Prediction of Traffic Accidents Severity Based on Machine Learning and Multiclass Classification Model

Mateja Iveta*, Aleksander Radovan**, Branko Mihaljević#
*Faculty of Electrical Engineering and Computing/Zagreb, Croatia
mateja.iveta@outlook.com
**BISS/Zagreb, Croatia
aleksander.radovan@biss.hr
#Rochester Institute of Technology Croatia/Zagreb, Croatia
branio.mihaljevic@croatia.rit.edu

2021 44th International Convention on Information, Communication and Electronic Technology (MIPRO) | 978-953-233-101-1/21/$31.00 ©2021 10.23919/MIPRO52101.2021.9596664

***Abstract* - Road traffic accidents are a common and seemingly inevitable problem. While its occurrences rely on many unpredictable factors, this paper shows how to utilize machine learning to predict both the possibility of the accident and its severity.**

The datasets used were related to road accidents in several countries in a period of a few years. Some of the parameters observed were the weather conditions, sun position, speed limit, and time of the day. To predict the severity of the accident given the circumstances and road conditions, a multiclass classification model is used. Different datasets were combined to cover different situations and scenarios that happen in traffic and taking the severity of accidents in prediction. The dataset values were normalized before the training process and the training set and validated on the validation dataset.

The prediction results show the correlation between used weather conditions, daylight time, and traffic accident severity.

Keywords - multiclass classification; deep learning; road accidents

I. INTRODUCTION

The number of road vehicles in operation has been on a steady increase. With more and more people participating in traffic, accidents are a common occurrence, and the number of fatalities is high. [1] Since they are dependent on many different outside influences, and ultimately the reflexes and reactions of the individuals involved, they are seemingly impossible to predict.

To determine the severity of the accident, we need to consider the most prominent consequences. A casualty is a person killed, slightly injured, or seriously injured. If a person has sustained injuries that have caused death within 30 days after the accident, their death is attributed to the accident. Generally, severity is classified depending on the most serious injuries and, sometimes, the material damage.

Knowledge about the seriousness of the accident could benefit police and hospitals. For example, it might help with better allocation of staff and resources for a quicker reaction. In this paper, the goal is to use machine learning and the data about the circumstances of the accident for severity prediction. The data used was the Road accidents and safety statistics provided by the United Kingdom government.

The existing research on this or a similar topic used many different facts surrounding the accidents. [2] took on a time-series approach, using the information about traffic volume, speed, and occupancy collected in short intervals. More specifically, they gathered the information from before and after the accident. With oversampling and a recurrent neural network, they achieved an accuracy of 96% and a detection rate of 75%. [3] used decision trees and oversampling as well. They used the available information like the traffic volume, gender of the driver, alcohol consumption, and some features also used in this research weather and light conditions. The achieved accuracy was up to 75.5%.

The next chapter provides a more detailed description of the dataset as well as the preprocessing techniques. Chapter three describes the methods applied for handling the imbalance of the dataset, evaluation metrics, and principal component analysis. In the fourth chapter, we show both the shallow and deep learning solutions. The fifth chapter shows the anomaly detection approach, both supervised and unsupervised. The last chapter discusses the achieved results and possible future improvements.

II. DATA

For this research, we chose the dataset provided by the UK government. [4] The data from January 2015 to December 2018 was used as the historical data to train the models. The remaining examples from the year 2019 were left to test the model performance.

Some of the provided information was not used. The first category is the features that do not reflect the circumstances of the accident, like the jurisdiction information or whether the police attended the scene. The second is features with imbalanced binary labels that were likely to affect model performance. After removing rows with missing data, 529102 training examples remained.

Sponsor: BISS d.o.o., Zagreb, Croatia

A. Data preprocessing

A relevant factor is the time when the accident occurred. Time representation in the original dataset is with the following variables Date, Time, and Day_of_Week. While they are technically discrete, they take on enough different values to be considered continuous. Another important property is that they are cyclical - January 1 is just one day away from December 31 - and in the original data, they are 364 places apart. To combat this issue, for each of these cyclical features, we calculated sine and cosine, using them instead of the initial value. Figure 1 displays the cyclical nature of the day of the year feature on 100 examples.

