Predicting Accident Severity

Babu Konnayil

Oct 31, 2020

1. Introduction

1.1 Background

Major traffic delays or traffic blockages are results of traffic accidents. The severe the accident, the more time to clear the traffic block.

Can you imagine missing your son or daughter's graduation ceremony or being late to best friend's wedding or miss your most important business meeting just because traffic block due to accidents? Wouldn't it be nice to have a tool that can predict whether an accident will happen and its severity given the weather condition and road condition so that you can plan ahead and be careful to avoid accidents?

Our capstone project and tool will help to predict traffic accidents and its severity given road condition and weather condition.

1.2 Problem

Seattle Depart of Traffic has huge data related to traffic accidents. But it has many fields directly and indirectly related to the accident and its cause. We aim to look at the data carefully and analyze which one to be used for predicting accident severity and which model to be used for prediction.

1.3 Interest

Our target audiences are vehicle owners in Seattle and various other government and corporate companies providing necessary support services various types of accidents.

2. Data acquisition and cleaning

2.1 Data sources

I will be using data directly obtained from Seattle Department of Traffic (SDOT)for this project as same data from Coursera had only two type of severity code samples (1 and 2) instead of at least four different types of severity samples.

Data directly downloaded from: https://data.seattle.gov/Land-Base/Collisions/9kas-rb8d

Attribute details can be found at:
https://www.seattle.gov/Documents/Departments/SDOT/GIS/Collisions_OD.pdf

2.2 Data Cleaning

Data that I downloaded from SDOT had 221266 records and 40 attributes. It had many missing values. The attributes were of multiple type and required a close look at whether we need to do any type changes as well.

