.851	0.391	0.675	0.495	0.774
0	XGBClassifier 50.0%	0.897	0.522	0.577	0.548	0.757

Figure 13

Oversampling - SMOTE

	Metric	Accuracy	Precision	Recall	F1 Score	AUC
0	LogisticRegression minorityminorityminoritymin...	0.839	0.366	0.663	0.471	0.762
0	LogisticRegression 50.0%	0.892	0.502	0.559	0.529	0.746
0	RandomForestClassifier minorityminorityminorit...	0.914	0.673	0.395	0.498	0.686
0	RandomForestClassifier 50.0%	0.913	0.671	0.392	0.495	0.684
0	XGBClassifier minorityminorityminorityminority...	0.915	0.690	0.391	0.499	0.685
0	XGBClassifier 50.0%	0.915	0.687	0.388	0.496	0.683

Figure 14

4.2.3. RandomUnderSampling post scaling

As most of the features were outcome of one-hot encoding of the categorical features, making the dataframe mostly binary i.e. 0 and 1. In order to have same influence of the continuous features on the model, standard scaling was implemented on the numerical features like `'AGE', 'VEHICLE_AGE',' POSTED_SPEED_LIMIT,' NUM_UNITS',' LANE_CNT'.`

The results obtained after cross validation of the same were similar to the one obtained without scaling. Hence, we proceeded with undersampled data and random forest for further betterment of the model.

4.3. Hyperparameter Tuning

Hyperparameter tuning was performed using GridSearchCV from scikit learn package to get the best hyperparameter combination for Random Forest to get the optimum Recall. Below were the parameters set and were changed/added in few iterations to get the best parameter which would not overfit the model.

```
parameters = {'n_estimators': [100, 300, 500, 800, 1000],
              'max_features': ['log2', 'sqrt','auto'],
              'criterion': ['entropy', 'gini'],
              'max_depth': [5, 8, 10, 13],
              'min_samples_split': [2, 5, 10, 15],
              'min_samples_leaf': [5, 8, 15]}
```

After running it through GridSearchCV, below were the best hyperparameter combination:

```
n_estimators=800,min_samples_split=15, min_samples_leaf=8,
max_features='log2', max_depth=13, criterion='gini'
```

After training Random Forest model with these parameters on Undersampled data, below were the scores obtained through cross_val_score with improvement especially on Recall and F1 Score.

Post Hyperparameter tuning on Undersampled train dataset with Random Forest		
	Final Model	**Baseline Model**
Recall	0.69	0.21
Precision	0.80	0.77
F1 Score	0.74	0.32

Since, the dataset was imbalanced and the test dataset will probably be imbalanced, we certainly were not relied on ROC AUC curve to understand the model performance. As Precision-Recall trade-off was more significant for us, we plotted them.

