Background

According to WHO, the number of road traffic deaths rising steadily up to 1.35 million in 2016. It is the 8th leading cause of death, less likely to survive than AIDs.

Problem

Prevention is always better than cure. This project is to predict the injury in car accidents. This is concerned by drivers for the purchase of insurance as well as the insurance company to adjust the insurance premium and the claim. On top of those, minimize the injury is most important purpose.

Data sources

The data I will us is obtained from this class's, provided by SDOT Traffic Management Division, Traffic Records Group, from 2004 to Present. The dataset comes with many factors that may affect the probability of accidents: Geographic data, accident details, timestamp, road types at which accident happened like intersection, driver behavior, weather, road condition, light condition.

Data cleaning and Feature selection

There were 194673 samples and 38 features in the data. Looking into the features, some id-type features don't help the analysis and thus dropped out. Duplicated and details of the crashes will also be removed because we are focusing on the injury only. As such I picked road types at which accident happened, driver behavior, weather, road condition, light condition.

In other words,

```
SEVERITYCODE       int64
INATTENTIONIND    object
UNDERINFL         object
WEATHER           object
ROADCOND          object
LIGHTCOND         object
SPEEDING          object
HITPARKEDCAR      object
PEDCYLCOUNT        int64
```

Where

1)Severity code is the prediction group,

2)inattentionind is the driver attention status,

3) underinfl is the status that driver is involved in drugs and alcohol

4) Weather, road and Light condition

5) whether the driver was speeding during the accident

6) hitparkedcar is whether the accident happened in car park

7) pedcylcount is whether bicycle involved in the accident

Data understanding and visualization

A code that corresponds to the severity of the collision:

- 3—fatality
- 2b—serious injury
- 2—injury
- 1—prop damage
- 0—unknown

But in the dataset, only 1 or 2 occurs and the portion is unbalanced:

```
1    136485
2     58188
Name: SEVERITYCODE, dtype: int64
```

To do a fair training, resampling is required:

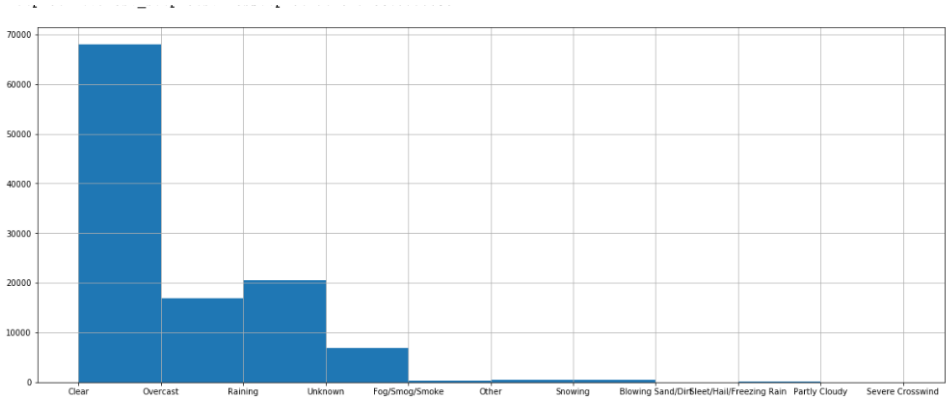
```
2    58188
1    58188
Name: SEVERITYCODE, dtype: int64
```

