Car Accident Severity Prediction Model For Seattle City

Jai Vigneshwar Aiyyappan Just now · 9 min read

Coursera Capstone Project

Intro and Business Understanding:

The seaport city of Seattle is the largest city in the state of Washington, as well as the largest in the Pacific Northwest. As of the latest census, there were 713,700 people living in Seattle. Seattle residents get around by car, trolley, streetcar, public bus, bicycle, on foot, and by rail. With such bustling streets, it's no surprise that Seattle sees car accidents every day.

In 2015, a crash occurred in Washington every 4.5 minutes. Seattle recorded the highest number of car accidents in the state that year, at 14,508 (in second place was Tacoma with just 4,756). Although the city is taking steps to make the roadways safer for citizens, vehicle collisions are still a serious danger. While car accident injuries can vary from person to person and from crash to crash. However, if we can build a model to predict the severity of a car accident, that would avoid a lot of unnecessary accidents and injuries, even deaths. Therefore, the Seattle government is planning to use this model that alert drivers, health systems, and police to remind them to pay more attention to critical situations.

Imagine a scenario that if there is a rainy day and you are planning to drive to another city. The model can predict the severity of the accident severity based on the condition of traffic, report the accident location, weather conditions, and other factors that provide you more options for traveling, reschedule, or drive more carefully. However, distraction and not paying enough attention while driving are important reasons that cause car accidents. This model will alert drivers driving more carefully and can be prevented by enacting harsher regulations.

The target audience of this project is the Seattle government, police, rescue groups, and drivers. This model and its prediction will provide advice for decision making and prevent unnecessary accidents and injuries for the city of Seattle.

Data understanding:

The dataset used for this project is based on car accidents that have occurred in the city of Seattle, Washington from 2004 to 2020. The dataset contains 37 independent variables and 194,673 rows. The independent variables include “INATTENTIONIND”, “WEATHER”, “ROADCOND” and other factors that would cause the accident. The dependent variable is “SEVERITYCODE” which contains numbers of “1” and “2”, they correspond to different levels of severity of car accidents. “1” represents for “Property Damage Only” and “2” means “Physical Injury”.

In my consideration, I will drop some non-critical and indecisive attributes. The following features which I choose to remain for building model and prediction.

Data Pre- Processing:

An unbalanced datset is used, provided by the Seattle Department of Transportation Traffic Management Division with 194673 rows (accidents) and 37 columns (features) where each accident is given a severity code. The steps taken in pre-processing the dataset are as follows.

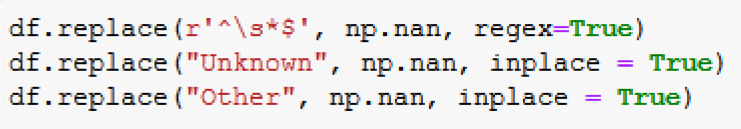
1. Removal of irrelevant columns or features

Columns containing descriptions and identification numbers that would not help in the classification are dropped from the data set to reduce the complexity and dimensionality of the data set. ‘OBJECTID’, ‘INCKEY’, ‘COLDETKEY’, ‘REPORTNO’, ‘STATUS’, ‘INTKEY’, ‘EXCEPTRSNCODE’ and more belong to this category. Certain other categorical features were removed as they had a large number of distinct values, example: ‘LOCATION’.

After performing this step, the dimensionality dropped from 37 to 11.

2. Identification and handling missing values

To identify columns and rows with missing values is the next step. Empty boxes, ‘Unknown’ and ‘Other’ were values considered as missing values. These were replaced with NA to make the dataset uniform.

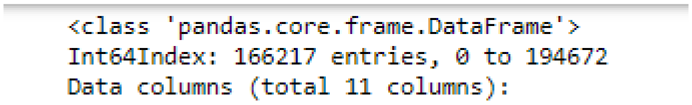


```
df.replace(r'^\s*$', np.nan, regex=True)
df.replace("Unknown", np.nan, inplace = True)
df.replace("Other", np.nan, inplace = True)
```

Replacing Missing Values with NA

For columns ("X”, ”Y”, ”UNDERINFL”, ”WEATHER”,”ROADCOND”, ”LIGHTCOND) which had less than 20% of its values missing, the respective rows were removed since most of the columns in this dataset are categorical type, goal was to not impute the non-numerical columns; hence it did not make sense to replace the values.