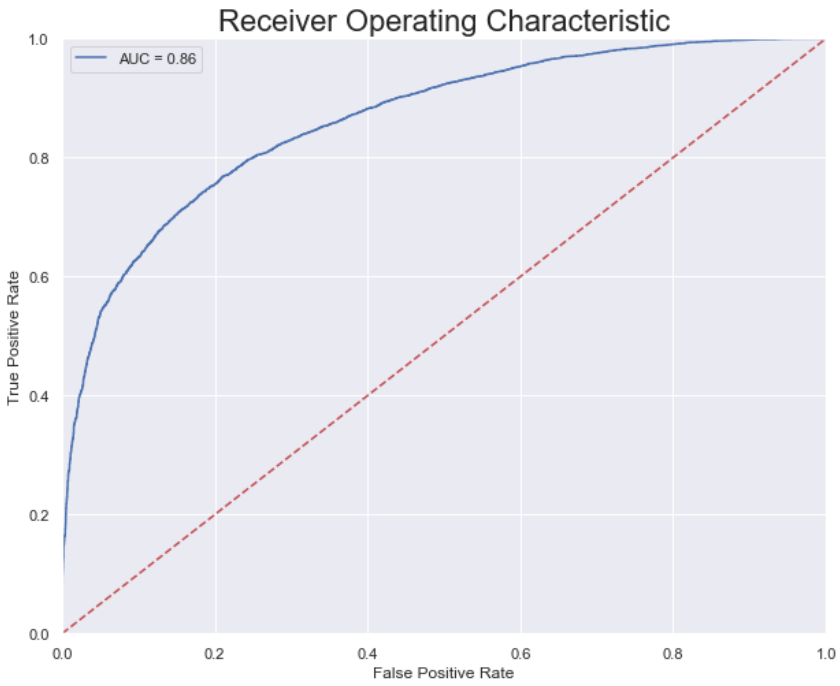


Figure 15

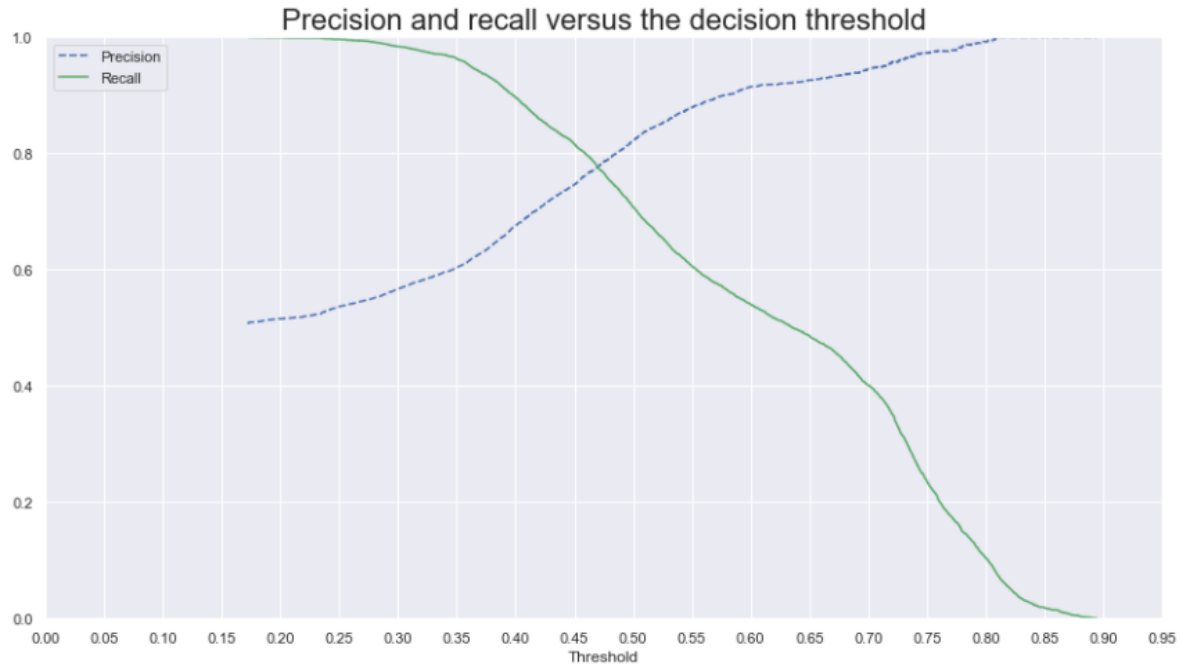


Figure 16

Here the Precision-Recall curve signifies that to get the perfect trade-off, we can modify our default threshold of 0.50 to 0.47 or move the threshold to higher values to achieve better Recall at the loss of Precision. We decided to keep the threshold as default 0.50 since it fulfilled our purpose.

4.4. Performance on test data

Post finalisation of model, the model was tested by predicting the model on test dataset which was pre-processed based on the properties and steps followed for train dataset e.g. feature engineering, imputation of the missing values based on the KNN imputation and fit on Train data, etc. Following were the prediction results obtained:

Test data results	
Recall	0.69
Precision	0.33
F1 Score	0.45

As the results obtained through test dataset are not as better as train dataset, we can surely know that the model is a little overfitting the train dataset. Since the amount of data lost in Undersampling is huge, we believed that the oversampling models may perform better on test dataset.

Post running through oversampled dataset with the best hyperparameters achieved through updated GridSearchCV run for the Random Forest classifier, the model was trained on oversampled train data and then tested on test data. The scores obtained were much lower than above ones, hence we finalised the model trained on undersampled data.

Below was the observed Precision-Recall curve obtained based on the prediction on test data:

Figure 17

This curve suggests that if we move the threshold towards the higher value of 0.50, we can achieve better Precision-Recall trade-off. But as we are interested in Recall more than Precision, for our case the default threshold of 0.50 would be optimum. Hence verified.

5. Results and Outlook

As explained before, our sensitivity value by 5-fold cross validation landed around **0.21** in our first baseline. Through data balancing (subsampling) and hyperparameter tuning efforts, we improved the value up to **0.69** on average on train set. The selected classifier's evaluation on **test set** gave us **0.68** sensitivity value with accuracy of 0.823 and precision of 0.326.

In the final step, **95% confidence interval**, which determine the outcome from 95% of the time, for the values of recall_scores was between **0.67** and **0.73,** across 1000 bootstrapped predictions. The histogram with normal distribution, with showed us the model can be recognized as reliable and it generalizes well on the yet-to-be-seen data.

