Car accident severity study report
Building a model to predict severity from UK collisions
13-October-2020

Chris Wood

Introduction and business problem

The Department Of Transport (DOT) in the UK reported that in 2018 there were 1,784 deaths on UK roads and 25,511 injuries. This equates to around 5 deaths and 69 injuries per day. There is an obvious personal impact for those involved in accidents but there is also a cost on society. The DOT estimates that accidents cost €36bn per year or or 4.2% of 2018 total government expenditure.

We will look at data to see if we can use machine learning to develop a model that will predict the severity of accidents given real time data on influencing factors. With this model drivers can alter their behaviour or change their travel plans in order to reduce the risk of severe accidents. Policy makers may also be able to use this data to implement dynamic safety measures that will reduce the overall cost of accidents.

Who will find this report useful

Road users, insurance companies, emergency response teams, transport policy makers, healthcare providers, insurance providers.

Data Understanding

Data on collisions was obtained from the DOT website and contains all UK collisions between 2005 and 2014. The raw data contains information on circa 1.5m individual collisions including data on:

- collisions effects and severity
- where the collisions took place
  - Geographically
  - Type of road
- the environment where the incident took place
  - Weather, road condition

- If there were any other external factors influence the incident

. The data does not contain information on the total road usage

Dependent variables

The data contains the following potential dependent variables:

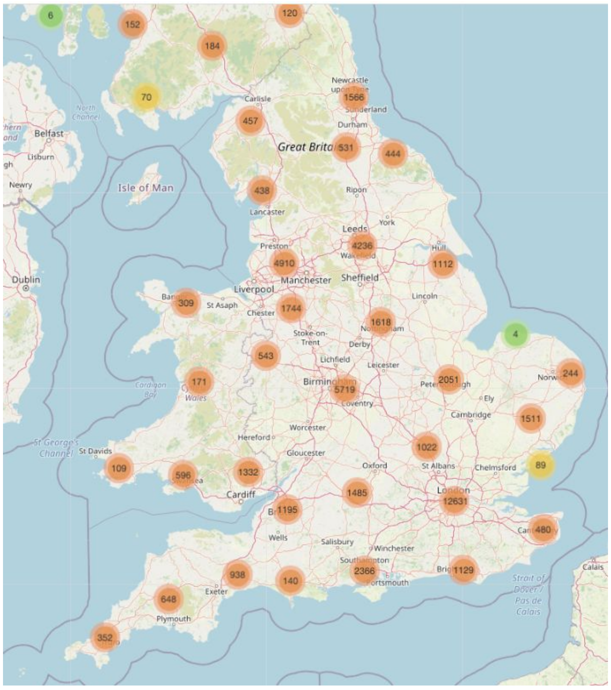
Field	Description
Accident_Severity	The overall Severity of the accident
Number_of_Vehicles	Number of vehicles involved in the collision
Number_of_Casualties	Number of injuries in the collision

We chose to focus on Accident_Severity as our main indicator since it is a derivative from the other two severity variables (Number_of_Vehicles and Number_of_Casualties).

The data contains the following potential dependent variables Independent variables

Field	Description
Urban_or_Rural_Area	Values of 1 or 2, 1 us urban, 2 is rural
Carriageway_Hazards	If the accident was caused by a hazard on the side of the road. Boolean resonises
Special_Conditions_at_Site	If the accident was caused by any special consitions. Boolean resonises
Road_Surface_Conditions	Describles the condition of the road at the time of the incident
Weather_Conditions	Describles the weather at the time of the incident
Light_Conditions	Describles the ligting at the time of the incident
Speed_limit	The speed limit of the road where the accident happened\
Day_of_Week	day of week that the accident happend
Longitude	Longitude of incident
Latitude	Latitude of incident
Date	Date if incident
Time	Time of incident

Location can be represented in the following way



Data Cleaning

Nulls and unknown entries

The data contained very few null points < 500 with no relationship. However in weather_conditions we have cira 60k values listed as “Unknown” or “Other” and within light_conditions we have around 16k values listed as “Unknown”. We will convert these points to NaN

The data set is large and the number of NaN rows is now around 4%. So we are judging that it is acceptable to just drop the NaN rows since we have no basis for replacing them.