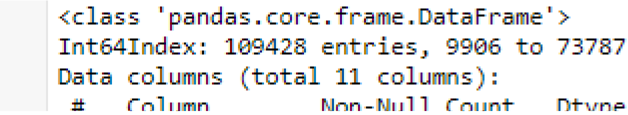
Once the above strategies were performed, the dataset reduced from having 194673 rows and 11 columns to having 166217 rows and 11 columns.



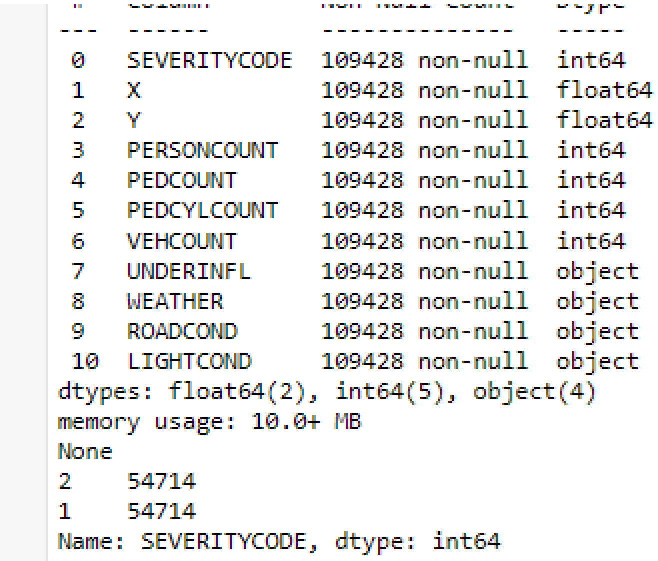
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 166217 entries, 0 to 194672
Data columns (total 11 columns):
```

3. Balancing the dataset

With the above two pre-processing steps complete, a dataset (166217rows) with 111503 rows for severity code 1 and 54714rows for severity code 2 is obtained. Training an algorithm on an unbalanced dataset w.r.t the target category will result in a biased model. The model will have learnt more about one the category that has more data. In order to prevent this, a new balanced dataset (111503 rows) is created by randomly sampling out 54714 rows with severity code 2 and then concatenating it with 54714 rows with severity code 1. The dataset is then shuffled to randomize the rows.



```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 109428 entries, 9906 to 73787
Data columns (total 11 columns):
 #   Column         Non-Null Count   Dtype
```



```
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   SEVERITYCODE   109428 non-null  int64
 1   X              109428 non-null  float64
 2   Y              109428 non-null  float64
 3   PERSONCOUNT    109428 non-null  int64
 4   PEDCOUNT       109428 non-null  int64
 5   PEDCYLCOUNT    109428 non-null  int64
 6   VEHCOUNT       109428 non-null  int64
 7   UNDERINFL      109428 non-null  object
 8   WEATHER        109428 non-null  object
 9   ROADCOND       109428 non-null  object
 10  LIGHTCOND      109428 non-null  object
dtypes: float64(2), int64(5), object(4)
memory usage: 10.0+ MB
None
2    54714
1    54714
Name: SEVERITYCODE, dtype: int64
```

Balanced Dataset Info

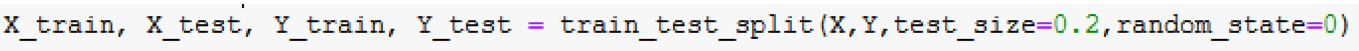
4. Encoding of data

The dataset is split into two datasets, X and Y, where Y contains the target feature (SEVERITYCODE) and X contains all the independent features/variables.

Machine Learning models are trained only on numerical data; hence all categorical features in the dataset have to be encoded so that the algorithms can be trained on those features. The 'get_dummies' method from pandas library is used to convert/encode each and every categorical feature. After application, number of features in dataset X increased from 11 to 31.

5. Splitting into training and testing datasets

The datasets X and Y are split into X_train, Y_train, X_test,and Y_test. The first two will be used for training purposes and the last two will be used for testing purposes. The split ratio is 0.8, 80% of data is used for training and 20% of is used for testing.