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 221266 entries, 0 to 221265
Data columns (total 40 columns):
X                  213797 non-null float64
Y                  213797 non-null float64
OBJECTID           221266 non-null int64
INCKEY             221266 non-null int64
COLDETKEY          221266 non-null int64
REPORTNO           221266 non-null object
STATUS             221266 non-null object
ADDRTYPE           217554 non-null object
INTKEY             71823 non-null float64
LOCATION           216680 non-null object
EXCEPTRSNCODE      100863 non-null object
EXCEPTRSNDESC      11775 non-null object
SEVERITYCODE       221265 non-null object
SEVERITYDESC       221266 non-null object
COLLISIONTYPE      194767 non-null object
PERSONCOUNT        221266 non-null int64
PEDCOUNT           221266 non-null int64
PEDCYLCOUNT        221266 non-null int64
VEHCOUNT           221266 non-null int64
INJURIES           221266 non-null int64
SERIOUSINJURIES    221266 non-null int64
FATALITIES         221266 non-null int64
INCDATE            221266 non-null object
INCDTTM            221266 non-null object
JUNCTIONTYPE       209299 non-null object
SDOT_COLCODE       221265 non-null float64
SDOT_COLDESC       221265 non-null object
INATTENTIONIND     30188 non-null object
UNDERINFL          194787 non-null object
WEATHER            194578 non-null object
ROADCOND           194658 non-null object
LIGHTCOND          194490 non-null object
PEDROWNOTGRNT      5188 non-null object
SDOTCOLNUM         127205 non-null float64
SPEEDING           9913 non-null object
ST_COLCODE         211853 non-null object
ST_COLDESC         194767 non-null object
