Highlights

Modeling the Need for an Ambulance based on Automated Crash Reports from iPhones

J. Bradford Burkman,Miao Jin,Malek Abuhijleh,Xiaoduan Sun

- Supports transferability and benchmarking of different approaches on a public large-scale dataset. We have attached the code we used to perform the analysis on the Crash Report Sampling System.
- Novel Application motivated by Emerging Technology: Machine Learning Classification Models for Dispatching Ambulances based on Automated Crash Reports
- New Use of Dataset: Used Crash Report Sampling System (CRSS), which has imputed missing values for some features, but not all of the ones we wanted to use. For the first time we have seen, we used the software the CRSS authors use for multiple imputation (IVEware) to impute missing values in more features.
- Perennial Machine Learning Challenge: Imbalanced Datasets.

Modeling the Need for an Ambulance based on Automated Crash Reports from iPhones

J. Bradford Burkman[a,b], Miao Jin[a], Malek Abuhijleh[a,c] and Xiaoduan Sun[c]

[a]School of Computing and Informatics, University of Louisiana at Lafayette, USA
[b]Louisiana School for Math, Science, and the Arts, USA
[c]Department of Civil Engineering, University of Louisiana at Lafayette, USA

ARTICLE INFO

Keywords:
Automated crash notification
Ambulance dispatch
Emergency medical services
Machine learning

ABSTRACT

Put abstract here.

1. Introduction

1.1. Motivation

A Google Pixel phone can detect the deceleration profile of a car crash and, if you have enabled the settings in the Personal Safety app, will, if you do not respond in 60 seconds, automatically call the police, reporting your location. Apple announced in November 2021 that it was planning to do something similar.

The crash victims who would most obviously benefit from such technology are those in crashes with no witnesses to call police ("unnoticed run-off roadway"), who survived the crash, and might have lived if help had arrived promptly, but died from their injuries. Such crashes, though, are very rare, about seventy-seven fatalities annually in the US in 2010-2018, (**?**) of the about 35,000 crash fatalities per year in the same time period. (**?**)

A much larger group who could benefit from are those injuries are serious and need prompt medical attention. Dispatching an ambulance automatically, rather than waiting for an eyewitness to call for one, would cut at least several minutes off of the ambulance response time. In a 1996 study on 1990 data for US urban interstates, freeways, and expressways (**?**), the average accident notification time was 5.2 minutes, and the additional time to EMS (emergency medical services) arrival was 6.2 minutes. Evanco estimated that reducing the notification time from 5.2 minutes to 2 minutes would cut fatalities by 15.9%. Even those who might not die may recover more fully and quickly with prompt medical attention, so dispatching an ambulance promptly when one is needed would be beneficial.

On the other hand, we do not want to send an ambulance to every accident scene, because only a small proportion of crashes have severe injury; most are property damage only (PDO) crashes. Ambulances and their crews are expensive and in finite supply.

Given the information available to the police from a phone's automated crash notification, can we build a model that will recommend (or determine) whether to send an ambulance immediately?

1.2. Difficulty in Solving Problem

Such a model will not be perfect, with some false negatives (not sending an ambulance when one is needed) and many false positives (sending an ambulance when one is not needed). We will show that the quality of the model depends largely on what information is available. Some information (location, time of day, day of week, weather) either comes with the automated report or is easy to get. Other information (age and sex of phone's primary user, vehicle likely to be driven by that person) may be very helpful in predicting injury, but getting that information would require instantaneous communication between private and public databases. Being able to interpret the location, (*e.g.*

ORCID(s):

Is that precise location inside an intersection of two roads with high speed limits?) in real time would require planning and preparation.

The problem is both political and ethical as well as technical. How many false positives will we tolerate to have one fewer false negative? We will show that, given such a marginal tolerance p, we can incorporate that tradeoff into the model, but each locality will have to decide that for itself. Implementing such a system would require budgets, cooperation, and possibly legislation, but knowing which data is most useful can help set priorities.

1.3. Machine Learning Challenges

We deal with several machine learning challenges in our study, and their solutions are often as much art as science.

Two Datasets For this study we used two datasets, the US Department of Transportation's National Highway Transportation Safety Administration (NHTSA)'s Crash Report Sampling System (CRSS) 2016-2020 and a census of crash reports in Louisiana 2014-2018. (**?**) Both datasets record about five hundred thousand crashes, with details on the crash and on each vehicle and each person involved. The two datasets are similarly structured.

We will look at the data at the level of each person involved in a crash ("crash person"), as each automated crash notification corresponds to a phone, which corresponds to a person.

The major difference between the two datasets is that, while the Louisiana is a census (all police crash reports), the CRSS is sample that intentionally over-represents more serious crashes. In the Louisiana data, 8.5% of the crash persons were transported to a hospital, while in the CRSS sample, it's 16.1%. Comparing the results will require interpretation and justification.

Feature Selection and Engineering We need to select the features most relevant to crash severity; too many less-relevant features will muddle the model building. CRSS has both "Make" and "Body Type." Do these two features give enough different information that we should use both? If not, which is more useful?

Some features have many values that we can usefully combine into bins or bands. For instance, CRSS has ninety-seven different categories of vehicle body type, and our model building would work better if we could condense those to fewer than ten categories. Is a compact pickup truck more like a car or a standard pickup? We look at the likelihood that a crash vehicle's occupants require transportation to a hospital when making such classification decisions, and classify the compact pickup truck with the cars.

The `AGE` feature has values 0-120. A simple approach would be to put it in decade bands, but in most states in the US, the driving age is 16. The crash severity profiles for new drivers are different than for experienced drivers, so a split at 15/16 makes sense. In our analysis, the crash severity profiles for ages 52-70 are similar to each other but different from 71+, so we broke them into bands there.

We can also merge features into useful new features. In both data sets, we have "Day of Week" and "Hour." We would like to take from each to make "Rush Hour," if it has a different hospitalization profile. When does it start and end? Is morning rush hour different from evening? Does it start earlier on Fridays?

Missing Data As with all real data, many samples (records) have missing values. The Louisiana data is raw, with many values listed as "unknown." For many features (fields, columns) of the CRSS, the authors have created another feature with the missing values imputed. Trusting that those authors understand the data better than we do and may have access to redacted data, we use the imputed features. For the Louisiana data, we will have to impute, delete, or ignore missing data; again, as much art as science.