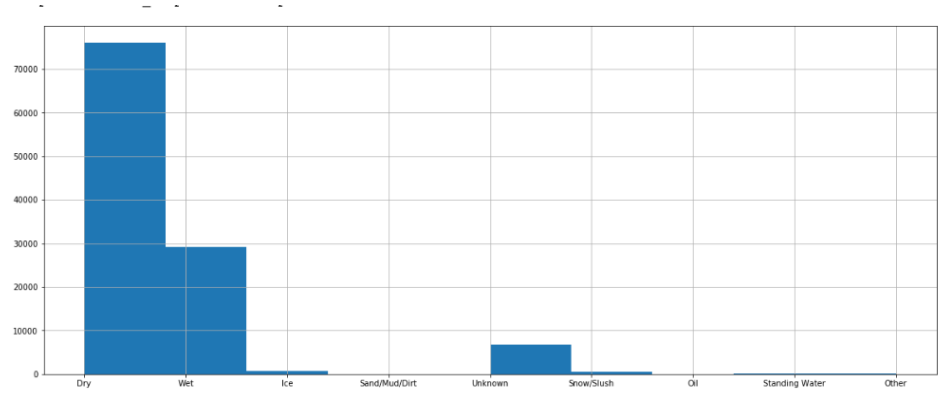
SPEEDING, INATTENTIONIND, UNDERINFL, HITPARKEDCAR, PEDCYLCOUNT are all Booleans but the raw data consist of non-numeric and empty cells and being cleaned as either 1 or 0 for training.

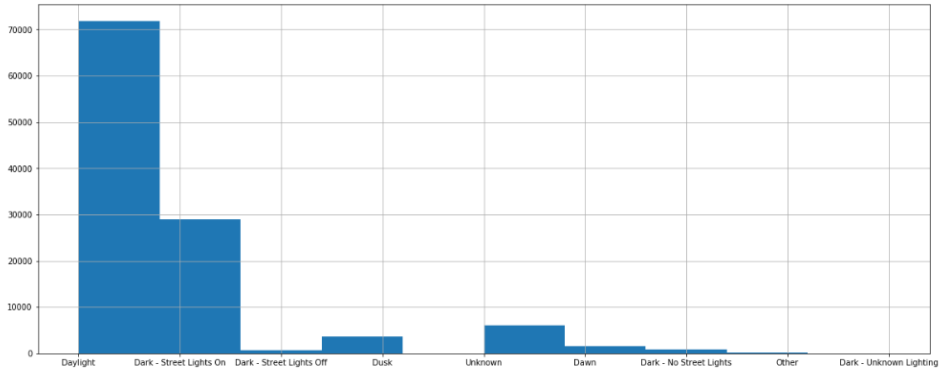
	SPEEDING	INATTENTIONIND	UNDERINFL	HITPARKEDCAR	PEDCYLCOUNT
186329	1	0	0.0	1	0
87783	0	0	0.0	0	0
113525	0	0	0.0	0	0
6518	0	0	0.0	0	0
19850	0	0	0.0	0	0
...	...	...	...	...	...
139109	0	0	0.0	0	0
22076	0	0	0.0	0	0
162883	0	1	0.0	1	0
162005	0	0	0.0	0	0
91428	0	0	0.0	0	0

116376 rows × 5 columns

Next, we visualize the weather, road and light condition:







As we can see, the above features don't give much impact on the likelihood of accident.