SEGLANEKEY         221266 non-null int64
CROSSWALKKEY       221266 non-null int64
HITPARKEDCAR       221266 non-null object
dtypes: float64(5), int64(12), object(23)
memory usage: 67.5+ MB
```

Features ROADCOND, JUNCTIONTYPE, COLLISIONTYPE, HITPARKEDCAR, LIGHTCOND had values like “Uknown” or “Other” which do not clearly decribe any quality of the feature itself hence removed those records.

It was also found that many of the features had 'Y' or numberic '1' value to represent the existence of certain characteristics and 'N' was not given and for such records it was just left blank. For UNDERINFL, INATTENTIONIND, SPEEDING and PEDROWNOTGRNT all 'Y' values were converted to 1 and 'N' or blank were converted to number 0. This transformation of text to numberic will help avoid additional work need to be done in converting categorical values to numberic for machine learning.

There are two features associated with dates. One is to record incident date INCDATE and the other INCDTTM is to record date and time the incident happened. Both were not in date format so first converted them to date format. Through detailed analysis it was found that many of the did not have time recorded and just only given date. As it would have been costly to drop all these records, I found top 10 time values and used that with random seeding to populate missing values for all records in INCDTTM field. Once this clean up is completed INCDTTM feature converted to a date frame. After the conversion to date frame, created three new features to see correlation of incident with dayofweek, hourofday, month of the year.