Weather_Conditions

This field contains 8 string values which describe if the weather had rain, strong wind or snow. For example one entry is “Raining with high winds”. I decided to break the data in this column down into individual columns representing boolean responses.

So we created and filled 3 new boolean columns as follows:

- Rain
- Snow
- High_Winds

And then dropped the original weather_conditions column

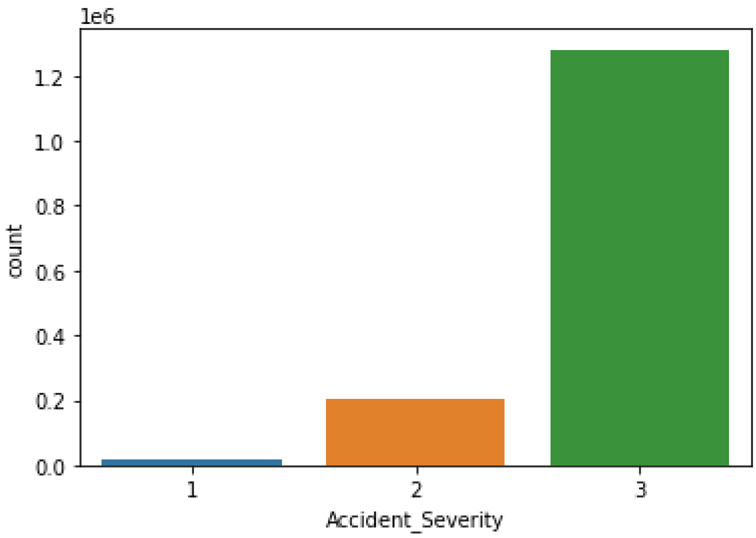
Urban_or_Rural_Area

This column contains circa 200 entries with value 3, the meta data shows that it should only contain two values so we will replace the 3s with NaN's and drop later.

Balancing data

It is important that our dependent variable is balanced. After initial observation we see that the row count per severity level is highly imbalance, as represented below:

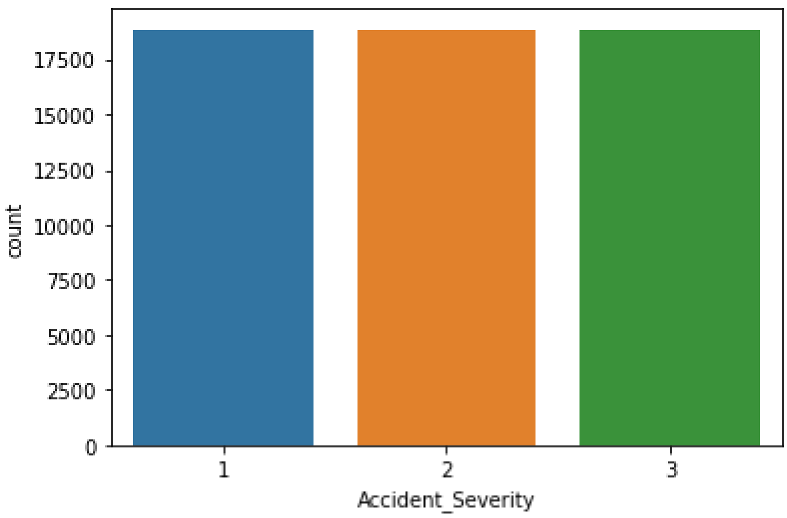
3 = 1,280,205 entries
2 = 204,504 entries
1 = 19,441 entries



Data needs to be balanced between the categories in order to improve the accuracy of the categorical ML models. We are using the imbalanced learn library, RandomUnderSampler to remove random rows across the dataframe.

We will balance data so we have an equal number of rows for each severity. This will greatly reduce the number of rows from circa 1.5m to circa 50k in the data set but it will also make the computing on a free platform more reasonable.