1.4. Imbalanced Data

Only a small proportion of crashes require immediate medical attention. In the Louisiana data, 8.5% of persons involved in a crash were transported by ambulance. If we built a model that classified all crashes as "Ambulance Not Needed," the model would have 91.5% accuracy, which would be excellent in some other applications, but not here. The toolkit for building models on imbalanced data is well established, but many of the tools only work for continuous data (our data is all categorical), and their use is as much art as science.

Balancing the Data

Loss Function

Metrics Machine learning algorithms work iteratively by evaluating a model, perturbing the model, evaluating the new model, and deciding whether the new model is better; repeat. What do we mean by "better," and how do we measure it?

Hyperparameter Tuning The ML algorithms we will use have some user-set parameters to optimize the model, and they can only be set by trial and error. As we use two datasets, and as we add data to our model, should we use the same hyperparameters, or can they change?

Boosting

1.5. Research Plan

1. On both raw data sets, do cleanup, feature selection, and feature engineering. To the extent possible, make the two engineered data sets the same.
2. Starting with the easiest-to-obtain data (general location, time/day, weather), and interatively adding more data (persons, vehicles, specific location), build and evaluate a model that predicts whether an ambulance is needed.
3. Combine results from the two datasets.
4. Interpret and discuss how the model improves as more data becomes available.

2. Literature Review

2.1. Apps to Detect Crashes

The idea of using the accelerometer in a cell phone to detect a crash and notify the police goes back to at least 2011 with *WreckWatch*, an app prototype by **?**. A few years later, **?** proposed an app that would detect not only a crash but its severity, and send the phone owner's medical information. In September 2019, Google hinted that it would have crash detection in its Pixel phones soon (**?**), and it is now available (**?**). In November 2021, the Wall Street Journal announced that Apple was thinking of introducing such an app in its iPhones and Apple watches in 2022 **?**, but as of this writing, such an app had not come out yet. The App Store does have some unverified third-party apps, like those from **?** and **?**.

2.2. Imbalanced Data

In building a model to predict, based on records of roadside inspections, traffic violations, and previous crashes, future crashes for trucks, **?** described the imbalanced data problem well.

> Initial models "correctly" classified no-crash versus crash instances 99% of the time, but almost never correctly predicting a crash—a major failure in achieving the goal of this analysis and a common issue in unbalanced datasets.

Our work in this paper uses the most straightforward of machine learning (ML) algorithms, the binary classifier. We want to answer, "Should we send an ambulance to this crash," and are using historical data answering for each crash, "Did we use an ambulance?" We want to build a model that will look at the data we have for a crash and return a prediction. To build it, we will separate our data into a training set and a test set; use the training set to build the model, then evaluate the model on the (unseen during model building) test set.

The model will not be perfect. We want to send ambulances to all of the crashes that need one (True Positive), and not send ambulances to crashes that don't (True Negative). Sending an ambulance when one is not needed (False Positive) is unnecessarily expensive, but if we don't send an ambulance when one is needed (False Negative), someone might die unnecessarily.

We can visually organize the success and failures of our binary classification model in a *confusion matrix* (also called *error matrix*, or *contingency table*).

		Prediction	
		N	P
Actual	N	TN	FP
	P	FN	TP

Most machine learning algorithms are designed to maximize *accuracy*, the proportion of classifications that were successful.

$$\text{Accuracy} = \frac{\text{TN} + \text{TP}}{\text{TN} + \text{FP} + \text{FN} + \text{TP}}$$

In many applications, the accuracy is straightforward and useful, but not if the data are imbalanced. In our case, in all of the reported crashes in Louisiana in 2014-2018, there were 645,748 people involved in the crashes, and 55,164 used an ambulance (`OCC_MED_TRANS_CD = A`), 8.5% of the total.

If we built a model that predicted that none of the crashes required an ambulance, our model would have 91.5% accuracy.

		Prediction	
		N	P
Actual	N	590,584	0
	P	55,164	0

$$\text{Accuracy} = \frac{590{,}584 + 0}{590{,}584 + 0 + 55{,}164 + 0} \approx 0.915$$

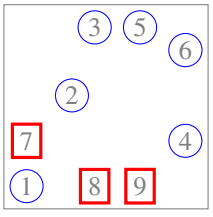
In many applications, 91% accuracy would be good. Why is it different in our context? For two reasons. First, our model is useless because we have entirely misclassified all of the positive samples. Second, accuracy gives the same weight to false positives and false negatives, but in our application, their costs are very different. The cost of a false positive (sending an ambulance when one is not necessary) is measured in money, while the cost of a false negative (not sending an ambulance promptly when one is necessary) is measured in lives. We are willing to trade off sending p number of unnecessary ambulances if it means that we will send one more necessary ambulance. [Choosing the value of p is a question for ethicists and politicians that we will discuss later.]

There are four [citation] categories of methods for building a machine learning model on an imbalanced dataset, and we will use all four in this paper.

- Data-level techniques to artificially balance the dataset
- Modifying the loss function in building the model
- Modifying the model-building algorithm
- Changing the metric in evaluating the model

2.3. Data-Level Techniques for Imbalanced Data

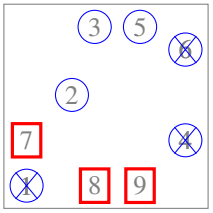
Consider this two-dimensional training dataset, which we will use to illustrate data-level techniques for handling imbalanced datasets. In real problems, of course, the dataset could have a hundred dimensions and a million samples. The six blue circles represent samples (elements) of the majority negative class (“no ambulance”), and the three red squares represent the minority positive class (“ambulance”).



Many algorithms, and variations thereon, have been proposed to balance the two classes before applying a machine learning algorithm to build a model to classify new samples as positive or negative. WARNING: Vast oversimplification ahead. Our goal here is to give the general idea of each method.

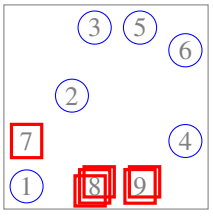
2.3.1. Random Undersampling

Random undersampling balances the two classes by randomly deleting elements of the majority class until the two are balanced. The major drawback of this method is that you throw away information about the majority class. If the majority class is many more times the size of the minority, you lose almost all of the data.