I have also forced int type to features INATTENTIONIND, UNDERINFL, PEDROWNOTGRNT, SPEEDING using df.astype fuction as those fields were seen as object by the system.

After all this cleanup, I then used dropna function to remaining records with any features that have null values.

2.3 Feature Selection

After all the cleanup, there were 147802 samples and 43 features. Upon examining the meaning of each feture with the help of attribute discription, it was cear that there was some redundancy in the features. For example, INCDATE and INCDTTM. Date and Time field itself has the same date information so it was not necessary to keep the date only field. Similarly SEVERITYCODE and SEVERITYDESC both describing same data. As SEVERITYCODE had text value and needed encoding, I decided to keep on SEVERITYDESC. SEVERITYDESC is our target value.

Many other features with duplicate or derivative information were dropped as as well. Examples are EXCEPTRSNCODE and EXCEPTRSNDESC, SDOT_COLCODE and SDOT_COLDESC, ST_COLCODE and ST_COLDESC. After all this clean up I ended up with 36 features.

Another deep analysis of remaining features revealed to me that there were many other features that are created by SDOT for administrative purpose and they were not at all any features that could have contributed to accident and cause of the accident. These are examples of such features. 'SDOTCOLNUM', 'INTKEY', 'STATUS', 'REPORTNO', 'INCKEY', 'OBJECTID','COLDETKEY'.

There was no explanation given for EXCEPTRSNCODE in the attributes description document hence I dropped that feature as well.

The above evaluation and cleanup resulted in final 30 features on which I decided to do exploratory analysis.

Table 1. Simple feature selection during data cleaning

