# **Written Analysis**

# **Traffic Collision Prediction**

## Short Description: We are using KSI large dataset, and we are predicting whether the accident is fatal or not. We have cleaned, transformed, and preprocessed our data so that we can fit our data into different models for prediction.

## Dataset:

We have 57 columns and 18194 instances in the dataset.  
Column names:

['X', 'Y', 'INDEX\_', 'ACCNUM', 'YEAR', 'DATE', 'TIME', 'STREET1', 'STREET2', 'OFFSET', 'ROAD\_CLASS', 'DISTRICT', 'WARDNUM', 'LATITUDE', 'LONGITUDE', 'LOCCOORD', 'ACCLOC', 'TRAFFCTL', 'VISIBILITY', 'LIGHT', 'RDSFCOND', 'ACCLASS', 'IMPACTYPE', 'INVTYPE', 'INVAGE', 'INJURY', 'FATAL\_NO', 'INITDIR', 'VEHTYPE', 'MANOEUVER', 'DRIVACT', 'DRIVCOND', 'PEDTYPE', 'PEDACT', 'PEDCOND', 'CYCLISTYPE', 'CYCACT', 'CYCCOND', 'PEDESTRIAN', 'CYCLIST', 'AUTOMOBILE', 'MOTORCYCLE', 'TRUCK', 'TRSN\_CITY\_VEH', 'EMERG\_VEH', 'PASSENGER', 'SPEEDING', 'AG\_DRIV', 'REDLIGHT', 'ALCOHOL', 'DISABILITY', 'HOOD\_158', 'NEIGHBOURHOOD\_158', 'HOOD\_140', 'NEIGHBOURHOOD\_140', 'DIVISION', 'ObjectId']

We have 46 columns that are object and 11 columns that contains number.

#### Null Values:

We can see that we have a lot of null values in our dataset in almost all the columns (40 columns have null values). Column names with count of missing values are shown below:  
A screen shot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA black background with white text

Description automatically generated

### Data Findings:

We can see that most accidents are occurred in the evening time and in the morning time at the office times. The most accidents are in between the time frame of 5-8 pm.  
A graph of blue lines

Description automatically generated

According to the data, most fatal accidents occurred on Friday weekend when people go out after office hours for party. Also, we can see that most non-fatal accidents happened on Friday and Tuesday because on Tuesday most of the people go to the office who are doing hybrid jobs.  
A graph of a number of days and months

Description automatically generated

We can see that most of the accidents are occurred at the region of Toronto and East York. We can also see that there are few duplicate district that can be merged for example in this case we have Toronto East York that is same as the other one and we have null values as well that we need to take care of.  
A graph of different colored bars

Description automatically generated

We can see that in our target column that is either the accident is fatal or not. We can see from the graph that we have 15.6 thousand non-fatal class and only 2.6k fatal class. Here, we can see that we have imbalanced classes. We need to make it balance before passing data into the model for training so that our model will give accurate predictions not the biased prediction. We have some other minor classes also that we can transform to the actual classes using the most frequent.

A graph of injury and injury

Description automatically generated

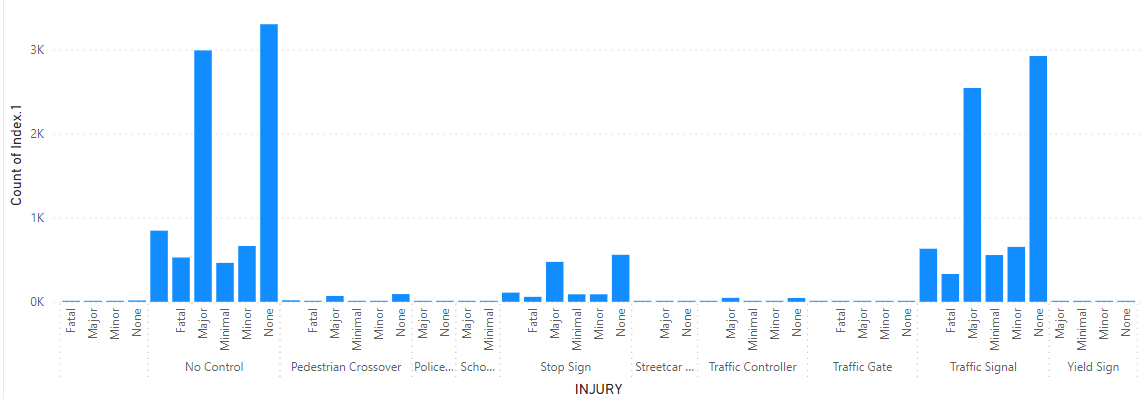
Here, we can see from the time and accidents graphs that the accidents occurred more in the summertime and then it slightly decreased when there is winter because in winter there are less crowd on the roads.  
We can also, analyse that after 2020 the graph decreased significantly of around 4% because at that time COVID came and everyone is in their home and number of accidents got dropped.