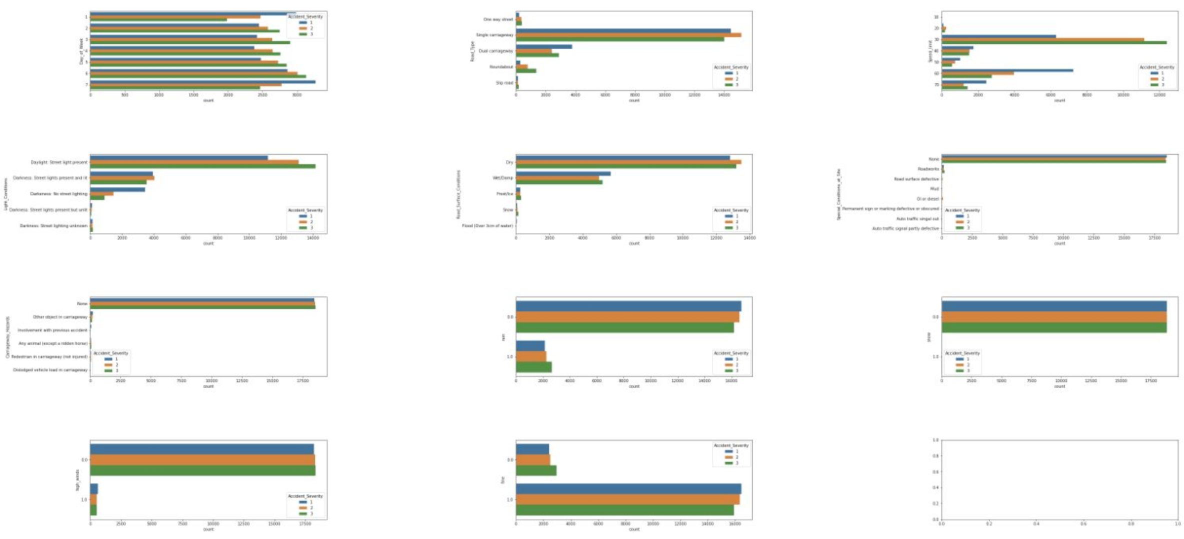
We can see the result of the undersampling



Feature Selection

Charting count values of variables

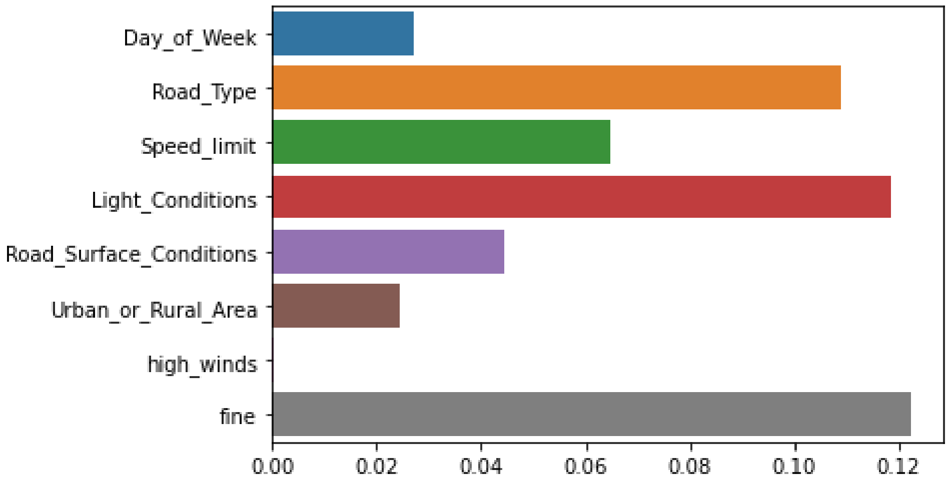
We want to select the features that will contribute most to building a robust model
Initially I plotted the count values for the options in each column, which can be seen below:



From the above charts we decided to drop Special_Conditions_at_Site, Carriageway_Hazards and snow columns as they add very little to the model. We can also drop rain as it is just the inverse of fine and their correlation is perfectly negative.