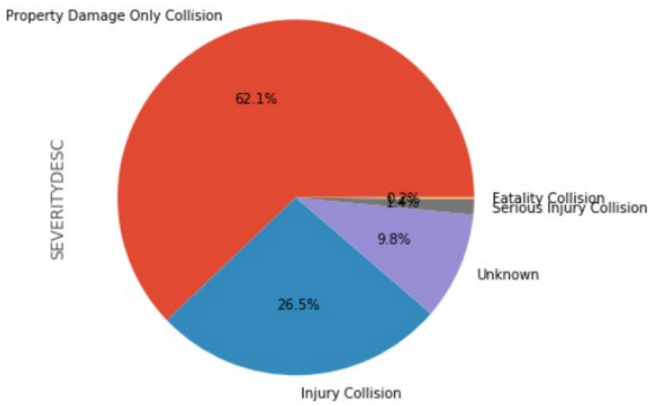
Kept Features	Dropped Features	Reason for Dropping Features
INCDTTM	INCDATE	As INCDTTM has date and time information, this is redundant
SEVERITYDESC	SEVERITYCODE	Target value. We only need to keep one as both representing same info
EXCEPTRSNCODE ST_COLDCODE	EXCEPTRSNDESC, ST_COLDESC	Duplicate features
	SDOTCOLNUM, INTKEY, STATUS, REPORTNO, INCKEY, OBJECTID,COLDETKEY, EXCEPTRSNCODE, SDOT_COLDESC	Administrative information added by SDOT
X, Y	LOCATION	Dropped location as X,Y geo coordinates refer to a location
Dayofweek, year, hr_sin, hr_cos, mnth_sin, mnth_cos	INCDTTM	Duplicate information

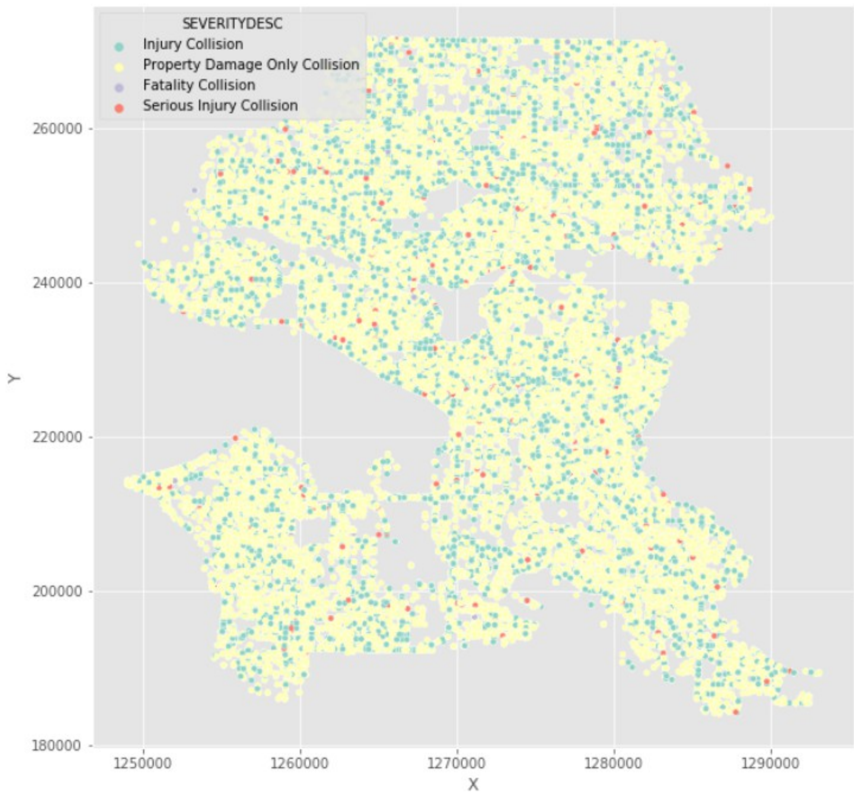
3. Exploratory Data Analysis

3.1 Target Value

As we have to predict severity of an incident, our target value is SEVERITYDESC. This is a categorical value and need special treatment prior to work with our models. We will deal with that little later.

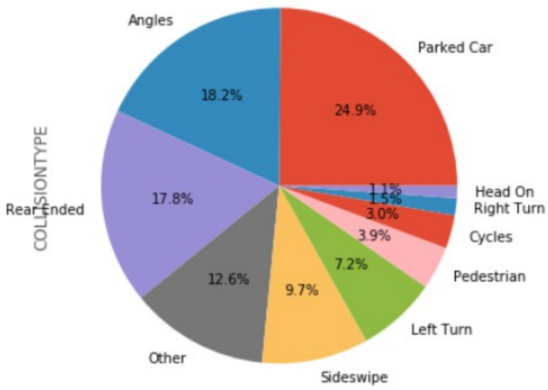
Within our target value there are 5 different type of severity recorded. Within this 'Unknown' is not useful hence we have removed that in the data cleanup. 62% of the accidents resulted in property damage and 26.5% resulted in injury collision.



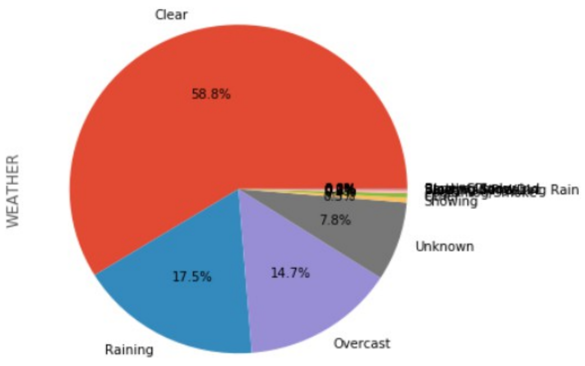


3.2 Key features and their contributions to the samples

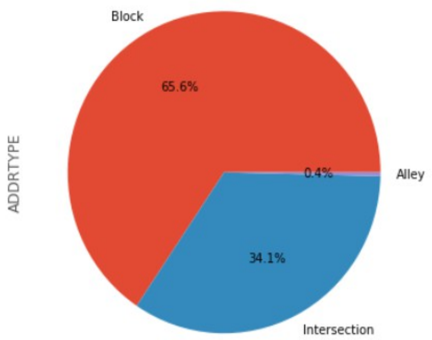
COLLISIONTYPE, majority of the accidents resulted with parked cars.



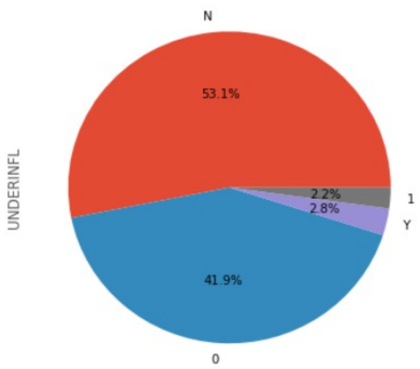
We can see that most of the accidents happen in clear weather condition.



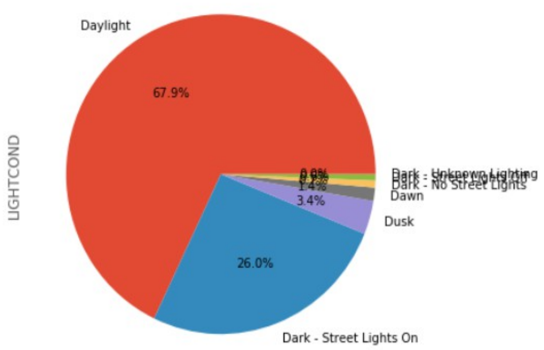