2.3.2. Random Oversampling

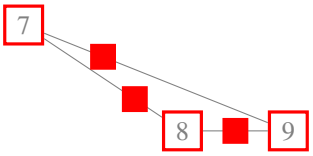
Random oversampling creates duplicates of minority class samples until the sets are balanced. This method has a similar effect to using class weights, introduced below.



2.3.3. Synthetic Minority Sampling TEchnique (SMOTE)

SMOTE (**?**) is one of the most popular oversampling methods for balancing a dataset with continuous numerical data. It creates new synthetic minority samples "between" original minority samples, not necessarily at the midpoint by choosing a number in (0, 1), multiplying the difference (in each dimension) from point A to B by that constant, and adding it to A.

In the diagram, the solid red squares represent new synthetic samples between pairs of original minority-class samples. SMOTE does not consider the positions of the majority-class samples, only considering the difference in number of nodes to bring the two classes closer to parity.



One challenge with SMOTE is that it is only useful for datasets with continuous numerical data, and our data is almost all categorical. What is between "car" and "school bus," or between "parking lot" and "highway"? SMOTE has a variant, SMOTE-NC (Nominal and Continuous) that can handle datasets with some nominal (categorical) features, but most of the features need to be continuous; thus, we will not be able to use SMOTE or similar techniques for our work.

2.3.4. Tomek's Links

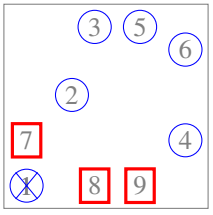
? proposed a method of undersampling that assumes that the majority and minority classes should (at least locally) be clustered. If an sample A of the majority class and a sample B of the minority class are each other's nearest neighbors, then one of them is not clustered with its own class. Since we are trying to undersample the majority class, assume that the element of the majority class is noise (or an error, or just not useful), and delete it.

In the diagram below, samples #1 and #7 are Tomek links, because they are each other's nearest neighbors and of different classes. Samples #4 and #9 are not Tomek links, because while 9 is 4's nearest neighbor, 9's nearest neighbor is 8, not 4.

In the context of modeling crash severity from police reports, why would sample #1 not need an ambulance when its characteristics are so close to those of #7 and not near most of the other crashes without serious injury? The reason

could be errors in the records, or luck/providence/fate. It could also be that the difference between property damage only and serious injury is influenced by thousands of variables we cannot measure or know, all of the physics of crash forces acting on the bones and structures of the human body. The best we can say is that the outcome in #1 cannot be predicted by the information that we have, so that sample will not help in constructing a model based on the available data; therefore, we can reasonably delete it from the training set.

Tomek's Links can also be run iteratively. Sample #7 had #1 as its nearest neighbor, but once #1 is deleted, then #2 and #7 are each other's nearest neighbors of different classes, thus are Tomek links, and we can delete #2.



2.3.5. Oversampling Image Data

Extracting knowledge from a database of tabular numerical or categorical data is difficult, but a database of images is a challenge of a different magnitude. An imbalanced labeled image dataset for crash prediction modeling might be a thousand images taken ten seconds before a crash and a million images taken ten seconds before ... nothing happened. Deep neural networks (DNN) and (deep) convolutional neural networks (DCNN and CNN) are common methods for image data. **?** introduced Generative Adversarial Networks, which can be used to generate synthetic samples to balance the dataset. Given the power of the tools for image recognition, many researchers make non-image data look (to the computer) like images to take advantage of the tools.

2.4. Modifying the Loss Function

Machine learning algorithms generally work iteratively by picking a starting point for the constants in the model (often a random guess), measuring the error, making a small perturbation in the model constants, measuring the new error in the candidate model, and comparing the two. If the new error is less, use the candidate model; if not, go back to the old one. Repeat.

A common way to measure the error in binary classification is log loss (binary cross-entropy loss, logistic loss). For each sample in the training set we have the answer to the question (the *label*), 0 for "no ambulance," 1 for "ambulance," and the candidate model gives a probability $p \in (0, 1)$ that this sample will need an ambulance. The log loss is the sum over the samples of the log of the error. If the sample is in the majority class, the true value is 0, and if the model gives a value of p, then $\log(1-p)$ gives $\log(1) = 0$ if the model is perfectly correct and $\log(0) = -\infty$ if the model is perfectly wrong on this sample. Similarly, for a sample in the minority class, $\log(p)$ gives $\log(1) = 0$ if the model correctly classifies the sample. If y is the label, then the log loss for each sample is (**?**)

$$L(y, p) = -\left(y \log(p) + (1-y) \log(1-p)\right)$$

Note that if $y = 0$, then the first term is 0 for any value of p, so only the second term (majority class) is relevant; correspondingly, if $y = 1$, then only the first term is relevant, so a clearer expression might be

$$L(y, p) = -\left(\left[\text{Loss if minority class } (y = 1)\right] + \left[\text{Loss if majority class } (y = 0)\right]\right)$$

Since the logs of $p \in (0, 1)$ will be negative, the negative in front makes the loss positive, and the iterations of the algorithm will seek to minimize it.

2.4.1. Class Weights (Sample Weights, Cost Sensitive Analysis)

Class weights change the error metric, giving more weight to misclassification of minority class samples (**?**). Giving double weight to misclassified minority class samples would have the same effect as duplicating all of the negative class samples in oversampling. To achieve balance in the contribution of the two classes to the loss function,

$$\text{Let } r = \frac{\text{Total number of samples}}{\text{Number of minority samples}} \qquad \text{Let } \alpha = \frac{r}{r+1} \qquad 1 - \alpha = \frac{1}{r+1}$$

$$L(y, p, \alpha) = -\left(\alpha y \log(p) + (1-\alpha)(1-y) \log(1-p)\right)$$

2.4.2. Focal Loss

This method is recent, but has appeared in the crash analysis (see **?** referenced below). Focal loss (**?**) adds another factor to the loss function that gives more weight to samples that are badly misclassified, and less weight to samples that are slightly misclassified.

$$L(y, p, \alpha, \gamma) = -\left(\alpha y \log(p)(1-p)^{\gamma} + (1-\alpha)(1-y)\log(1-p)\,p^{\gamma}\right)$$

The paper by Lin et al. tested values of $\gamma \in [0.0, 5.0]$, and found that $\gamma = 2.0$ gives good balance, but the best value depends on the dataset and the goals. Yu et al's paper using focal loss for real-time crash prediction allowed the γ for minority and majority classes to have different values.

$$L(y, p, \alpha, \gamma_1, \gamma_2) = -\left(\alpha y \log(p)(1-p)^{\gamma_1} + (1-\alpha)(1-y)\log(1-p)\,p^{\gamma_2}\right)$$

2.4.3. Comparison of Loss Functions

The table below compares the three loss functions. For α, we will assume that the minority class is 10% of the dataset, so $r = 10 \rightarrow \alpha = r/(r+1) = 10/11 \approx 0.9090$. For focal loss, we will use $\gamma = 2$.