Some of the given features denoted nominal categorical data like Light_conditions, Weather_conditions, etc. Selected encoding was dummy encoding instead of one-hot encoding to avoid the problem of multicollinearity.

After those transformations, 32 features remained, and scaling was the last step of our data preprocessing. The Standardization technique was used to centre the values around zero.

B. Data analysis

Looking at the covariance matrix of the dataset in Figure 4, seemingly the most connected variable to the accident severity is the speed limit.

The most prominent problem with both training and test dataset is the skew of output labels. Some labels were underrepresented, with the number of examples several orders of magnitude lower for the 1st class than the 3rd which can be seen in Figure 2.

As the accident severity is an ordinal categorical variable transformation was not an option so the methods of handling the data imbalance using oversampling and undersampling are described in the following chapter.

III. METHODS

A. Handling data imbalance

Since the dataset was severely skewed, the most common class more than 65 more often than the least common class, we tried to balance the dataset using different sampling techniques and compared their performance.

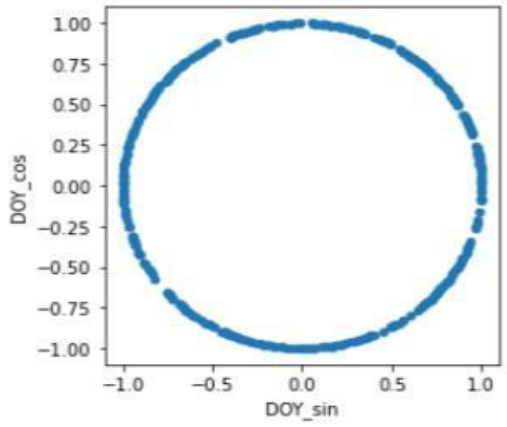

Figure 1. Sine and cosine values of the day of the year feature. Displays the cyclical nature of the variable.

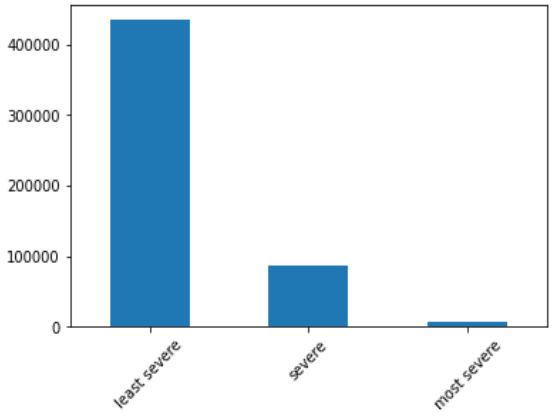

Figure 2 . Number of examples for each output class in the training dataset.

1) SMOTE

Synthetic Minority Oversampling Technique (SMOTE) is an oversampling method that synthesizes new examples. [13] The implementation used is provided by the imbalanced-learn library. [5] The cost of a severe accident being a false negative is larger than the misclassification of an accident with no causalities, so we opted for this method to increase the number of the underrepresented class.

2) NearMiss

On the other hand, to decrease the number of examples of the least severe accidents, we used the NearMiss undersampling technique. Specifically, the NearMiss-1 version, also from the imbalanced-learn library, that selects the majority class examples with the minimum average distance to three closest minority class examples.

3) Weighted sampling

The final sampling method used was the custom weighted sampling. A small, random, and equal number of examples (6655) of each class was selected from the training dataset. This data was fed into a simple feedforward neural network with two hidden layers. After training and optimizing this network, the F1 score, further explained in the following chapter, was calculated. Its values were 0.40, 0.30, and 0.59 for each of the labels, respectively. To take into consideration class frequency and its distinctiveness, we calculated the sampling weights with the following formula:

$$samplingWeight[class] = (1 - F1[class]) * frequency[class] \quad (1)$$

Those sampling weights served as the probability that an example will be sampled. With 300,000 examples selected using this approach, the distribution was still skewed but less so which is shown in Figure 3.