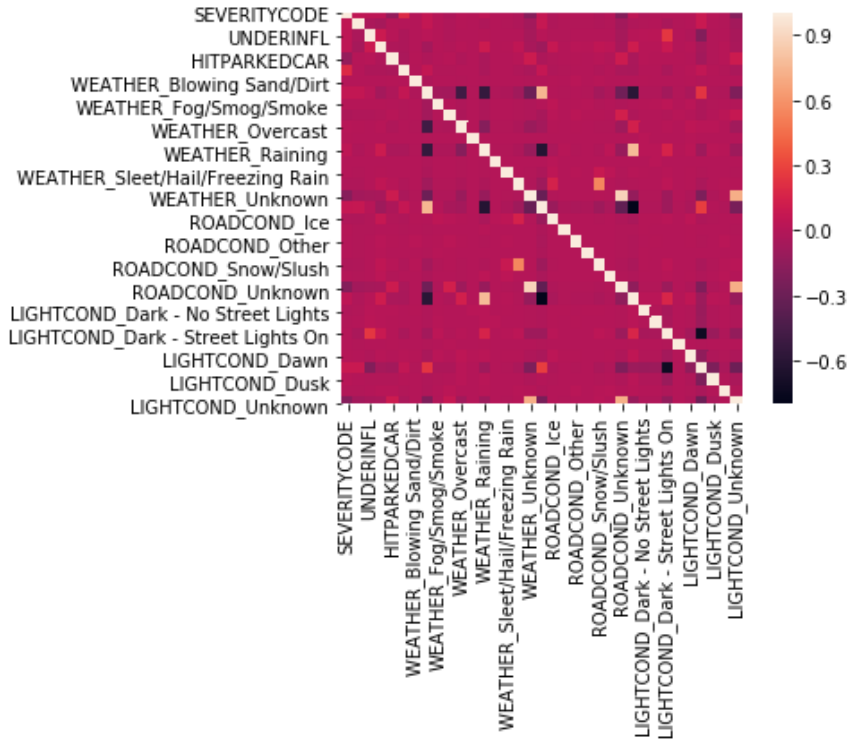
The top outcomes were clear weather, dry road, and daylight, in contrast with common sense.