Machine learning algorithms use the loss function when comparing two candidate models, only asking which one is less, with no concern for the actual magnitude. For this reason, the choice of base for the logarithm is inconsequential (we arbitrarily choose base 10 for the chart below); also, that the raw focal loss values are each less than the raw class weights, which are each less than the raw log loss, is not relevant, so we have included normalized values for comparing the three loss functions.

Class	y	p	Raw			Normalized		
			Log Loss	Class Weights	Focal Loss	Log Loss	Class Weights	Focal Loss
Minority	1	0.9	0.04576	0.04160	0.00042	0.04560	0.08292	0.00464
	1	0.7	0.15490	0.14082	0.01267	0.15438	0.28069	0.14136
	1	0.5	0.30103	0.27366	0.06842	0.30002	0.54548	0.76309
Majority	0	0.5	0.30103	0.02737	0.00684	0.30002	0.05455	0.07631
	0	0.3	0.15490	0.01408	0.00127	0.15438	0.02807	0.01414
	0	0.1	0.04576	0.00416	0.00004	0.04560	0.00829	0.00046

Note that, with focal loss, most of the total loss comes from the one badly misclassified minority sample, so to minimize the loss, the algorithm needs to do a better job classifying that sample.

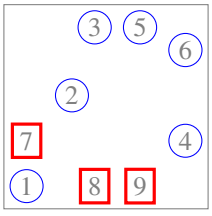
2.5. Algorithm-Level Methods for Imbalanced Data

2.5.1. Bagging

"Bagging" is short for Bootstrapped Aggregating, a variation on random undersampling (**?**). In general, bagging takes many random subsets (with replacement) of the samples, run the classifier on each subset, then aggregate the results. In imbalanced data applications, each subset of the samples is all of the n minority samples and n randomly chosen majority samples.

Balanced Random Forest [need citation] is a form of bagging.

In our example, bagging would make a subset of the data with the three minority-class samples (#7, 8, and 9), and three randomly chosen from the majority-class samples, run the classifier; repeat some number of times. Use an ensemble classifier to merge the results.



2.5.2. Boosting

Boosting is an iterative method that runs the classifier multiple times. At the end of each iteration, it determines which samples would be misclassified under the current model. In the next iteration, the classifier gives higher weight to the misclassified samples, improving the model on marginal cases. While boosting is not just for imbalanced data, the challenge in imbalanced data is that the minority class samples get misclassified, so boosting would help. A popular implementation is AdaBoost, introduced by **?**.

2.6. Using Different Metrics to Evaluate a Model built on Imbalanced Data

Our goal in building a model is to make an informed guess about a future situation, in our case data from an automated call from the phone of a person involved in a crash. Does the person need an ambulance? We don't know for certain, but using past examples we want a model that will make an informed guess.

To build a model (in supervised learning), we have a set of data (historical data in our case) for which we know the answer to the question, "Did this crash person need an ambulance?" We split the data into two parts, training and test, use the training data to build a model, then evaluate the quality of the model using the test data, for which we now have both the actual answer to the question and the model's educated guess.

When building a model, we test many different models with different algorithms and hyperparameters (user-set parameters, like the γ in Focal Loss) and see which model best solves our problem. To compare two models, we need a metric.

When testing a candidate model, for each sample in the test set, the model returns a probability $Pr \in (0, 1)$ that the sample is a member of the positive (minority) class, in our case, the probability that the crash person needs an ambulance. We then use a threshold (0.5 by default) to classify the prediction as negative (N, 0) if the probability is less than the threshold, and positive (P, 1) if more. Then we compare the prediction with the actual classification (ground truth), which we can organize into a confusion matrix (truth table).

		Prediction	
		N	P
Actual	N	TN	FP
	P	FN	TP

The most obvious metric is *accuracy*, the proportion of classifications that were successful.

$$\text{Accuracy} = \frac{\text{TN} + \text{TP}}{\text{TN} + \text{FP} + \text{FN} + \text{TP}}$$

Because accuracy gives the same weight to the two classes, it may not be the best metric for imbalanced data sets (because it may overfit the majority class), nor for situations where the cost of a false positive (sending an ambulance when one is not needed) is different from the cost of a false negative (not sending an ambulance when one is needed). Both situations apply to our problem, so we need to consider different metrics, some of which are affected by data imbalance, and some not.

2.6.1. Recall

Recall, or the True Positive Rate (TPR), in our context is, of the calls that actually needed an ambulance, what proportion did the model classify correctly? Increasing recall would save lives. We really want this number to be high, and are willing to trade off other errors to achieve that goal.

$$\text{Recall} = \frac{\text{TP}}{\text{FN} + \text{TP}}$$

Recall only considers the positive (minority) class of the dataset, so it is unaffected by class imbalance.

2.6.2. True Negative Rate

True Negative Rate (TNR, Selectivity, or Specificity), in our context is, of the calls that did not need an ambulance, what proportion did the model classify correctly?

$$\text{TNR} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

TNR is also not affected by class imbalance because it only considers the classification of the negative (majority) class.

2.6.3. Precision

Precision in our context is, of the ambulances the model would say to send, what proportion were actually needed? Increasing precision would save money.

$$\text{Precision} = \frac{\text{TP}}{\text{FP} + \text{TP}}$$

Note that precision considers elements of both the majority and minority classes, so class imbalance affects the precision, but we can fix that.