```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y,test_size=0.2,random_state=0)
```

Splitting into train and test sets

6. Normalizing/ Feature scaling of data

Feature scaling of data is done to normalize the data in a dataset to a specific range. It also helps improve the performance of the ML algorithms. Standard Scaler metric is used to scale/normalize all the numerical data for both, the X_train and X_test datasets. This completes the pre-processing stage, we can move on to training our models.

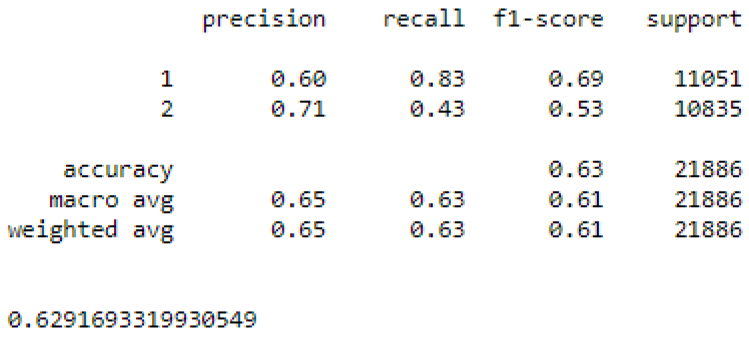
Modeling:

As for building the model for accident severity prediction in this project, the cleaning and re-sampling data were split into testing and training sets, 20% of the total data samples for testing, and the rest 80% data we used for training the model. Four machine learning models have been built and evaluated.

1)Logistic Regression Classifier

Logistic Regression is a classifier that estimates discrete values (binary values like 0/1, yes/no, true/false) based on a given set of an independent variables. It basically predicts the probability of occurrence of an event by fitting data to a logistic function**.** Hence it is also known as logistic regression. The values obtained would always lie within 0 and 1 since it predicts the probability.

The chosen dataset has only two target categories in terms of the accident severity code assigned; hence it was possible to apply this model to the same. The results, confusion matrix, classification report and accuracy, are:



```
              precision    recall  f1-score   support

           1       0.60      0.83      0.69     11051
           2       0.71      0.43      0.53     10835

    accuracy                           0.63     21886
   macro avg       0.65      0.63      0.61     21886
weighted avg       0.65      0.63      0.61     21886


0.6291693319930549
```

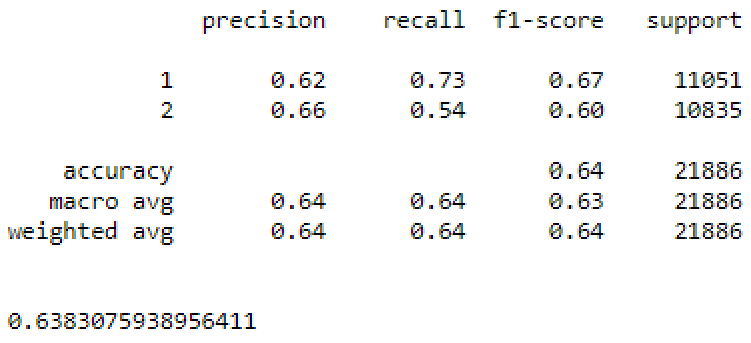
LR results

2)K Nearest Neighbours Classifier

K nearest neighbours algorithm used for both classification and regression problems. It basically stores all available cases to classify the new cases by a majority vote of its k neighbours. The case assigned to the class is most common amongst its K nearest neighbours measured by a distance function (Euclidean, Manhattan, Minkowski, and Hamming).

In order to arrive at the optimum values for nearest neighbours (k) and the distance metric (Euclidean and Manhattan), a hyper parameter KNN was used. The best accuracy was obtained for 59 nearest neighbours with Euclidean being the distance metric when applied for the problem in question.

The results, confusion matrix, classification report and accuracy, are:



```
              precision    recall  f1-score   support

           1       0.62      0.73      0.67     11051
           2       0.66      0.54      0.60     10835

    accuracy                           0.64     21886
   macro avg       0.64      0.64      0.63     21886
weighted avg       0.64      0.64      0.64     21886