## A graph showing a line Description automatically generated

A graph of a line

Description automatically generated with medium confidence

We can analyse that when there are no control and when there is traffic signal then the most accidents occurred.



## Balance or Imbalance Classes

Non-fatal injury 15599  
 fatal injury 2573  
Property Damage 17

The classes are completely imbalanced. We have very less instances of fatal injury as compared to non-fatal. If I am talking about property damage, there are only 17 instances and these instances happened from 2018.  
A screen shot of a computer code

Description automatically generated

## Transformation:

We did the transformation of our columns and have converted values into numeric.

* Have replaced ‘yes’ to 1 and any null values of ‘No’ to 0 for columns ['PEDESTRIAN', 'CYCLIST', 'AUTOMOBILE', 'MOTORCYCLE', 'TRUCK', 'TRSN\_CITY\_VEH', 'EMERG\_VEH', 'PASSENGER', 'SPEEDING', 'AG\_DRIV', 'REDLIGHT', 'ALCOHOL', 'DISABILITY']
* For column ‘DRIVCOND’ we have replaced every value with 0 except the normal. Have created a new column named ‘ABNORMAL’ and has dropped the original column. Now, the new column has value either the driving condition is abnormal or not.
* We have done the same things and have simplified the columns and have converted them into 1 and 0. Columns are [‘DRIVACT’, ‘ACCLASS’, ‘INVTYPE’, ‘ROAD\_CLASS’, ‘TRAFFCTL’, ‘LOCCOORD’]
* We have transposed the time column to Date and Time different columns.
* From the time columns we have transposed the time into classes like Morning, Afternoon, and Night.

## Preprocessing

We have utilized different techniques to consider which columns should we use for the model. For that we have used, co-relation matrix, null values evaluation, and have used feature selection using SelectKBest.

|  |  |
| --- | --- |
| Columns | Selected |
| TRSN\_CITY\_VEH | Yes |
| TRAFFCTL | Yes |
| LATTITUDE | Yes |
| LONGITUDE | Yes |
| MOTORCYCLE | Yes |
| YEAR | Yes |
| AUTOMOBILE | Yes |
| CYCLIST | Yes |
| TRUCK | Yes |
| PEDESTRIAN | Yes |

\*Note: We are taking only 10 features for our model training and testing

We are doing Stratified Shuffling so that X and y train and test data will get equal number of classes of target column.

#### Preprocessing Columns

We will be preprocessing our numerical columns and categorical columns using different techniques.  
Numerical columns: ['X', 'Y', 'INDEX\_', 'ACCNUM', 'YEAR', 'TIME', 'WARDNUM', 'LATITUDE', 'LONGITUDE', 'FATAL\_NO', 'ObjectId']  
Categorical columns: ['DATE', 'STREET1', 'STREET2', 'OFFSET', 'ROAD\_CLASS', 'DISTRICT', 'LOCCOORD', 'ACCLOC', 'TRAFFCTL', 'VISIBILITY', 'LIGHT', 'RDSFCOND', 'ACCLASS', 'IMPACTYPE', 'INVTYPE', 'INVAGE', 'INJURY', 'INITDIR', 'VEHTYPE', 'MANOEUVER', 'DRIVACT', 'DRIVCOND', 'PEDTYPE', 'PEDACT', 'PEDCOND', 'CYCLISTYPE', 'CYCACT', 'CYCCOND', 'PEDESTRIAN', 'CYCLIST', 'AUTOMOBILE', 'MOTORCYCLE', 'TRUCK', 'TRSN\_CITY\_VEH', 'EMERG\_VEH', 'PASSENGER', 'SPEEDING', 'AG\_DRIV', 'REDLIGHT', 'ALCOHOL', 'DISABILITY', 'HOOD\_158', 'NEIGHBOURHOOD\_158', 'HOOD\_140', 'NEIGHBOURHOOD\_140', 'DIVISION']

* We are using Simple Imputer using mean and MinMaxScalar for our numerical columns pipeline.
* For our categorical columns, we are using Simple Imputer using most frequent values and OneHotEncoder.