2.6.4. Balanced Accuracy and Balanced Precision

Balanced accuracy solves the first problem with accuracy, that it is biased towards the majority class. To balance accuracy, start with the definition of accuracy and scale the majority (negative) class elements (True Negative and False Positive) by the proportions of the positive to negative class, P/N. The results turns out to be the average of the true positive rate and the true negative rate (derivation in the technical paper). That TPR and TNR are unaffected by class imbalance confirms that balanced accuracy is unaffected.

$$\text{Balanced Accuracy} = \frac{\left(\text{TN} \cdot \frac{\text{P}}{\text{N}}\right) + \text{TP}}{\left(\text{TN} \cdot \frac{\text{P}}{\text{N}}\right) + \left(\text{FP} \cdot \frac{\text{P}}{\text{N}}\right) + \text{FN} + \text{TP}} = \frac{\text{TPR} + \text{TNR}}{2}$$

In the same way, we can make a balanced precision. It can also be written in terms of TPR and TNR, (derivation in the technical paper), or more naturally in terms of TPR and the false positive rate (FPR), the proportion of the negative (minority) class that the model misclassified as positive. In our application, FNR is the proportion of crash persons who did not need an ambulance to whom the model recommends that we should send an ambulance. We have not seen this metric in the literature. We developed it because balanced accuracy is used extensively in the literature, and it seemed natural to want to balance a useful metric (precision) in the same way that accuracy has been balanced.

$$\text{Balanced Precision} = \frac{\text{TP}}{\text{TP} + \left(\text{FP} \cdot \frac{\text{P}}{\text{N}}\right)} = \frac{\text{TPR} + (1 - \text{TNR})}{\text{TPR}} = \frac{\text{TPR} + \text{FPR}}{\text{TPR}}$$

2.6.5. F1 and G-Mean

These two popular metrics are each the combination of two other metrics.

F1 is the harmonic mean of Precision and Recall. It is often weighted to emphasize one or the other; if $\alpha = 0.5$, the weighted is the same as the original. We can also substitute Balanced Precision (above) to make a Balanced F1.

$$\begin{aligned}
\text{F1} &= \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} \\
\alpha\text{-weighted F1} &= \frac{1}{\frac{\alpha}{\text{Precision}} + \frac{1-\alpha}{\text{Recall}}} \\
\text{Balanced F1} &= \frac{2}{\frac{1}{\text{Balanced Precision}} + \frac{1}{\text{Recall}}}
\end{aligned}$$

Gmean is the geometric mean of Precision and Specificity (TNR).

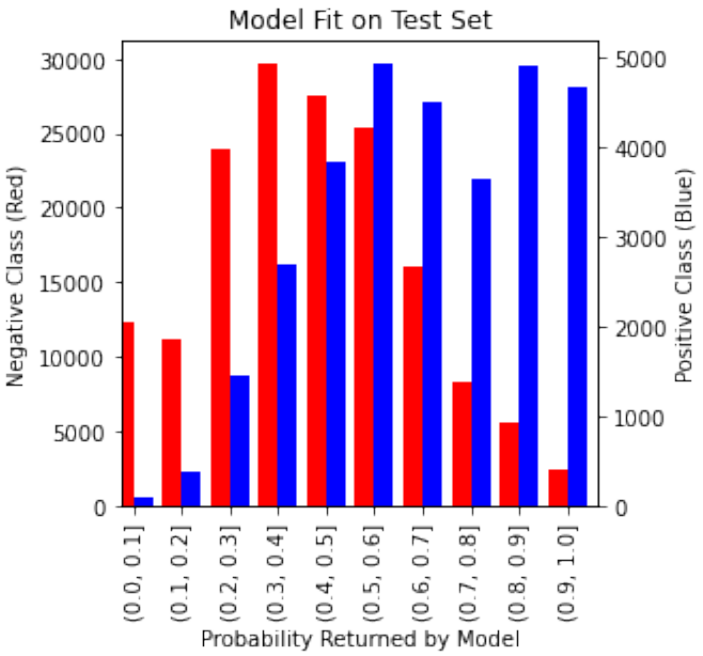
$$\text{Gmean} = \sqrt{\text{Precision} \times \text{Specificity}} = \sqrt{\frac{TP}{TP + FP} \times \frac{TN}{TN + FP}}$$

In this paper we have a binary classification model, but **?** adapted class-weighted F1 for classification of multiple classes in a multi-modal recommendation system. That paper has an excellent discussion of why and how they applied the class-weighted F1 metric in post-processing, because the metric is not differentiable.

2.6.6. Over-Predicting the Minority Class

One tests a model by using it to classify the samples in a test subset of the data. For each sample, the model gives a probability $Pr \in (0, 1)$ that the sample belongs to the positive class. Ideally, most of the samples will fall close to the extremes, clearly classified as either positive or negative, indicating that our model is strong and is more likely to correctly classify future samples, but in practice many samples will not be so clearly classified.

The bar graph below illustrates one model classifying the samples of the CRSS dataset into negative (red, ambulance not needed, majority class) and positive (blue, ambulance needed, minority class). The horizontal axes represents the probability $Pr \in (0, 1)$ given by the model that a sample belongs to the class, and the vertical is the number of samples that fall in each probability range.



By default, one uses a threshold $Pr = 0.5$ to assign the samples to the negative (majority, $Pr < 0.5$) or positive (minority, $Pr > 0.5$) classes, but we are free to change that threshold. In the bar graph, all red elements to the left of the threshold are TN, all red elements to the right are FP, all blue elements to the left are FN, and all blue elements to the right are TP. The threshold is a hyperparameter that affects all of the above metrics; for example, decreasing the threshold would shift samples from TN to FP, and from FN to TP. Decreasing the threshold would over-predict the minority class, and increasing it would under-predict.

The chart below gives metrics on that model on the CRSS dataset for different thresholds t. The last column is the marginal tradeoff between unneeded ambulances sent and needed ambulances not sent; for instance, at threshold $p = 0.50$, the difference between $p = 0.45$ and $p = 0.55$ is 5.99 ambulances unnecessarily sent for every needed ambulance not sent. If we (as a society) believe that that cost is too high and we should send fewer ambulances, we should choose a larger threshold value and under-predict the minority class. For the purposes of this paper we have arbitrarily chosen 10 as the number of additional false alarms we are willing to tolerate for each additional ambulance sent when needed, so we could change the threshold to somewhere in $0.3 < t < 0.4$, over-predicting the minority class.