B. Principal Component Analysis

Principal Component Analysis (PCA) is a dimensionality reduction method, first proposed in [6], that creates new uncorrelated variables using the linear combinations of existing features. The goal is to minimize the number of features while retaining the maximal data variance. Since the number of features was 32, and the training dataset contained over half a million examples, this

	Longitude	Latitude	Number_of_Vehicles	Number_of_Casualties	Day_of_Week	Speed_limit	Accident_Severity
Longitude	1.978102	-0.885483	0.015210	-0.034845	0.007324	-1.695467	0.020412
Latitude	-0.885483	1.974886	-0.022462	0.026412	0.006196	1.475634	-0.020686
Number_of_Vehicles	0.015210	-0.022462	0.525233	0.134215	-0.001386	1.111531	0.021859
Number_of_Casualties	-0.034845	0.026412	0.134215	0.608578	-0.002448	1.830273	-0.023905
Day_of_Week	0.007324	0.006196	-0.001386	-0.002448	3.715582	-0.488648	0.003947
Speed_limit	-1.695467	1.475634	1.111531	1.830273	-0.488648	203.661153	-0.521754
Accident_Severity	0.020412	-0.020686	0.021859	-0.023905	0.003947	-0.521754	0.187327

Figure 2. Covariance matrix of dataset with selected features.

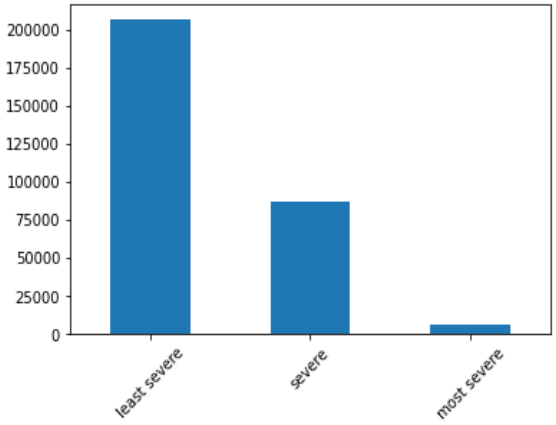

Figure 3. Number of examples for each output class in the training dataset after weighted sampling.

would not be considered a high dimensional problem, but PCA can help the models with better generalization. We used the implementation defined in [7] provided by the scikit-learn library. [8] Table 1 shows how the number of PCA features changes as the amount of explained variance in data changes.

TABLE I. NUMBER OF PCA FEATURES PER PERCENTAGE OF VARIANCE RETAINED

Variance retained [%]	***Number of PCA features***
100	32
95	27
90	24
80	20

C. Evaluation

Although it is one of the commonly used metrics, accuracy can be misleading, especially with skewed datasets. Furthermore, since the least represented class in our dataset is also the least desirable to misclassify, we opted for an additional metric to best evaluate each model.

A confusion matrix shows how the classification model is confused while making predictions. Each row of the matrix corresponds to the true class and each column to the predicted class. True positives (TP) are correctly classified values, and they are on the diagonal of the confusion matrix. False negatives (FN) are row-wise sums – without the diagonal – and they represent the number of examples of a class that have not been sorted in it. False positives (FP) are column-wise sums – without the diagonal – that represent the number of examples that are not members of a certain class but were classified as such.

Precision, recall, and subsequently the F1 score are then calculated from the confusion matrix using the following formulas (2), (3), and (4) respectively.

$$precision = \frac{TP}{TP + FP} \tag{2}$$

$$recall = \frac{TP}{(TP + FN)} \tag{3}$$

$$F1 = 2 * \frac{precision * recall}{precision + recall} \tag{4}$$

IV. EXPERIMENTS AND RESULTS

A. Random Forest classifier

The Random Forest classification ensemble method has shown to be effective on a similar dataset in [3] and is less prone to overfitting than decision trees so it was used as the baseline model. The other way to prevent overfitting decision trees is pruning which was done to limit the depth of each tree to maximally 20. This number was chosen because it is the smallest number of features we got after applying PCA (corresponding to 80% of variance retained as can be seen in Table 1).