K best sklearn

We ran the best K method from skLearn to determine the relative importance of each of the remaining features and got the following output :



Based on this understanding we decided that the X variables should be:

- Speed_limit
- Urban_or_Rural_Area
- Light_Conditions
- Road_Type
- Fine
- Road_Surface_Conditions

Predictive modeling

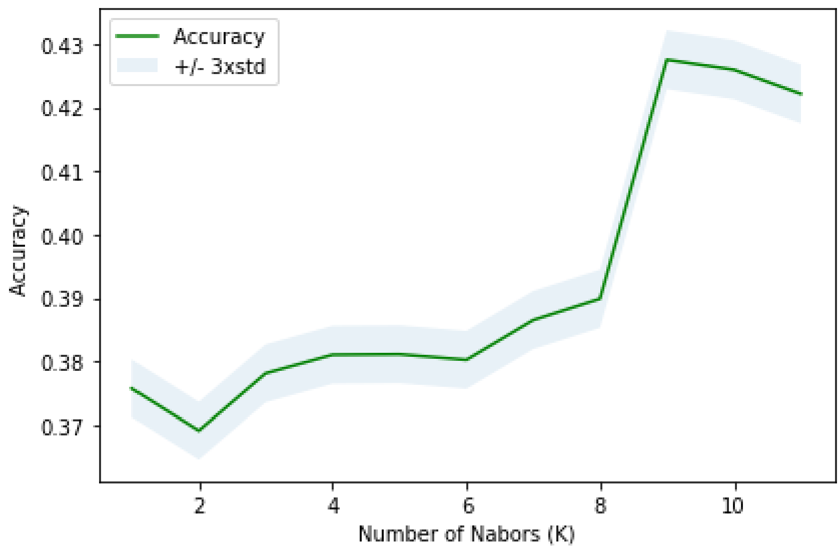
Firstly I will give an overview of the techniques used and will present conclusions of all models at the end of the report.

Preparing data

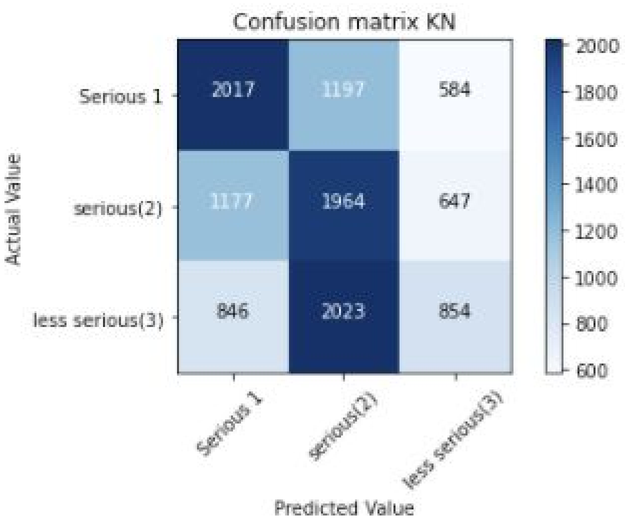
- We one hot encoded the categorical variables and normalised all the X values.
- We split data into train and test with a 80/20 ratio giving a train set of 45k and a test set of 11k.

Models considered

KNN
We established the best value for K by running the model on all k values between 1 and 12. We understand that 9 is the strongest K by a large margin.



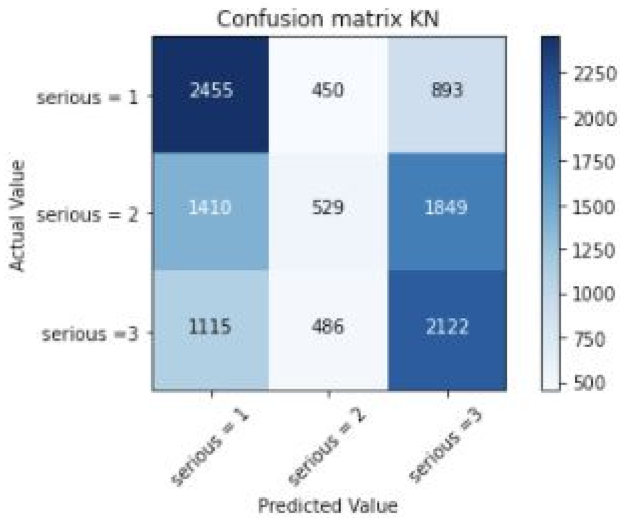

```
classification_report KNN
              precision    recall  f1-score   support

           1       0.50      0.53      0.51      3798
           2       0.38      0.52      0.44      3788
           3       0.41      0.23      0.29      3723

    accuracy                           0.43     11309
   macro avg       0.43      0.43      0.42     11309
weighted avg       0.43      0.43      0.42     11309

Confusion matrix, without normalization


[[2017 1197  584]
 [1177 1964  647]
 [ 846 2023  854]]
```