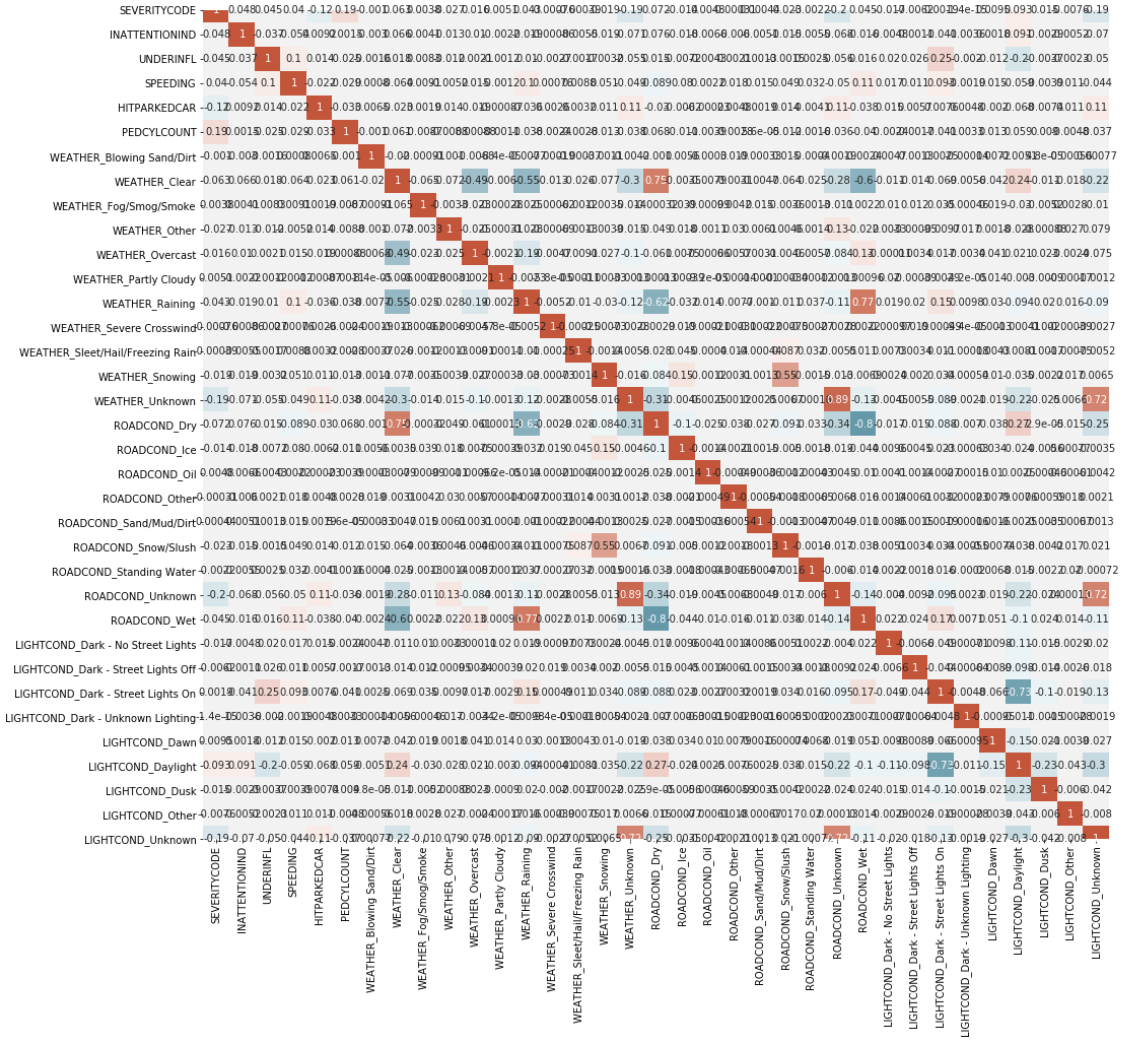
Dummy Variables

To do the training, dummy variables are used to replace above 3 features, which ends up:

['SEVERITYCODE', 'INATTENTIONIND', 'UNDERINFL', 'SPEEDING', 'HITPARKEDCAR', 'PEDCYLCOUNT', 'WEATHER_Blowing Sand/Dirt', 'WEATHER_Clear', 'WEATHER_Fog/Smog/Smoke', 'WEATHER_Other', 'WEATHER_Overcast', 'WEATHER_Partly Cloudy', 'WEATHER_Raining', 'WEATHER_Severe Crosswind', 'WEATHER_Sleet/Hail/Freezing Rain', 'WEATHER_Snowing', 'WEATHER_Unknown', 'ROADCOND_Dry', 'ROADCOND_Ice', 'ROADCOND_Oil', 'ROADCOND_Other', 'ROADCOND_Sand/Mud/Dirt', 'ROADCOND_Snow/Slush', 'ROADCOND_Standing Water', 'ROADCOND_Unknown', 'ROADCOND_Wet', 'LIGHTCOND_Dark - No Street Lights', 'LIGHTCOND_Dark - Street Lights Off', 'LIGHTCOND_Dark - Street Lights On', 'LIGHTCOND_Dark - Unknown Lighting', 'LIGHTCOND_Dawn', 'LIGHTCOND_Daylight', 'LIGHTCOND_Dusk', 'LIGHTCOND_Other', 'LIGHTCOND_Unknown']

Checking Correlations


SEVERITYCODE
UNDERINFL
HITPARKEDCAR
WEATHER_Blowing Sand/Dirt
WEATHER_Fog/Smog/Smoke
WEATHER_Overcast
WEATHER_Raining
WEATHER_Sleet/Hail/Freezing Rain
WEATHER_Unknown
ROADCOND_Ice
ROADCOND_Other
ROADCOND_Snow/Slush
ROADCOND_Unknown
LIGHTCOND_Dark - No Street Lights
LIGHTCOND_Dark - Street Lights On
LIGHTCOND_Dawn
LIGHTCOND_Dusk
LIGHTCOND_Unknown
0.9
0.6
0.3
0.0
−0.3
−0.6



Modeling and results

We will use the following models:

- **K-Nearest Neighbor (KNN)**

KNN will help us predict the severity code within k distance and look for similar data points

- **Decision Tree**

A decision tree model gives us a layout of all possible outcomes so we can fully analyze the factors of a decision. Decision tree will look in all possibility in combination of all features

- **Logistic Regression**

Because our dataset only provides us with two severity code outcomes, our model will only predict one of those two classes, which is the job of Logistic Regression.