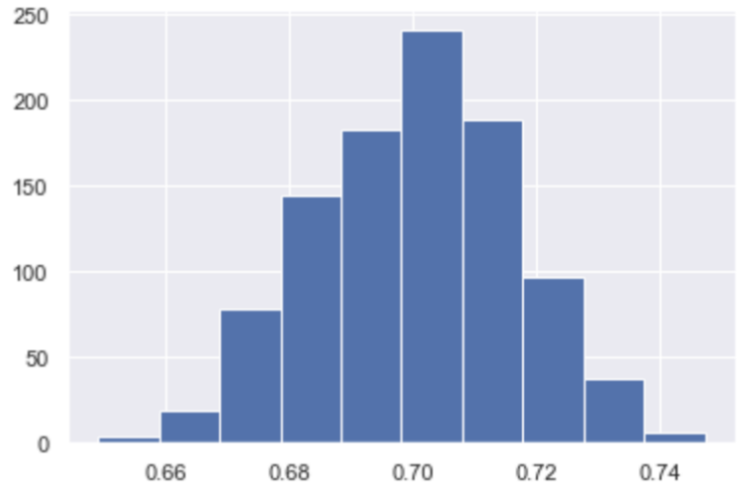


Figure 18: Histogram with 1,000 bootstrapped predictions

From the Chicago administration's perspective, this recall result shows that the model with its pre-selected features can **accurately predict 7 out of every 10 accidents which result in injuries or fatalities**. Thus, from this result, client or related stakeholders can believe in the model's performance in a real-world scenario and can look deeper into pre-selected features in order to come up with preventive actions and strategies to reduce the number of injuries and fatalities which are the main objectives of this project.

From our team's perspective, next steps can be separated into two main categories, namely **key focuses** and **improvements**. Our first suggested next step is to focus on this primary objective by analyzing these contributing factors, including **high risk areas/neighborhoods**, **speed limits**, and **primary contributory causes** which we identify as important features of the model as explained in modeling section.

Considering **a multivariate analysis of contributing factors**, there are various insights which can be beneficial for the clients and primary goal of the project. For high risk areas, **eight communities**, which are only 15% of total areas and 13% of total populations, are collectively accountable for more than 65% of total fatal crashes. These mentioned areas include South Deering, West Englewood, North Lawndale, Lake View, West Elsdon, Garfield Ridge, Pullman, and Hegewisch. Moreover, focusing on area aspect, the data shows that most of the accident happen at night with **more than 55% fatal injuries rate** compared to total fatal injuries with highly influential factors such as intersection, road crossing, and behavior of driver (improper lane use, failing to reduce speed, and failing to yield right-of-way).

Apart from high risk areas, the data from **speed limit variable** indicates that accidents with injuries in **mid speed area (25 - 40 mph)** occur almost twice as often as accidents in other areas and this also applies to fatal injuries. Another interesting aspect to consider is a multivariate analysis of primary contributory cause with age and gender. According to the impact from the crashes, **people age over 60** years old are accountable for 41% of total fatalities and 30% of injuries, while the numbers are 27% and 25% respectively for **people**

age between 19 and 30. While one of the main reason of crashes happening to males age over 60 is **following too closely**, and it is **improper overtaking and passing** for females age over 60. For people age between 19 and 30, the primary contributory cause for the crash is **failing to reduce speed to avoid crash** and **failing to yield right-of-way** for both males and females. All these multivariate analyses should be closely monitored, then come up with various solutions to handle each case specifically. Also, the results from the analysis or the contributing features from the model can change as time passes. Thus, we would recommend data science team to look into these mentioned findings and variables, also monitor how and why the values change in the future, then conduct suggestions which are most effective based on each time period.

For next step of improvements, we would suggest to improve the **data collection** and **data quality**. For data collection, we believe that there are topics that require further information, such as **crashes involving cyclists.** The current data doesn't contain much information about cyclists who are one of the most impact parties from the crashes. Hence, we would recommend future data scientists working for this project to try to get some insightful variables to address the questions like **"was there a bike lane on the road during crashes?**, or "**is the bike involved in crashes rented or city bike?**". With these new variables, it will enable us to identify and analyze more causes of the crashes involving cyclists and come up with preventive actions specifically for this impact group.

Lastly, for data quality, we observed that information about alcohol test are **missing together** with license plate, license class, driver action and physical condition. This issue should be looked into in order to identify the root causes, because we all know that alcohol level of the drivers are one of the main factor to crashes. Also, to improve the data quality, we would like to suggest the data science teams to conduct **data quality report on a monthly basis** to monitor the quality and completeness of data stored in the system. Additionally, the team should set-up the quality **control standard for each feature**, especially important features, which will then be used to cross-check with the data quality report.