TABLE II. PERFORMANCE OF THE RANDOM FOREST CLASSIFIER WITH DIFFERENT COMBINATIONS OF SAMPLING AND PCA

	F1 score per label			
Model	***1***	***2***	***3***	***Overall accuracy***
SMOTE + PCA (90)	0.05	0.16	0.71	0.53
NearMiss + PCA (90)	0.05	0.18	0.03	0.06
Weighted sampling + PCA (100)	0.07	0.10	0.81	0.68
No sampling + PCA (80)	0.007	0.003	0.90	0.81

The importance of sampling can be seen in both Table 2 and Table 3. Since with no sampling there were so many more examples of the class labeled with '3' (the least severe accidents), the network did not learn to generalize. Since the accuracy is not a reliable metric in this case, a model that almost always produces just one label reached the highest accuracy.

Weighted sampling method also gave a high, which can also be explained with the skewed data. But, looking at the per class F1 scores, its performance is

similar to SMOTE + PCA (90) which had an equal number of examples for each class.

The NearMiss undersampling method performed poorly, and the change in number of features did not significantly affect the results.

B. Deep learning

A deep learning model used was a feedforward neural network built and trained using the Keras API. [9] The optimization of hyperparameters was done using a grid search and the final, best-performing configuration had the learning rate of 0.02, trained for 35 epochs and had a dropout layer between all layers with the probability set to 0.1. The model was optimized using the Adam optimizers and categorical cross-entropy was chosen as the loss function. It has three hidden layers with 32, 128, and 512 hidden units respectively, and the outermost layers have the ReLu activation function, and the middle the Tanh activation.

TABLE III. PERFORMANCE OF THE DEEP LEARNING CLASSIFIER WITH DIFFERENT COMBINATIONS OF SAMPLING AND PCA

	F1 score per label			
Model	***1***	***2***	***3***	***Overall accuracy***
SMOTE + PCA (80)	0.06	0.24	0.61	0.45
SMOTE + PCA (95)	0.06	0.25	0.57	0.42
NearMiss + PCA (80)	0.13	0.23	0.03	0.13
Weighted sampling + PCA (100)	0.01	0.03	0.82	0.69
No sampling + PCA (80)	0.00	0.00	0.90	0.82

As can be seen from the data in Table 3, even though the model with weighted sampling has the highest overall accuracy, it failed to correctly classify most of the more severe examples, and with the same data, it was less successful than the corresponding baseline. SMOTE in combination with PCA that provided 20 features performed the best. It still did not outperform the baseline in terms of accuracy, but the F1 scores over the labels are more balanced. The NearMiss method once again did not provide good results, which can possibly be due to an insufficient number of examples for the network to train on.

V. ANOMALY DETECTION

Since the most severe accidents are also the most crucial to label correctly, and all the previous models showed to be unsuccessful in this task, anomaly detection methods were used. So, the examples of the underrepresented class (the most severe accidents) were treated as an anomaly.

A. K-means clustering

A k-means clustering algorithm is an unsupervised shallow learning approach that groups the data into the specified number of clusters. [10] Considering the existing knowledge about the number of classes, we tested the algorithm only for k = 2 and k = 3. Former for potentially separating one of the classes from the others, and latter for the original classification.

TABLE IV. NUMBER OF EXAMPLES PER CLUSTER

Number of clusters	**Sorted numbers of examples per cluster**		
	1	***2***	***3***
K = 2	81434	218566	
K = 3	70924	76550	152526
Original data	3364	65883	230753

a. In this table the label number does not represent the value of the 'Accident_Severity' variable, but merely the index of the cluster.