At what type of locations majority of the accidents happen? Majority of the accidents are happening at the block type of locations.



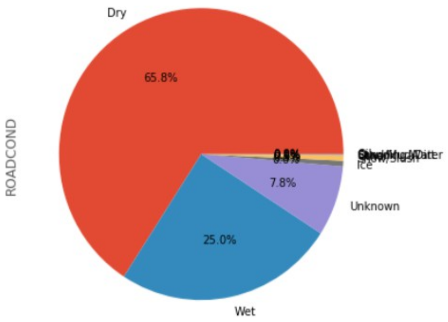
Only 5% of accidents happened under the influence of alcohol.



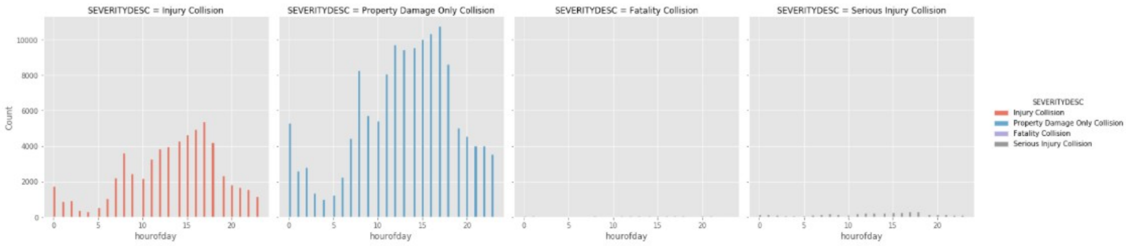
67% of accidents happened under good lighting conditions.



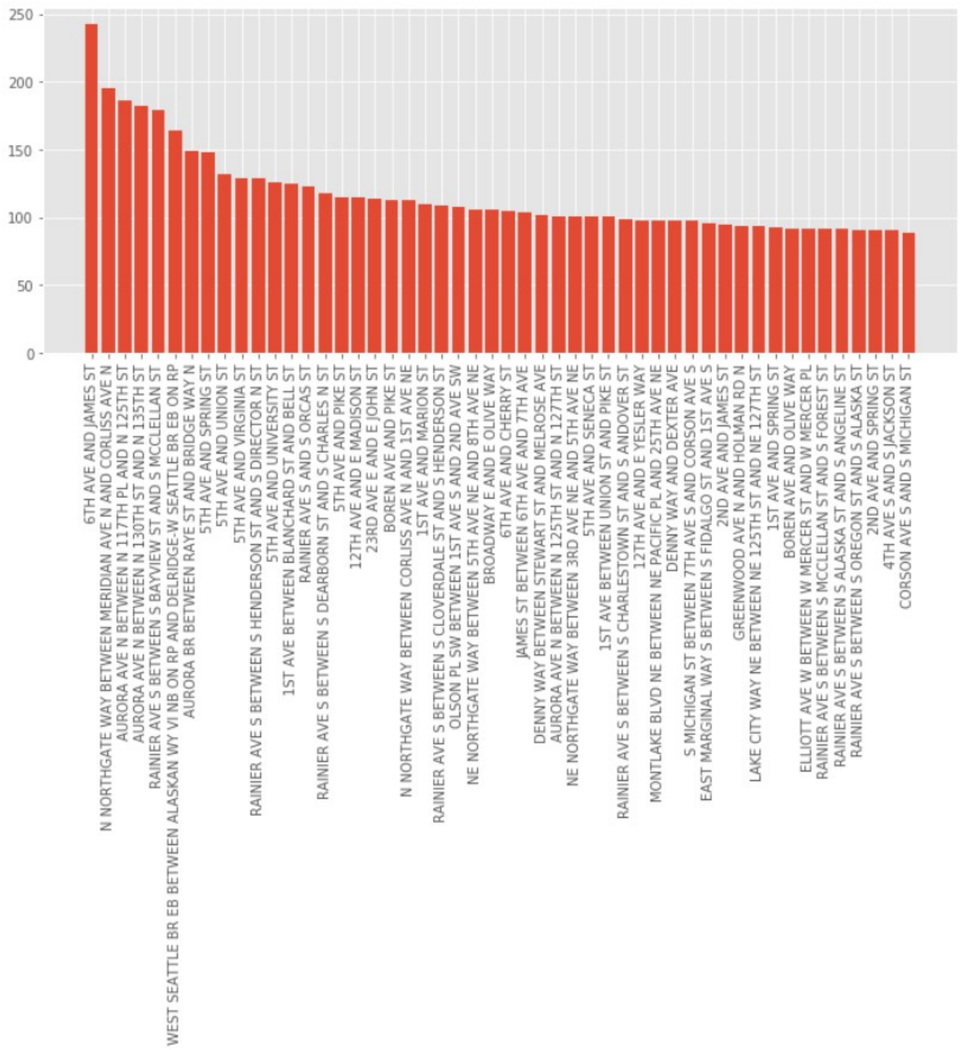
66% of accidents happened under dry road conditions.



Majortity of the accidents happen around 5PM time. Is it the peak time, tired people going back to home causing more accidents?



Do a particular location contribute more accidents?



4. Predictive Modeling

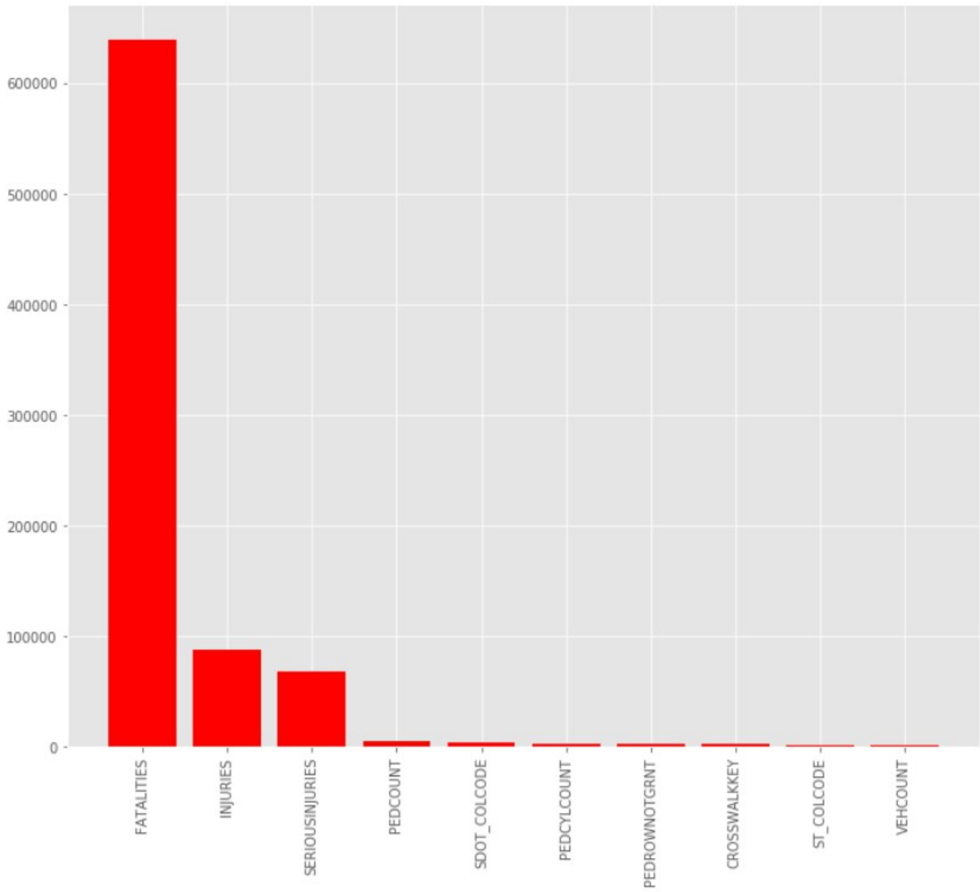
Severity prediction is a classification problem with categorical output. When we have an output of categorical type and input with both numeric and categorical features, we need to be using appropriate training models.

4.1 Feature refinement/reduction using statistical methods

Here I will take the approach of splitting our features into two Numeric and Categorical so that we can run additional statistical methods on those features to select best features that we need to include for training and testing.

I have used scikit learn's SelectKBest library to analyze and select top 10 features out of it. Out of 24 features supplied following top 10 features were selected.

Result of ANOVA Test F-Value of Top T10 Numeric Features against Categorical Target



Next we need look at the Categorical Features. Before we can run chi2 test on categorical Features we need to encode them to numeric using one hot encode feature. Six of the categorical features were encoded and run through chi2 test. Below is the result of chi2 test.