Decision Tree

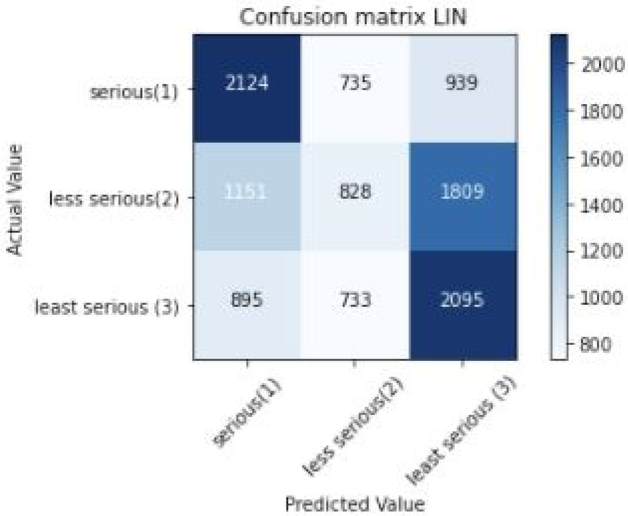
We ran a decision tree model with entropy and gini options to find the best accuracy. Gini provided marginally better accuracy via f-score.

```
classification_report Decision Tree
              precision    recall  f1-score   support

           1       0.49      0.65      0.56      3798
           2       0.36      0.14      0.20      3788
           3       0.44      0.57      0.49      3723

    accuracy                           0.45     11309
   macro avg       0.43      0.45      0.42     11309
weighted avg       0.43      0.45      0.42     11309

Confusion matrix, without normalization


[[2455  450  893]
 [1410  529 1849]
 [1115  486 2122]]
```



SVM

We ran a SVM model and considered the following kernels, linear, polynomial, RBF and Sigmoid. The most accurate average F score was from a linear model.

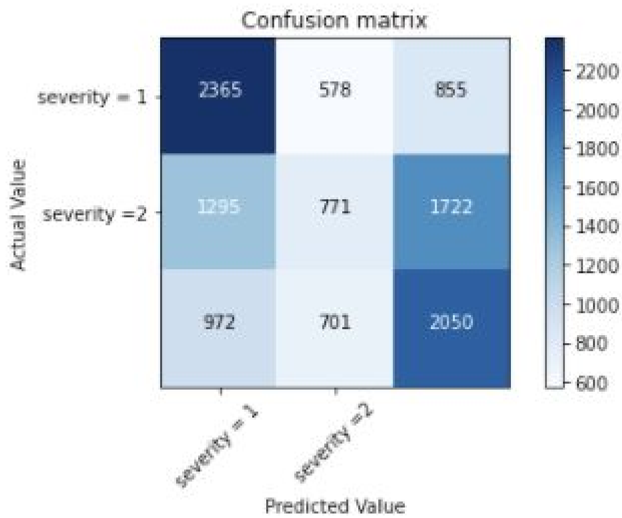


classification_report LIN

	precision	recall	f1-score	support
1	0.51	0.56	0.53	3798
2	0.36	0.22	0.27	3788
3	0.43	0.56	0.49	3723
accuracy			0.45	11309
macro avg	0.43	0.45	0.43	11309
weighted avg	0.43	0.45	0.43	11309