t	TN	FP	FN	TP	TPR	FPR	$\frac{\Delta TPR}{\Delta FPR}$	$\frac{\Delta FP}{\Delta TP}$
0.00	0	162200	0	31083	1.00	1.00	0.02	246.76
0.10	12322	149878	84	30999	1.00	0.92	0.09	58.08
0.20	23556	138644	455	30628	0.99	0.85	0.26	20.33
0.30	47541	114659	1895	29188	0.94	0.71	0.37	14.10
0.40	77254	84946	4590	26493	0.85	0.52	0.60	8.74
0.50	104764	57436	8425	22658	0.73	0.35	0.87	5.99
0.60	130098	32102	13362	17721	0.57	0.20	1.23	4.25
0.70	146094	16106	17860	13223	0.43	0.10	1.77	2.95
0.80	154351	7849	21506	9577	0.31	0.05	3.09	1.69
0.90	159855	2345	26404	4679	0.15	0.01	8.00	0.65
1.00	162200	0	31083	0	0.00	0.00	14.18	0.37

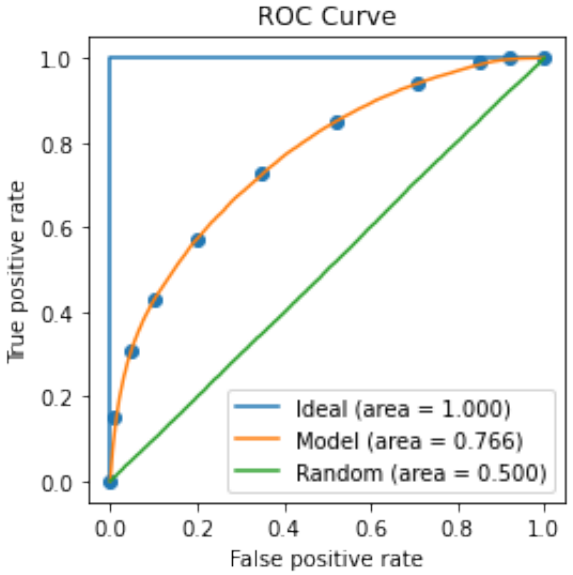
2.6.7. Area Under the ROC Curve [Needs Citations]

The Receiver Operating Characteristic (ROC) shows how much the positive and negative classes are interleaved. It is a parameterized curve following the probability threshold from $t = 0$ to $t = 1$, and plotting the true positive rate (TPR) versus the false positive rate (FPR). The orange "Model" curve in the graph below is from the same dataset predictions in the chart above. The eleven points correspond to the (FPR,TPR) coordinates in the chart, going from $t = 0.0$ in the upper right to $t = 1.0$ in the lower left. The next-to-last column in the chart gives the slope of the ROC curve.

If the model gave entirely random results, the bars in the bar chart above would all have the same height (1/20 of the number of elements in the class), and the slope of the ROC curve would be 1, illustrated by the green line, and the area under the curve (AUC) would be 0.5. If our model and dataset were ideal, with all of the negative elements having $Pr = 0.0$ and all of the positive elements having $Pr = 1.0$, the ROC curve would be the blue Γ, with area under the curve of 1.0. In practice, the area under the curve is between 0.5 and 1.0.

The area under the ROC curve is useful for comparing different models on the same dataset. A higher area generally means that the model more clearly separates the two classes, and is more likely to correctly classify elements of a different (but related) set of samples.

Because both TPR and FPR are unaffected by class imbalance, the ROC curve is as well.



2.7. Imbalanced Data in Crash Analysis

Imbalanced data is a frequent concern in crash analysis. In crash prediction, "non-crash" samples are much more numerous than "crash" samples.

2.7.1. Oversampling

? used SMOTE to balance the dataset of crashes on Chicago's Eisenhower expressway. **?** used SMOTE to study crashes on urban arterials. **?** used SMOTE to consider risky driving behavior in crash prediction. **?** used SMOTE in building a a real-time conflict prediction model. **?** used SMOTE in studying crash prediction for collision avoidance systems. **?** compared three oversampling methods, random over-sampling, SMOTE, and adaptive synthetic sampling, for crash data analysis. **?** compared three oversampling methods studying contributing factors in fatal crashes in Ghana, and the same group applied the majority weighted minority oversampling (MWMOTE) method to study multiple fatal injury crashes (**?**).

2.7.2. Multiple Techniques

? used SMOTE and maximum dissimilarity undersampling to balance an Australian dataset to build a model to predict crashes, while in a later paper Schögl used a balanced bagging approach (**?**). **?** used SMOTE and Tomek's links in studying crash potential in lane-changing behavior. **?** used undersampling, SMOTE, cost-sensitive algorithms, and boosting. **?** used the proximity weighted synthetic oversampling technique (ProWSyn) method to build a traffic violation prediction model. **?** compared Synthetic Minority Over-Sampling Technique for panel data (SMOTE-P) with Random Under-sampling of the Majority Class (RUMC) technique, Cluster-Based Under-Sampling (CBUS), and mixed resampling to identify explanatory factors that affect the crash risk of buses in Hong Kong. **?** used ENN-SMOTE-Tomek Link (EST) in estimating crash risk in lane changing.

2.7.3. Image (or Image-like) Data

? used Deep Neural Network (DNN) models with a Mean Squared False Error loss function to analyze images from front-facing cameras in cars on a UK roadway to predict crashes. **?** used Generative Adversarial Networks (GAN) to overcome data imbalance for their incident detection model. **?** found that a variational autoencoder was more useful than SMOTE, ADASYN, and GAN for generating minority samples to balance crash and non-crash events. **?** used Deep Convolutional Generative Adversarial Networks technique with random undersampling to use image and image-like data to build an accident-prediction model for a section of highway in Chile. **?** used Wasserstein GAN (WGAN) and random undersampling to study the transferability of a model built on one dataset to other datasets. **?** compared deep convolutional generative adversarial network (DCGAN) with SMOTE and random undersampling to study the effects of proactive traffic safety management strategies such as variable speed limits and dynamic message signs.

2.7.4. Other

? used bagging in predicting crashes for trucks and finding ways to improve truck safety. **?** used boosting in studying the effects of ramp metering on traffic safety. **?** implemented the new focal loss technique in real-time crash prediction. **?** used undersampling to analyze factors predicting risky and safe driving. **?** used cost-sensitive semi-supervised logistic regression (CS3LR) for hit-and-run analysis.

2.7.5. Imbalanced Data in Other Transportation Areas

? used the Adaptive Synthetic Sampling Approach (ADASYN) on a dataset of foot-by-foot track geometry and tonnage to identify the factors that predict rail defects. **?** developed a hierarchical over-sampling bagging method based on Grey Wolf Optimizer (GWO) algorithm and Synthetic Minority Over-sampling Technique (SMOTE) to study lane changing for autonomous vehicles. The data was severely imbalanced because lane changing is rare compared with lane keeping. **?** used SMOTE and Tomek links and "average balanced recall accuracies" for flight delay prediction. **?** used bagging for ride-hailing demand prediction.