0.6383075938956411
```

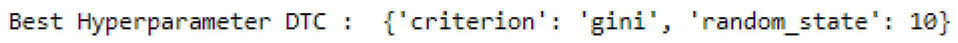
KNN Results

3)Decision Tree Classifier

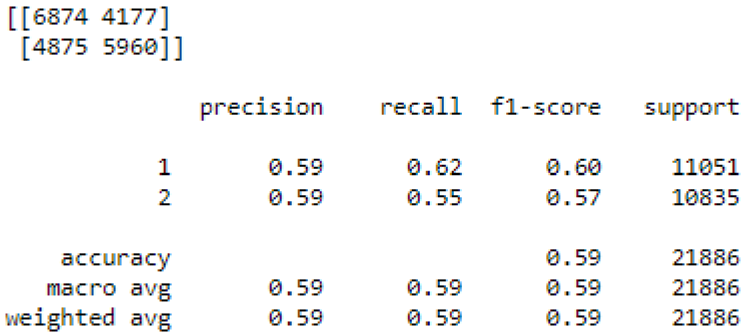
Decision Tree makes decision with tree-like model. It splits the sample into two or more homogenous sets (leaves) based on the most significant differentiators in the input variables. To choose a differentiator (predictor), the algorithm considers all features and does a binary split on them (for categorical data, split by category; for continuous, pick a cut-off threshold). It will then choose the one with the least cost (i.e. highest accuracy), and repeats recursively, until it successfully splits the data in all leaves (or reaches the maximum depth).

Information gain for a decision tree classifier can be calculated either using the Gini Index measure or the Entropy measure, whichever gives a greater gain. A hyper parameter Decision Tree Classifier was used to decide which tree to use, DTC using entropy had greater information gain; hence it was used for this classification problem.

The results, confusion matrix, classification report and accuracy, are:



```
Best Hyperparameter DTC :  {'criterion': 'gini', 'random_state': 10}
```



```
[[6874 4177]
 [4875 5960]]
              precision    recall  f1-score   support

           1       0.59      0.62      0.60     11051
           2       0.59      0.55      0.57     10835

    accuracy                           0.59     21886
   macro avg       0.59      0.59      0.59     21886
weighted avg       0.59      0.59      0.59     21886
```

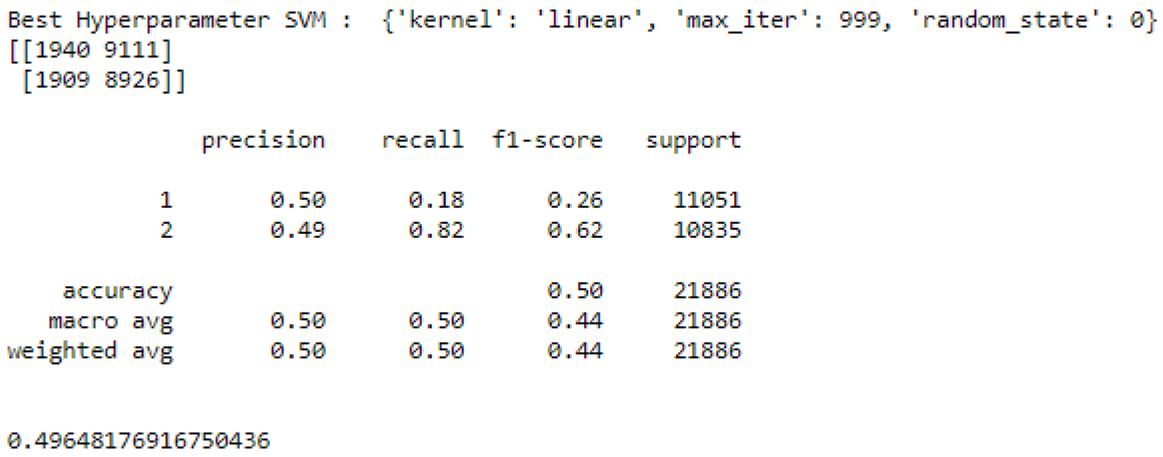
DTC Results

4)Support Vector Machine Classifier

Support Vector Machine is an algorithm which can be used for both classification and regression challenges. However, it is mostly used in classification problems. In the SVM algorithm, each data item is plotted as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, classification is performed by finding the hyper-plane that differentiates the two classes.

Hyper parameter SVC was used to choose between Linear SVC and a Kernel SVC and the latter arrived on top with a greater accuracy when applied on the dataset in question. It used the 'radial basis function' kernel for performing the classification.

The results, confusion matrix, classification report and accuracy, are:



```
Best Hyperparameter SVM :  {'kernel': 'linear', 'max_iter': 999, 'random_state': 0}
[[1940 9111]
 [1909 8926]]
              precision    recall  f1-score   support

           1       0.50      0.18      0.26     11051
           2       0.49      0.82      0.62     10835

    accuracy                           0.50     21886
   macro avg       0.50      0.50      0.44     21886
weighted avg       0.50      0.50      0.44     21886


0.49648176916750436
```

SVC Results

Evaluation:

The models are evaluated based on several factors:

· **Precision:** the ratio of correctly predicted positive observations to the total predicted positive observations.

· **Recall (Sensitivity):** the ratio of correctly predicted positive observations to all observations in the actual class

· **F1 Score:** the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have a similar cost. If the cost of false positives and false negatives are very different, it’s better to look at both Precision and Recall.

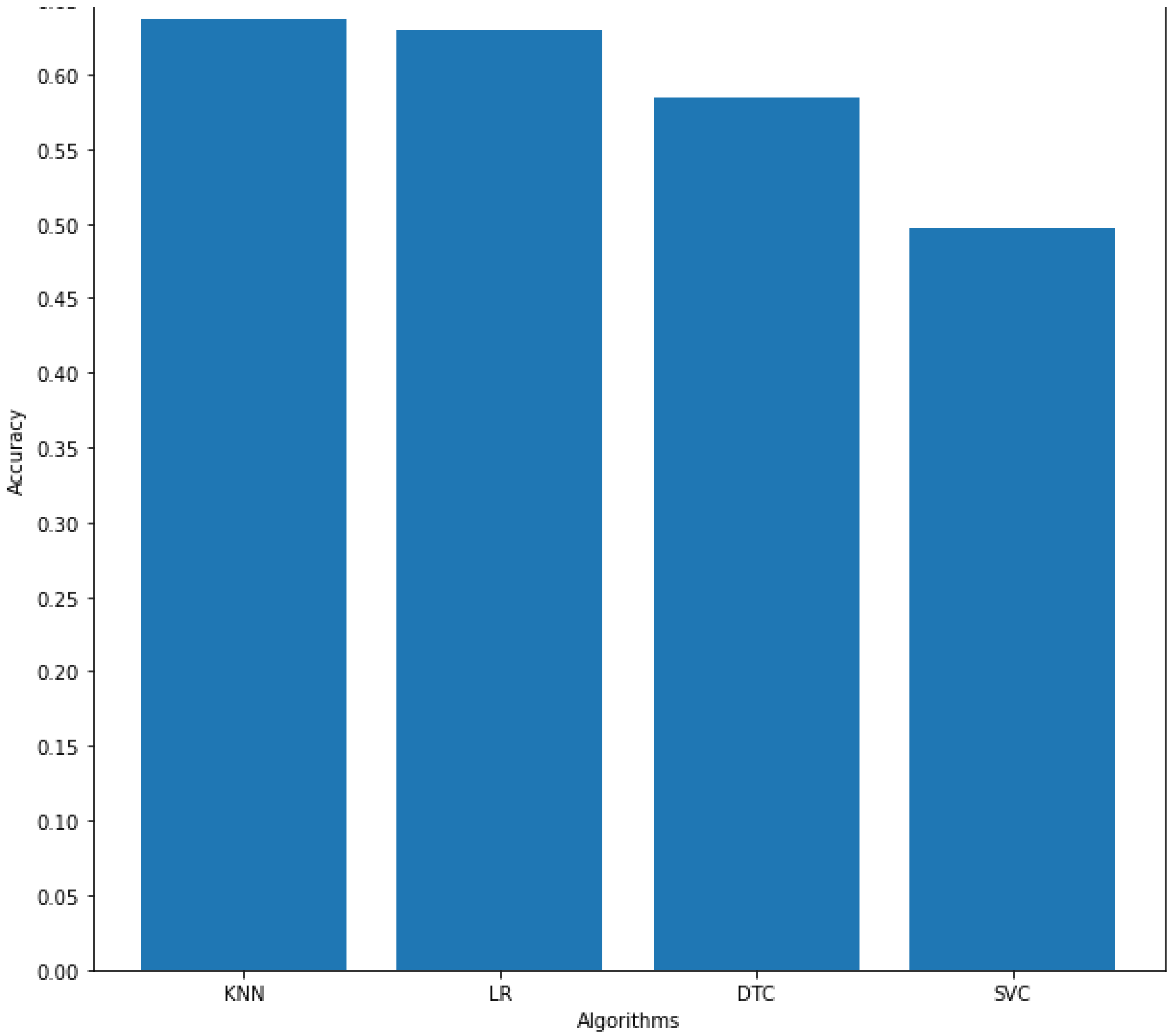
Discussion

From the analysis of the result of 4 machine learning algorithm models on the city of Seattle collision dataset, it can be seen that four models are performed very similarly, but KNN and Decision Tree methods are both 4% or more higher than the rest of the others in F1 Score . In Support Vector Machine, the F1 score is 0.50 which is a relatively low value indicates a high uncertainty/entropy of my model.

