Car accident severity prediction

Introduction: Business understanding.

In this project the goal is to perform an analysis on the car accident dataset provided by the Seattle Department of Transportation, which contains the records of all the accidents and collisions that have occurred from 2004 to present. The main focus here is to predict the severity of the collisions using features from the dataset which will help the machine learning model to predict its severity.

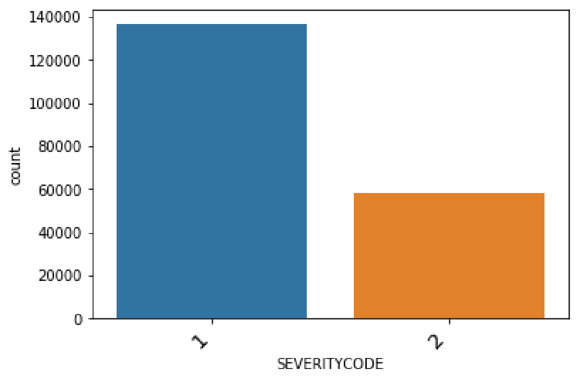
The Dataset

The Dataset used here is in the .csv format. The datasets contains all the collisions provided by SPD and recorded by Traffic Records in the city of Seattle. The dataset contains a total of 194673 entries spanning over 37 data columns.

Basic exploration of the dataset

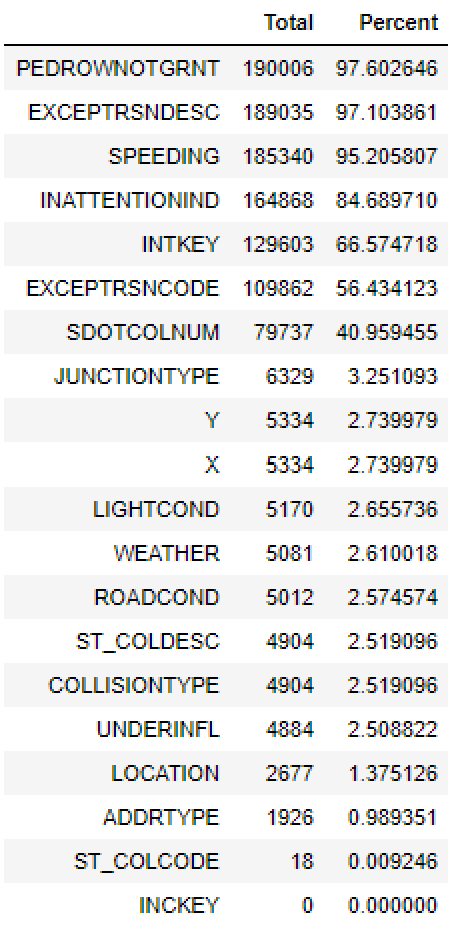
The dataset contains of 194673 entries from 2004 to present and is updated on a weekly basis by SDOT. The data spans over a total of 37 columns which are also called as features. Severity is the feature we are trying to predict and it is given in the SEVERITY CODE which denotes the severity of each of the accident that has occurred. The other features include the weather conditions, road conditions, light conditions, junction types, etc. We make use of these features to predict the car accident severity.

The dataset is an unbalanced dataset. The severity code counts present in the dataset are 1 and 2. Here we see the values are having a huge difference and in order to have a better accuracy and training we need to balance out this. So the need for some pre-processing is needed.



Missing values

Missing values often are a hurdle to get a good prediction or to perform an analysis task. Let us see the missing values percentage in this dataset