Generating Artificial instances

After selecting all the features that we will be using for the model training and prediction we are creating artificial instances using SMOTE. This will make our classes balanced.

### Model Building

We are creating 5 models for predicting whether the accidents are fatal or not.

The models we are using are: XGBoost, SVM, Decision Tree, Random Forest, and Logistic Regression. We are using Grid Search CV for getting the fine-tuning models.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | XGBoost | SVM | Decision Tree | Random Forest | Logistic Regression |
| Training Accuracy | 91.70 | 71.88 | 74.27 | 99.86 | 60.11 |
| Testing Accuracy | 87.38 | 71.33 | 70.92 | 97.53 | 59.81 |
| Cross Validate Score |  |  |  |  |  |

Confusion Matrix:

|  |  |
| --- | --- |
| XGBoost | [[2860 263]  [ 196 319]] |
| SVM | [[2391 732]  [ 311 204]] |
| Decision Tree | [[2232 891]  [ 167 348]] |
| Random Forest | [[3081 42]  [ 48 467]] |
| Logistic Regression | [[1874 1249]  [ 213 302]] |

Classification Matrix:

|  |  |
| --- | --- |
| XGBoost | precision recall f1-score support  0 0.94 0.92 0.93 3123  1 0.55 0.62 0.58 515  accuracy 0.87 3638  macro avg 0.74 0.77 0.75 3638  weighted avg 0.88 0.87 0.88 3638 |
| SVM | precision recall f1-score support  0 0.88 0.77 0.82 3123  1 0.22 0.40 0.28 515  accuracy 0.71 3638  macro avg 0.55 0.58 0.55 3638  weighted avg 0.79 0.71 0.74 3638 |
| Decision Tree | precision recall f1-score support  0 0.93 0.71 0.81 3123  1 0.28 0.68 0.40 515  accuracy 0.71 3638  macro avg 0.61 0.70 0.60 3638  weighted avg 0.84 0.71 0.75 3638 |
| Random Forest | precision recall f1-score support  0 0.98 0.99 0.99 3123  1 0.92 0.91 0.91 515  accuracy 0.98 3638  macro avg 0.95 0.95 0.95 3638  weighted avg 0.98 0.98 0.98 3638 |
| Logistic Regression | precision recall f1-score support  0 0.90 0.60 0.72 3123  1 0.19 0.59 0.29 515  accuracy 0.60 3638  macro avg 0.55 0.59 0.51 3638  weighted avg 0.80 0.60 0.66 3638 |

Based on the provided information, the Random Forest model seems to be the most suitable choice among the models evaluated. Here's why:

1. **High Test Accuracy:** The Random Forest model achieved the highest test accuracy of 0.9753 among the evaluated models, indicating strong predictive power on unseen data.
2. **Balanced Precision and Recall:** The classification report shows a good balance between precision and recall for both classes (0 and 1), indicating that the model performs well for both the majority and minority classes.
3. **Low Overfitting:** While the model achieved a very high training accuracy of 0.9986, the test accuracy is still quite close, suggesting that the model generalizes well to new data and is less likely to overfit. The model also exhibits a high cross-validation mean score (0.9546), which suggests consistent performance across different folds of the dataset. Additionally, Random Forests are known for their ability to handle complex relationships in data, reduce overfitting, and work well even with unbalanced datasets. Given the significant difference in performance metrics, Random Forest seems to be the most suitable choice for this classification task.
4. **Interpretability:** Random Forest models can provide feature importance scores, which can be useful for understanding the factors that contribute the most to the model's predictions.
5. **Robustness:** Random Forest models are known for their robustness against noisy and irrelevant features in the dataset, making them suitable for various types of data.

Overall, the Random Forest model combines strong performance, robustness, and interpretability, making it a solid choice for this classification problem.