To begin, I'm going to split 20% testing and 80% training set

Results:

KNN:

```
k: 1 f1: 0.5226053208824671 jaccard_score: 0.355622571478281
k: 5 f1: 0.5553165015792468 jaccard_score: 0.31457922307639974
k: 10 f1: 0.5605243150341689 jaccard_score: 0.3553231939163498
```

My laptop is not capable to handle many k iterations, so I pick k=1,5,10 only

K=10 scores highest on f1 score

Decision Tree

```
d: 1 f1: 0.412432257149831 jaccard_score: 0.5139237480578464
d: 2 f1: 0.4383288255980701 jaccard_score: 0.1006062685884237
d: 3 f1: 0.4713584327963392 jaccard_score: 0.1385340373587691
d: 4 f1: 0.4835567013911636 jaccard_score: 0.15347654041831543
d: 5 f1: 0.4834394264300292 jaccard_score: 0.15314550803349175
d: 6 f1: 0.4832896422100986 jaccard_score: 0.15339432960578336
d: 7 f1: 0.5625511418198904 jaccard_score: 0.316254744052578
d: 8 f1: 0.5553869567802918 jaccard_score: 0.2859655121879732
d: 9 f1: 0.5560758809294956 jaccard_score: 0.2871577097671694
d: 10 f1: 0.5574280325964162 jaccard_score: 0.2901483823887132
d: 11 f1: 0.5576194221995643 jaccard_score: 0.2907208477130219
d: 12 f1: 0.558679378422088 jaccard_score: 0.29269594758553746
d: 13 f1: 0.556558927642541 jaccard_score: 0.28548268238761976
d: 14 f1: 0.5578615416804503 jaccard_score: 0.2886431472329598
d: 15 f1: 0.5645481480573207 jaccard_score: 0.3109862612718946
d: 16 f1: 0.5645276942806187 jaccard_score: 0.3114212958596624
d: 17 f1: 0.5646620297407514 jaccard_score: 0.311843159432584
d: 18 f1: 0.5646064263416588 jaccard_score: 0.3117311589762926
d: 19 f1: 0.5647169753527681 jaccard_score: 0.31258527405316394
d: 20 f1: 0.5649314384391231 jaccard_score: 0.31218776510509944
```

Iterated up to 20 depths and d=1 scored most, 51% of jaccard score

Logistic Regression

```
C: 1 solver: lbfgs log_loss: 0.6431721123010561
C: 1 solver: saga log_loss: 0.6431608485235721
C: 1 solver: liblinear log_loss: 0.6431609093458044
C: 1 solver: newton-cg log_loss: 0.6431609039013503
C: 1 solver: sag log_loss: 0.6431602624844622
C: 0.1 solver: lbfgs log_loss: 0.6431412805537222
C: 0.1 solver: saga log_loss: 0.6431402838025269
C: 0.1 solver: liblinear log_loss: 0.6431400618837683
C: 0.1 solver: newton-cg log_loss: 0.6431399232443681
C: 0.1 solver: sag log_loss: 0.6431403744594415
C: 0.01 solver: lbfgs log_loss: 0.6441853598311507
C: 0.01 solver: saga log_loss: 0.6441849016916386
C: 0.01 solver: liblinear log_loss: 0.6441802023142388
C: 0.01 solver: newton-cg log_loss: 0.6441847076825 79
C: 0.01 solver: sag log_loss: 0.6441853825031765
```

LR gives a better result but the tuning of hyperparameters C and solvers does not change the accuracy much, all are around 64%. So, any C and solver will work

Discussion

The entire dataset is categorical and True values of each features didn't occur frequency in the dataset. This may be one of the reasons why the accuracy is low in every model. I would look for alternative dataset in Kaggle or other sets, adding back some of the dropped features like locations and time and redo the training. Other way that I come up with is dropping some features and see if the features are overwhelming. I failed to do SVM because it takes forever, I may need to get a better computer for training.

Conclusion

Models and data have great room to improve.

Will look for datasets with speed data at crush and more severity types for training.