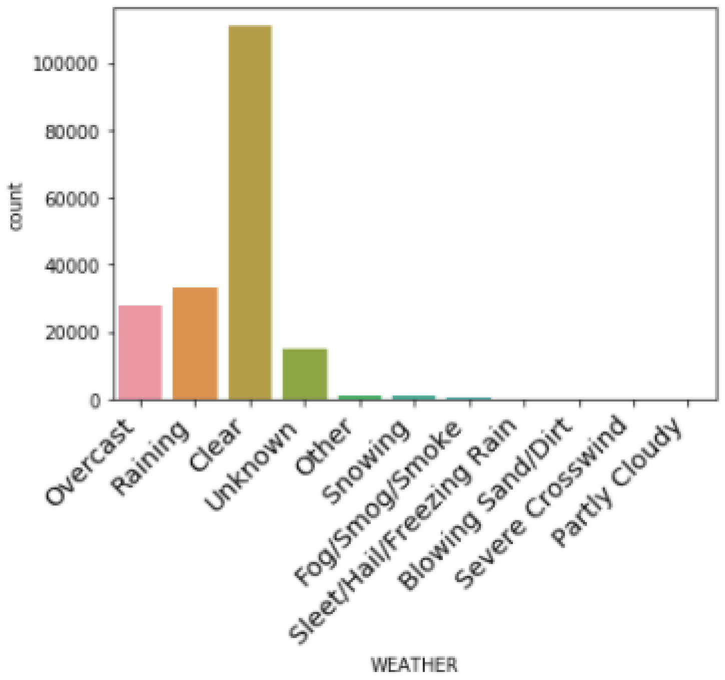
	Total	Percent
PEDROWNOTGRNT	190006	97.602646
EXCEPTRSNDESC	189035	97.103861
SPEEDING	185340	95.205807
INATTENTIONIND	164868	84.689710
INTKEY	129603	66.574718
EXCEPTRSNCODE	109862	56.434123
SDOTCOLNUM	79737	40.959455
JUNCTIONTYPE	6329	3.251093
Y	5334	2.739979
X	5334	2.739979
LIGHTCOND	5170	2.655736
WEATHER	5081	2.610018
ROADCOND	5012	2.574574
ST_COLDESC	4904	2.519096
COLLISIONTYPE	4904	2.519096
UNDERINFL	4884	2.508822
LOCATION	2677	1.375126
ADDRTYPE	1926	0.989351
ST_COLCODE	18	0.009246
INCKEY	0	0.000000

Here we can see that seven columns have missing values of over 40% and that is a lot missing data. We can simply omit these values since they will not have much significance in the prediction.

EDA

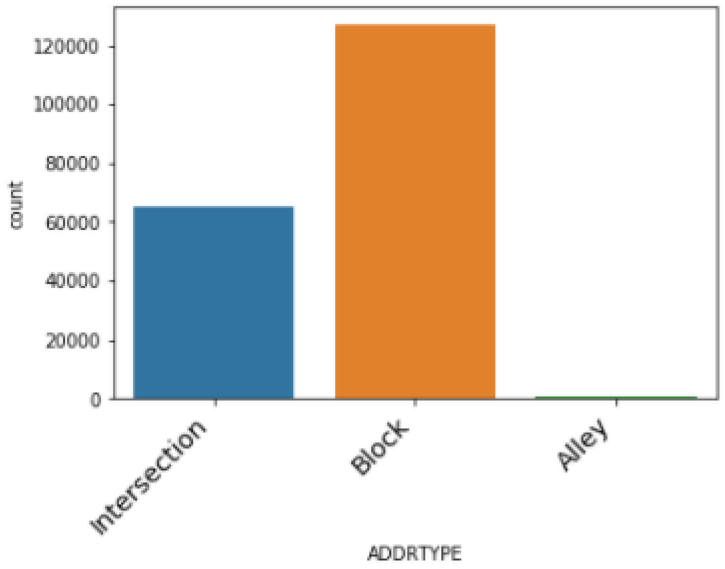
Some visualizations were we can get an idea of the features of the data

1) Count of the accidents taken in different weather conditions



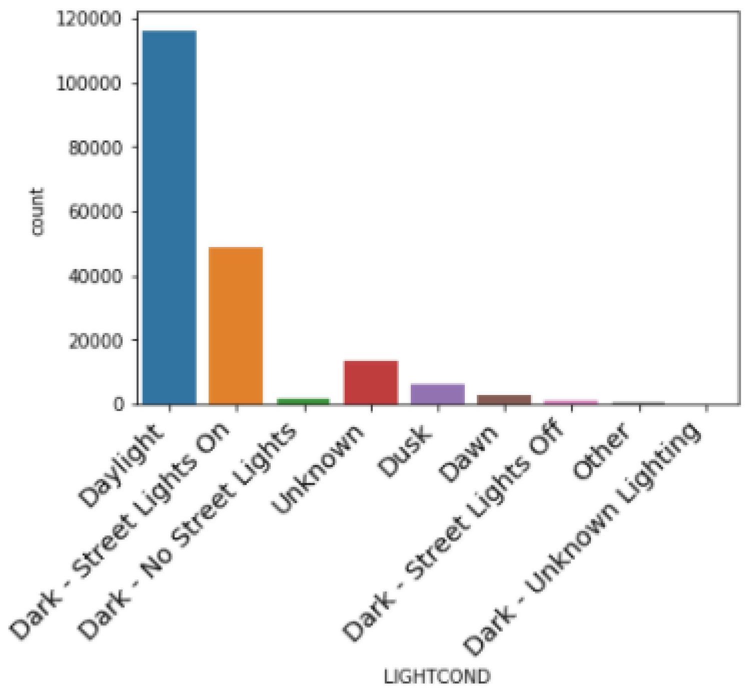
The highest number of collisions were taken place in clear weather.