In crash severity, most reported crashes are just property damage only (PDO), and many PDO crashes aren't even reported. Other aspects of transportation also have imbalanced data, including

- **?** used similar data and addressed the challenges we'll have with it.
- **?** works with several imbalanced methods. Use this paper as a model.

2.7.6. Ambulances

- **?** has a full-page table categorizing studies of ambulance location, relocation, and dispatching using different optimization methods.

3. Datasets

The dataset we want for this study, unfortunately, does not exist. Such a dataset would have several years of automatic notifications from cell phones to police of a crash, with accompanying data on (a) whether it actually was a crash, (b) whether the user of that phone needed an ambulance, and (c) whether anyone else involved in the crash needed an ambulance. The dataset does not yet exist because the technology is too new. The app developers must have testing data, but we have not seen any publicly available.

To do the best work we can with what is available, we need an appropriate proxy dataset, but that will be challenging. We do not know how well the apps detect a crash, currently or in the future. For instance, if the crashes the apps detect were those crashes where the airbag deploys, they would miss most of the crashes requiring an ambulance. (These data are from CRSS; see below.)

		Air Bag Deployed	
		No	Yes
Ambulance	No	479,287	61,377
	Yes	64,699	38,911

The apps using the phone's accelerometer will have a hard time distinguishing low-speed crashes from hard braking, so the apps will not detect many non-injury crashes; therefore, we may need to either underrepresent non-injury crashes in our work, or start with a database that does that, like CRSS.

For this study we used two datasets, the Crash Report Sampling System 2016-2020 (**?**), and a tabular assembly of all of the Louisiana crash records 2014-2018. While the CRSS data and a helpful guide are available online, the Louisiana data is not publicly available.

3.1. CRSS: Crash Report Sampling System

CRSS, as its name suggests, is a curated sample of crashes in the US, scrubbed of personally identifying information and with missing values imputed. It is intentionally not a representative sample, but intentionally over represents serious crashes; for instance, "crashes with killed or injured pedestrian" represent 9% of the crashes in the dataset but only 1.9% of crashes in the US. Its sample design is given on page 18 of the CRSS Analytical Users Manual (**?**). Because the dataset is not representative, we have to be careful in drawing inferences. Since we do not know, in detail, the present and future capabilities of the cell phone app, this dataset that overrepresents more serious crashes may be a good proxy, and we will use it as such.

3.2. Louisiana Data

The structure of the Louisiana data is similar to CRSS. Key differences are that it is a census of all crash reports, and missing data is not imputed. While CRSS data is given entirely in attribute codes, many fields in the Louisiana data, like city and street names, are text, uncorrected; the city of Shreveport is spelled at least nineteen different ways.

3.3. Imputing Missing Data

All data is dirty, with incorrect and missing values. The CRSS dataset is reasonably correct in that only the values that should appear in a feature actually appear; for instance, a feature that should have numerical values does not have text values for a few samples. For CRSS, we will not tackle the question of whether the values are correct, but most of the features have values that signify "Missing" or "Unknown," and we want to impute values for those incomplete samples, using data in other features.

The methods for imputing those values are well developed. If the feature were continuous numeric, we could use the Numpy, Pandas, and scikit-learn methods to replace missing values in a feature with the mean or median of that feature. For categorical data, the same packages will impute the most common value in that feature.

In CRSS, the data is almost all categorical, and the data is so imbalanced that the most common value often corresponds to a minor crash with no injury. To impute values using the most common value in the feature would make our dataset even more imbalanced. For instance, of the 644,274 people in the dataset, 429,574 (67%) of the people have "No Apparent Injury," and 21,595 (3.3%) are "Unknown/Not Reported." Assigning the most common value in that feature to the missing elements would worsen the imbalance; a better method would build a model of the data and use the model to fill in the holes.

Scikit-learn does have an experimental multiple imputation method, but it only works for continuous data.

The CRSS authors used a Sequential Regression Multivariate Imputation (SRMI) method to impute missing data in some features, employing the implementation in the University of Michigan's "IVEware: Imputation and Variance Estimation Software" (**?**).

In SEX, for instance, the samples attributes "Not Reported" and "Reported as Unknown" are assigned to either "Male" or "Female" in the feature SEX_IM.

		Imputed	
		Male	Female
Original	Male	339,365	0
	Female	0	278,766
	Not Reported	8,748	7,168
	Reported as Unknown	5,799	4,428

The CRSS authors did not impute missing values for all of the features, including some we want to use. The reasons they gave for not imputing more features include wanting to be consistent with the features and methods in the predecessor to CRSS, the National Automotive Sampling System General Estimates System (NASS GES), 1998-2015, which also used IVEware's SRMI in 2011-2015 (**?**). Which features are imputed even changes from year to year, for instance with RELJCT1_IM being discontinued in 2019 and brought back in 2020. Wanting all of the features we were to use to have missing values imputed, we followed CRSS's methods to run IVEware ourselves on the data, using the features imputed by CRSS to check that our process was similar to theirs.

The table below gives the frequency of values in the INJ_SEV feature. The original values include "9: Unknown/Not Reported." The last two rows show the results from the CRSS authors' imputations, and our imputations trying to replicate their method.

INJ_SEV Imputed INJ_SEV Original		0	1	2	3	4	5	6
No Apparent Injury	0	429574	0	0	0	0	0	0
Possible Injury	1	0	95761	0	0	0	0	0
Suspected Minor Injury	2	0	0	57299	0	0	0	0
Suspected Serious Injury	3	0	0	0	32556	0	0	0
Fatal Injury	4	0	0	0	0	5587	0	0
Injured, Severity Unknown	5	0	0	0	0	0	1883	0
Died Prior to Crash	6	0	0	0	0	0	0	19
Unknown/Not Reported	9	14986	4065	1401	876	114	153	0
Unknown/Not Reported	9	15423	3104	1777	1061	180	49	1

Imputation methods are given on page 19 of the CRSS Analytical User's Manual and in the CRSS Imputation report. The imputation report gives the model selection criteria used in IVEware, and we have used those in our work, particularly 10 cycles, the minimum marginal r-squared required for a predictor to be included in the model set to 0.01, and the maximum number of predictors in a model set to 15 (footnotes on pages 7 and 8).