Based on the dataset from knowing the driver is under influenced by alcohol/drug or not, weather, road, and light conditions, we build different models and it can be concluded that particular conditions have a somewhat impact on the severity code which results in property damage (class 1) or injury (class 2).

Results

None of the algorithms implemented above gave an accuracy score equal to or greater than 0.7, they all ranged from 0.5 to 0.65. Meaning, these models can predict the severity code of an accident with an accuracy equaling 50–65%. A bar plot is plotted below with the bars representing the accuracy of each model in descending order respectively.



Accuracy Comparison

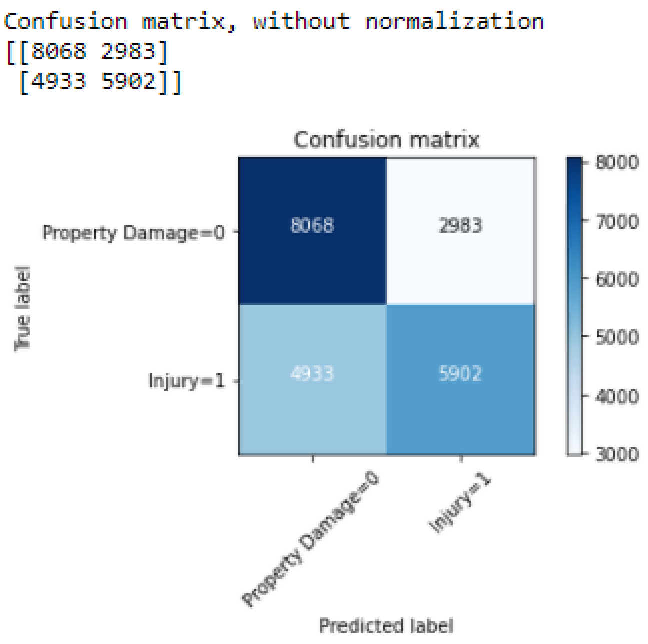
Conclusion

The accuracy of the classifiers is not great, highest being 64%. This usually means that the model is under fitted i.e. it needs to be trained on more data. Though the dataset has a lot of variety in terms of scenarios, more volume of the data for such scenarios has to be collected.

Certain features with missing values were removed, this reduced the dimensionality of the dataset, these features could have been correlated to other important features but they had to be removed. A better effort has to be made to collect data to reduce the number of missing values.

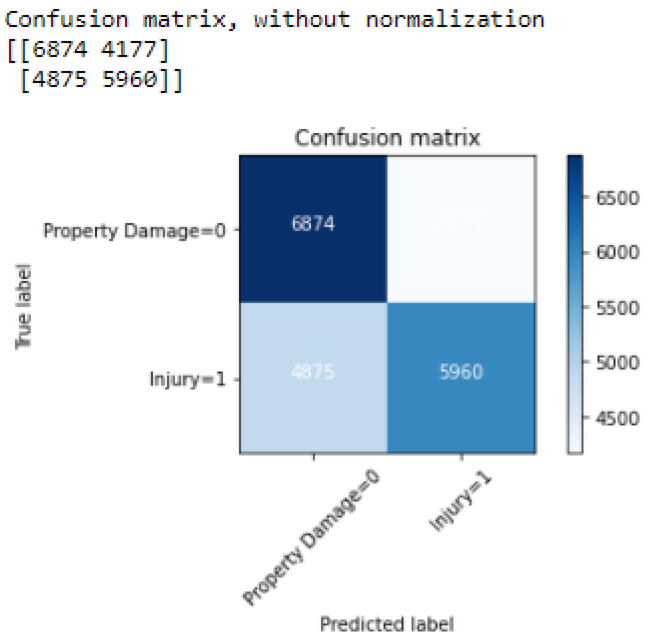
Appendix

· KNN Confusion Matrix



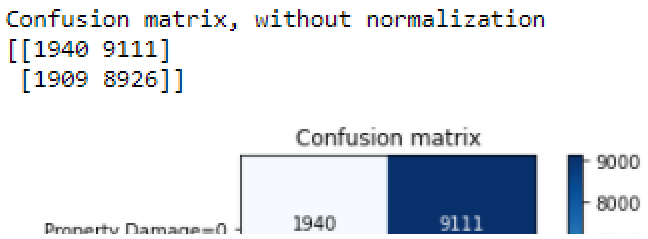
```
Confusion matrix, without normalization
[[8068 2983]
 [4933 5902]]
```

· Decision Tree Confusion Matrix

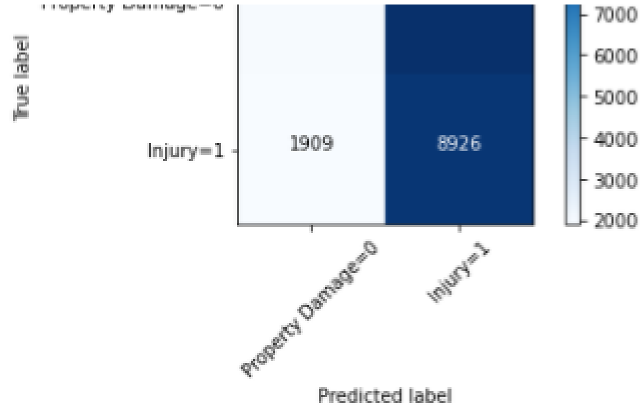