2) Count of the accidents taken in different address types



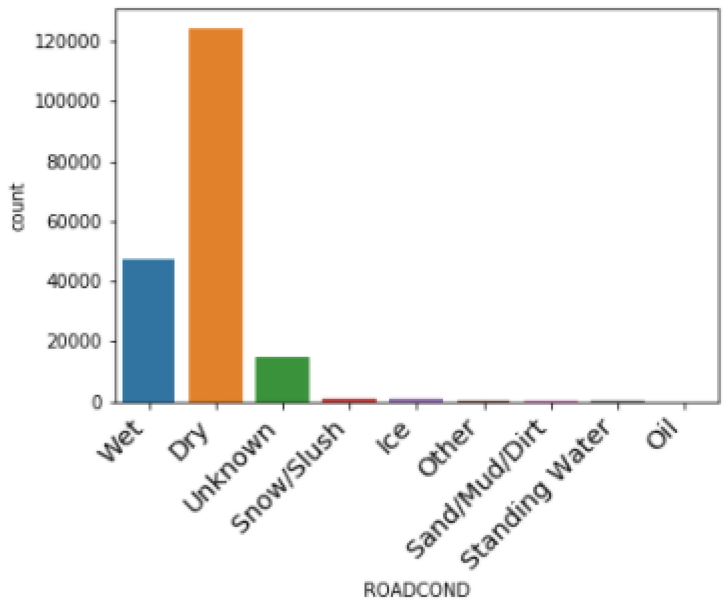
The highest number of collisions have taken in the block type.

3) Count of collisions in different light conditions



Highest collisions have taken place in daylight

4) Count of accidents in different road conditions



The highest collisions have taken place when the road was dry.

Balancing the dataset

Balancing the dataset is needed to adjust the values of the Severity code so that we don’t get a biased result. The difference between the 2 codes I very high as seen previously.

Feature selection

Going through the data, we see that there are 37 columns totally out of which seven were dropped due to the high missing value content in them. Looking at the rest of the data we also can see that mst of the features logically aren’t required to predict the severity like the coordinates, report number etc. So it is good to pick some features which will help in the better prediction of our data. The features used to predict the severity here are [ROADCOND','WEATHER','LIGHTCOND','ADDRTYPE','COLLISIONTYPE'].

Label encoding

All the features selected to predict the severity are of categorical type and we need numerical inputs for our machine learning model. So a good way to convert this categorical type of data to a machine learning model is by a method called label encoding. We use the label encoder function from the scikit learn library.

Splitting into test and training sets.

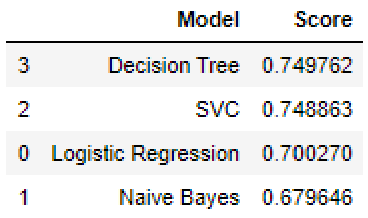
The train_test_split function from sklearn library is used for this. For this data, 80% is allotted to training whereas 20% is allotted to the test data.

Fitting the various ML models.

The algorithms I chose for this model were Logistic regression, Support vector clustering, Decision tree and Naive bayes. All of these were implemented by the use of the scikit-learn library. We got the classification report, the confusion Matrix and the accuracy score for each model. The models then were compared on basis of their accuracy scores.

Results

The following are the results obtained from training the different models.



	Model	Score
3	Decision Tree	0.749762
2	SVC	0.748863
0	Logistic Regression	0.700270
1	Naive Bayes	0.679646

- It is seen that Decision Tree and SVC have very similar accuracy scores and have performed the best compared to logistic regression and naive bayes.
- The accuracy range lies between 60%-70% which is not high, which suggests some amount of underfitting.

Conclusion

- The data had a lot of missing values which had to be dropped which might be the reason for underfitting of the data.
- A little more of feature engineering can be done to improve the overall accuracy of all the models.