From the values in Table 4 we can see that the algorithm does not recognize the smallest class as an outlier because there is no group with even remotely as few examples as the original number.

B. Supervised anomaly detection

The last approach was a supervised version of anomaly detection. After mild oversampling (tripling the number of examples) and undersampling (reducing the non-anomalies by the factor of five), we fit a binary classification model.

Similarly, to k-means clustering, it was difficult for the model to differentiate between the classes which resulted in an overall model accuracy of 24.56%. Even after introducing weighted penalization [11], [12] – penalizing the model more for mislabeling certain classes, with the weights being relative frequencies the model, performance improved, but remained at the test set accuracy of 47.32%.

VI. CONCLUSION

From our analysis so far, we have reached a few possible conclusions. Firstly, it is possible that, while the connection between the circumstances surrounding the accident and its severity exist, they might be too random or the human factor may be too influential, to predict the consequences more accurately.

Secondly, as the dataset is extremely skewed, the model might not have been able to correctly grasp the differences between the least and most severe classes. Poor anomaly detection results could imply that the underrepresented label is not easily detectable. Also, the covariance matrix in data analysis shows that the accident severity is not really correlated to the majority of features which might play a part in the quality of fit. A possible improvement may be reached with using different data, similar to features used in [3], like alcohol consumption, seatbelt, and the year when the car was manufactured.

Future research could also make use of the spatiotemporal features to create sequence models like it was done for the Chicago metropolitan area in [2]. A time-series approach might capture the previous road conditions which ultimately lead to an accident.

ACKNOWLEDGEMENT

This research was supported by BISS d.o.o., Zagreb, Croatia.

REFERENCES

[1] European road safety statistics, https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Road_safety_statistics_-

_characteristics_at_national_and_regional_level&oldid=463733#General_overview, last accessed: Februrary 8th, 2021

[2] A.B. Parsa, R. Singh Chauhan, H .Taghipour, S. Derrible,A. Mohammadian "Applying Deep Learning to Detect Traffic Accidents in Real Time Using Spatiotemporal Sequential Data"

[3] R. E. AlMamlook, K. M. Kwayu, M. R. Alkasisbeh and A. A. Frefer, "Comparison of Machine Learning Algorithms for Predicting Traffic Accident Severity," 2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT), 2019, pp. 272-276, doi: 10.1109/JEEIT.2019.8717393.

[4] Road traffic accidents dataset, https://data.gov.uk/dataset/cb7ae6f0-4be6-4935-9277-47e5ce24a11f/road-safety-data, last accessed: February 8th, 2021

[5] "imbalanced-learn," *PyPI*. [Online]. Available: https://pypi.org/project/imbalanced-learn/. [Accessed: 29-Jul-2021].

[6] K. Pearson, "LIII. On lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

[7] M. Tipping "Mixtures of Probabilistic Principal Component Analyzers," *Graphical Models*, 2001.

[8] "scikit-learn," *PyPI*. [Online]. Available: https://pypi.org/project/scikit-learn/. [Accessed: 29-Jul-2021].

[9] K. Team, "Simple. Flexible. Powerful.," *Keras*. [Online]. Available: https://keras.io/. [Accessed: 29-Jul-2021].

[10] A. Likas, N. Vlassis, J.J. Verbeek, "The global k-means clustering algorithm" in Pattern Recognition, Volume 36, Issue 2, 2003, Pages 451-461, ISSN 0031-3203

[11] Y. Ho and S. Wookey, "The Real-World-Weight Cross-Entropy Loss Function: Modeling the Costs of Mislabeling," in *IEEE Access*, vol. 8, pp. 4806-4813, 2020, doi: 10.1109/ACCESS.2019.2962617.

[12] S. Lu, F. Gao, C. Piao and Y. Ma, "Dynamic Weighted Cross Entropy for Semantic Segmentation with Extremely Imbalanced Data," 2019 International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM), Dublin, Ireland, 2019, pp. 230-233,

[13] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.