```
Confusion matrix, without normalization
[[6874 4177]
 [4875 5960]]
```

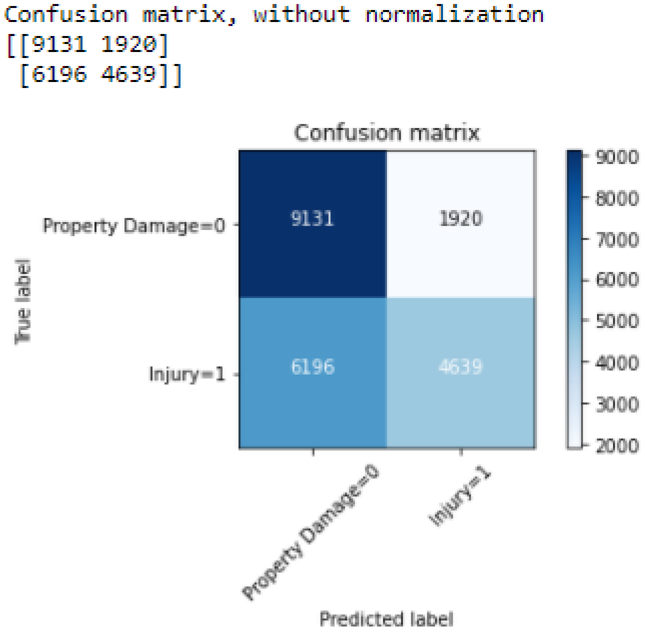
· SVM Confusion Matrix



```
Confusion matrix, without normalization
[[1940 9111]
 [1909 8926]]
```



· Logistic Regression Confusion Matrix



```
Confusion matrix, without normalization
[[9131 1920]
 [6196 4639]]
```

Coursera Capstone Applied Data Science Coursera Knn Algorithm Decision Tree