```
Feature 0: Block: 2171.505057
Feature 1: Intersection: 3226.379038
Feature 2: Angles: 538.267139
Feature 3: Cycles: 7485.204756
Feature 4: Head On: 268.372183
Feature 5: Left Turn: 217.891963
Feature 6: Parked Car: 10337.235636
Feature 7: Pedestrian: 13017.240829
Feature 8: Rear Ended: 1542.854209
Feature 9: Right Turn: 224.482049
Feature 10: Sideswipe: 3355.696075
Feature 11: At Intersection (but not related to intersection): 3.002649
Feature 12: At Intersection (intersection related): 3312.125511
Feature 13: Driveway Junction: 8.629904
```

```
Feature 14: Mid-Block (but intersection related): 112.572272
Feature 15: Mid-Block (not related to intersection): 2956.614423
Feature 16: Ramp Junction: 0.281486
Feature 17: Blowing Sand/Dirt: 1.036267
Feature 18: Clear: 3.692209
Feature 19: Fog/Smog/Smoke: 7.924677
Feature 20: Overcast: 5.737416
Feature 21: Partly Cloudy: 1.820709
Feature 22: Raining: 31.040929
Feature 23: Severe Crosswind: 39.550899
Feature 24: Sleet/Hail/Freezing Rain: 1.563449
Feature 25: Snowing: 65.343436
Feature 26: Dry: 5.912757
Feature 27: Ice: 27.246070
Feature 28: Oil: 1.124540
Feature 29: Other: 2.185593
Feature 30: Sand/Mud/Dirt: 1.033507
Feature 31: Snow/Slush: 74.719975
Feature 32: Standing Water: 0.479230
Feature 33: Wet: 27.202447
Feature 34: Dark - No Street Lights: 44.131461
Feature 35: Dark - Street Lights Off: 15.862539
Feature 36: Dark - Street Lights On: 104.564506
Feature 37: Dark - Unknown Lighting: 1.604267
Feature 38: Dawn: 9.931060
Feature 39: Daylight: 53.460964
Feature 40: Dusk: 2.238798
```

Though ADDRESSTYPE, COLLISIONTYPE, and JUNCTIONTYPE gave highest scores, I decided to keep all categorical fields in this modeling.

So my final features included Top 10 Numerical features and all categorical features.

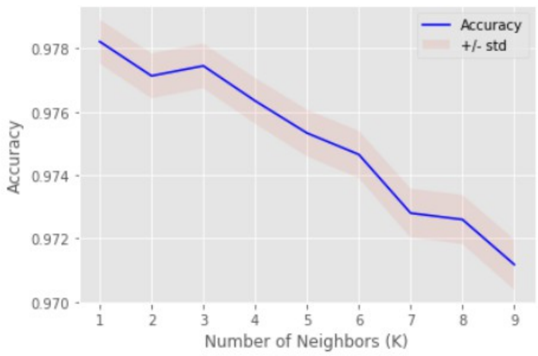
4.2 Modeling

After all feature fine tuning, I have ended up with 147,802 samples and 51 Features. Our data is unbalanced, hence I have decided to run three different types of modeling. 1. Modeling with original unbalanced data, 2. Modeling with under-sampled data and finally modeling with over-sampled data.

4.2.1. Modeling with original unbalanced data

4.2.1.1. K Nearest Neighbour (KNN)

K1 provided the best result and the modeling accuracy was 97.8



	precision	recall	f1-score	support
Fatality Collision	0.51	0.26	0.35	68
Injury Collision	0.97	0.97	0.97	14819
Property Damage Only Collision	0.99	0.99	0.99	28706
Serious Injury Collision	0.84	0.63	0.72	748
accuracy			0.98	44341
macro avg	0.83	0.71	0.76	44341
weighted avg	0.98	0.98	0.98	44341

4.2.1.2. Support Vector Machine with RBF Kernel

SVM only could provide model of 67.71% accuracy.

	precision	recall	f1-score	support
Fatality Collision	0.00	0.00	0.00	68
Injury Collision	0.82	0.07	0.12	14819
Property Damage Only Collision	0.66	1.00	0.80	28706
Serious Injury Collision	0.00	0.00	0.00	748
accuracy			0.67	44341
macro avg	0.37	0.27	0.23	44341
weighted avg	0.70	0.67	0.56	44341

4.2.1.3. Logistic Regression

Logistic regression model with liblinear algorithm provided 98.01 accuracy.

	precision	recall	f1-score	support
Fatality Collision	1.00	0.49	0.65	68
Injury Collision	0.95	0.99	0.97	14819
Property Damage Only Collision	1.00	1.00	1.00	28706
Serious Injury Collision	1.00	0.01	0.03	748
accuracy			0.98	44341
macro avg	0.99	0.62	0.66	44341
weighted avg	0.98	0.98	0.97	44341

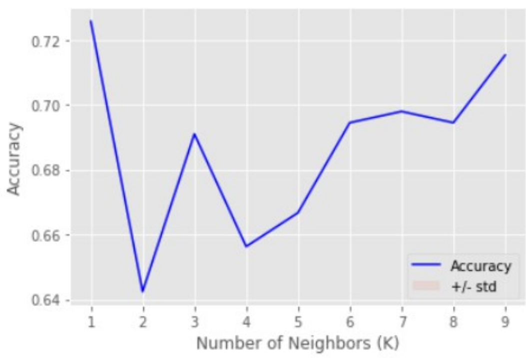
4.2.1. Modeling with under-sampled data

With the under-sampling method, samples were considerably reduced to 240 samples only.

I have used 70:30 ratio for training and testing samples.

4.2.1.1. K Nearest Neighbor

With the under-sampling KNN accuracy went down to 72.5%



	precision	recall	f1-score	support
Fatality Collision	1.00	1.00	1.00	70
Injury Collision	1.00	1.00	1.00	72
Property Damage Only Collision	1.00	1.00	1.00	71
Serious Injury Collision	1.00	1.00	1.00	75