Logistic regression

```
Confusion matrix, without normalization


[[2365  578  855]
 [1295  771 1722]
 [ 972  701 2050]]
              precision    recall  f1-score   support

           1       0.51      0.62      0.56      3798
           2       0.38      0.20      0.26      3788
           3       0.44      0.55      0.49      3723

    accuracy                           0.46     11309
   macro avg       0.44      0.46      0.44     11309
weighted avg       0.44      0.46      0.44     11309
```



Model Selection

We can see from the following model evaluation metrics that logistic regression models give the most accuracy, however it is still low.

```
results on test data
               Precision    Recall  f1 score  accuracy
model
KNN             0.429411  0.429411  0.416304  0.427536
Decision Tree   0.430130  0.451499  0.418021  0.451499
Log Reg         0.443303  0.458573  0.438555  0.458573
SVM (Polly)     0.390536  0.438589  0.353165  0.438589


results on train data
               Precision    Recall  f1 score  accuracy
model
KNN             0.438780  0.438780  0.423332  0.434316
Decision Tree   0.440251  0.459122  0.425268  0.459122
Log Reg         0.447472  0.462261  0.442424  0.462261
SVM (Polly)     0.391541  0.442651  0.356577  0.442651
```

Chart of accuracy on test data

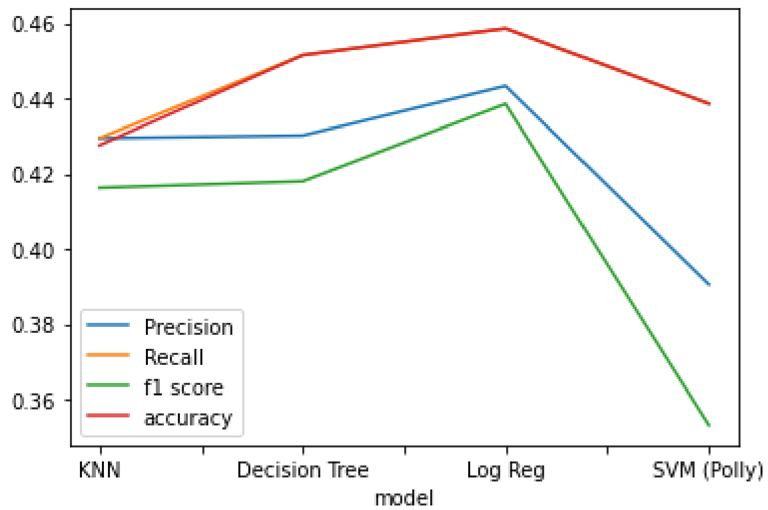
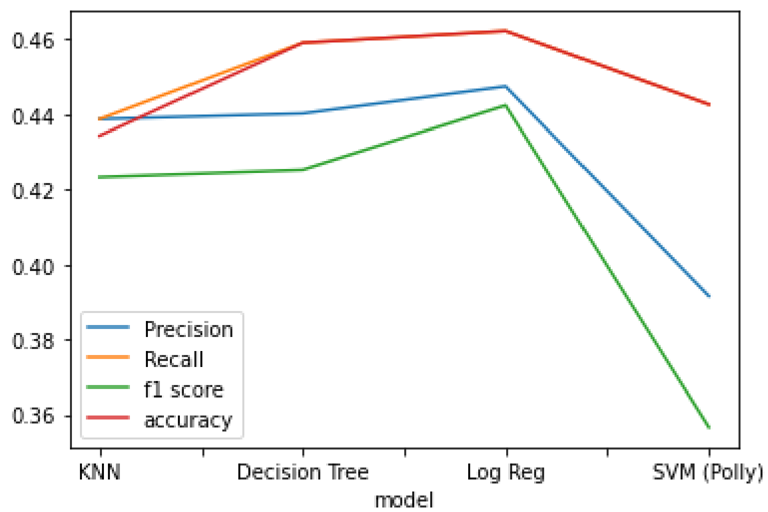


Chart of accuracy on train model



Discussion

1. By comparing the models on test and train data we can see that the model is not over or under fitted.
2. The model is not a good predictor for the severity of an accident since it predicts less than half the results correctly.
3. All the models were best at predicting true positives for the most sever accidents - level 1

Further work

1. Work needs to be done to compare the frequency of accidents relative to the vehicle KM's driven