accuracy			1.00	288
macro avg	1.00	1.00	1.00	288
weighted avg	1.00	1.00	1.00	288

4.2.1.1. Support Vector Machine

SVM model performed very poorly with under-sampled data. Accuracy was only 29.8%

	precision	recall	f1-score	support
Fatality Collision	0.54	0.21	0.31	70
Injury Collision	0.00	0.00	0.00	72
Property Damage Only Collision	0.27	1.00	0.43	71
Serious Injury Collision	0.00	0.00	0.00	75
accuracy			0.30	288
macro avg	0.20	0.30	0.18	288
weighted avg	0.20	0.30	0.18	288

4.2.1.1. Logistic Regression

Logistic regression model with liblinear algorithm provided 79% accuracy.

	precision	recall	f1-score	support
Fatality Collision	0.96	0.96	0.96	70
Injury Collision	0.64	0.50	0.56	72
Property Damage Only Collision	0.69	0.72	0.70	71
Serious Injury Collision	0.84	0.99	0.91	75
accuracy			0.79	288
macro avg	0.78	0.79	0.78	288
weighted avg	0.78	0.79	0.78	288

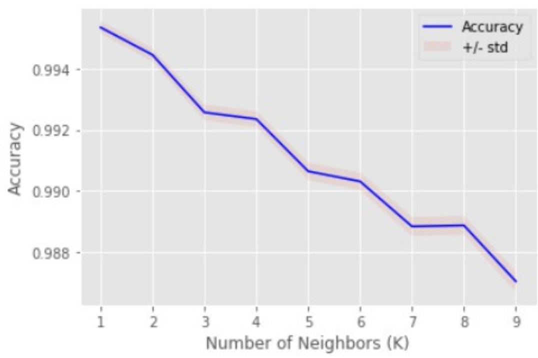
4.2.1. Modeling with over-sample data

With the help of RandomOverSampler library from Scikit Learn, resampled the samples to balance the samples. Resampling produced a total of 267,957 samples for our training. This is a huge dataset to handle. Some of the models below ran for days in my personal computer.

I have used 70:30 ratio for training and testing of our samples.

4.2.1.1. K Nearest Neighbors

With K-Nearest Neighbors classifier on our over-sampled dataset, model accuracy has improved and achieved 99.5% accuracy when K=1. When increased the K, discovered that the accuracy also decreased.



	precision	recall	f1-score	support
Fatality Collision	1.00	1.00	1.00	28595
Injury Collision	0.99	0.99	0.99	28750
Property Damage Only Collision	0.99	0.99	0.99	28778
Serious Injury Collision	1.00	1.00	1.00	28716
accuracy			1.00	114839
macro avg	1.00	1.00	1.00	114839
weighted avg	1.00	1.00	1.00	114839

4.2.1.1. Support Vector Machine (SVM) on Over-sample

To my surprise, SVM performed badly on over-sampled samples and could only produce a model with 30% accuracy.

	precision	recall	f1-score	support
Fatality Collision	0.52	0.22	0.31	28595
Injury Collision	0.00	0.00	0.00	28750
Property Damage Only Collision	0.28	1.00	0.44	28778
Serious Injury Collision	0.00	0.00	0.00	28716
accuracy			0.30	114839
macro avg	0.20	0.30	0.19	114839
weighted avg	0.20	0.30	0.19	114839

4.2.1.1. Logistic Regression

Surprisingly logistic regression did took only less than a minute to run the model on over-sampled dataset. Accuracy achieved is 76.42% which is less than unbalanced and under-sampled dataset.

	precision	recall	f1-score	support
Fatality Collision	1.00	0.99	1.00	28595
Injury Collision	0.54	0.65	0.59	28750
Property Damage Only Collision	0.65	0.46	0.54	28778
Serious Injury Collision	0.87	0.97	0.91	28716
accuracy			0.76	114839
macro avg	0.76	0.76	0.76	114839
weighted avg	0.76	0.76	0.76	114839

4.3 Performance of different modeling

From the below model performance comparison, Logistic Regression model performed best with the unbalanced and balanced (under-sampled) dataset while K-Nearest Neighbors performed best on balanced (over-sampled) dataset.

In all comparison, Logistic Regression provided fastest computing performance.

Sample Type \ Model Used	KNN	SVM	Logistic Regression
Unbalanced	97.80%	66.71%	98.01%
Balanced (under-sampled)	72.56%	29.86%	79.16%
Balanced (Over-sampled)	99.53%	30.39%	76.42%

5. Conclusions

In this capstone project, I was able to build a accident severity prediction model using different classification models such as KNN, SVM and Logistic regression. Compared performance of each models on balanced and unbalanced dataset.

6. Future Directions

I would further work on analyzing the final features used and see whether there is any additional features need to be reduced so that for a practical deployment the number of inputs to be given for model prediction can be reduced to minimum and still get better accuracy in the prediction.