Two feature's imputations are inexplicably different from the others, MAX_SEV, the maximum injury severity in a crash, and NUM_INJ, the number of people injured in the crash. Not only are missing values imputed, but some other values are changed. Another odd imputation is VEVENT_IM, the imputed values of M_HARM, the most harmful event. Category 4, "Gas Inhalation," does not appear any of the original samples, but three of the missing entries get imputed to that category. Perhaps these samples were imputed by hand.

Ambulance Dispatch

Original		Imputed 0	1	2	3	4	5	6	8
No Apparent Injury	0	120,142	1,300	422	266	29	51	0	0
Possible Injury	1	0	58,392	222	125	16	0	0	0
Suspected Minor Injury	2	0	0	40,247	93	20	0	0	0
Suspected Serious Injury	3	0	0	0	26,767	9	0	0	0
Fatal	4	0	0	0	0	5,115	0	0	0
Injured, Severity Unknown	5	0	16	6	2	1	1,250	0	0
Died Prior to Crash	6	0	0	0	0	0	0	11	0
No Person Involved in Crash	8	0	0	0	0	0	0	0	95
Unknown/Not Reported	9	2,859	887	383	290	38	23	0	0

We considered using `MAX_SEV` as our target variable, but ended up not using it at all. We instead decided to use `HOSPITAL`, which "identifies the mode of transportation to a hospital or medical facility provided for this person." Five of the values of that data element correspond to the person being transported to a hospital by some means, and the other four either not transported or unknown. We binned it as in this chart.

`HOSPITAL` Field in CRSS

Binned	Original		Count
FALSE	Not Transported	0	522,801
	Other	6	4,341
	Not Reported	8	12,447
	Unknown	9	1,075
TRUE	EMS Air	1	2,549
	Law Enforcement	2	605
	EMS Unknown Mode	3	30,368
	Transported Unknown Source	4	8,926
	EMS Ground	5	61,162

3.4. Lit Review: Imputing Missing Data in CRSS [Rough]

- **?** does a thorough description of imputing missing data in CRSS. Does not mention IVEware. Also deals with imbalanced data well. Need to spend time with this article.
- **?** says CRSS "can be weighted to produce annual national estimates." Also, "Police-reported crash sampling methods changed when NHTSA converted from NASS GES to CRSS, which may have affected the comparability of the 2017 data on all crash involvements with earlier years."

  In this study, "Imputed data were utilized when available to account for missing data."
- **?** gives a thorough description of CRSS. They took out CRSS-imputed variables. Also removed post-accident information, as it was not relevant. They imputed missing continuous variables, but don't say how. They left missing categorical variables as "Unknown" and "Missing" categories.

  > Employing descriptive analytics, we distinguished and removed variables with a large percentage of missing values (more than 70%), as well as the identification, irrelevant, repetitive, and CRSS-imputed variables. We also removed the variables with post-accident information, such as whether the vehicle was towed afterward or the number of injured people. Using such variables contradicts the basic assumption of time order in causal relations, where a cause should precede its effect. Furthermore, we handled other missing values by considering them separate categories for nominal variables and imputing numeric ones.

- **?** used CRSS but did not mention missing or imputed data.

- **?** says that "CRSS is a representative sample of all police-reported crashes in the United States," which is not true. They used FARS and CRSS as their primary data sources, but did not mention imputed or missing data.

  "Each record in CRSS includes a statistical weight to indicate the number of crashes in the population represented by each record in the sample."

- **?** says that "CRSS sampling weights were used in those data to generate national estimates," and "The CRSS data set handles missing data for some variables by statistically imputing values, which were used when available."

- **?** uses the phrase, "restricted access database." I should use that for the Louisiana crash database.

- **?** just dropped samples with missing values.

- As far back as 2002, NHTSA was working on multiple imputation methods for its related database, FARS. (**?**)

4. Methods

Here we give an overview of our methods with the CRSS data. For further details, see the technical paper and code. As the Louisiana data is not public, we give just a sketch of that work.

4.1. Feature Selection

In selecting features from the CRSS dataset, we chose features that could be known or inferred at the time of the crash without an eyewitness. We separated them into three sets in terms of how hard the data would be to get in good quality: Easy, medium, and challenging. The medium level would require coordination of private (cell service providers, Apple Google) and public databases (police, government vehicle registries). The challenging ones would require implementation and/or development of technology to, for instance, to correlate several notifications from different phones at about the same time from about the same location to see that they are probably the same crash event, or for the phone to distinguish between a pedestrian and a crash person in a moving car, possibly by analyzing the acceleration over the last several minutes before the crash.

Location

Easy

Urban/Rural

Interstate

Medium: Requires precise location and maps

Position relative to the road

Position relative to an intersection

Medium: Requires up-to-date maps

Work Zone

Time

Easy

Hour

Weekday/Weekend

Month

Weather

Medium: Requires detailed maps per time of day

Lighting Conditions

Phone User

Medium: Requires identifying phone user

Age

Sex

Medium: Requires storing acceleration data for several minutes

Pedestrian / In moving vehicle

Hard: Requires identifying the vehicle used by phone user

Make of vehicle

Body type of vehicle

Hard: Requires identify licensing of phone user (driver) or ages from several phones (students)

School bus

Correlate Multiple Notifications

Hard

Number of people

Number of vehicles

4.2. Missing Data

The CRSS `HOSPITAL` feature, which we used as our target feature, has missing values. CRSS has imputed missing values for several features, but not for `HOSPITAL`.

For its imputed features, CRSS used IVEware, which uses sequential regression multivariate imputation (SRMI).

We first tried to use scikit-learn's SimpleImputer, which imputed all of the missing values as "0: Not Transported."

Then we tried scikit-learn's IterativeImputer, but it only works with numerical continuous data, and all of our features are unordered categorical.

4.3. Feature Binning

Above we illustrated how the `HOSPITAL` field had nine attributes, but we put them into two bins. We were interested only in whether a crash person was transported to a hospital, not exactly how, so for our purposes, "EMS Air" and "EMS Ground" were equivalent. Several of the features had about ten attributes that we could assign to fewer bins in obvious ways. Details in the technical paper.

Some features had attributes that did not fall into bins in obvious ways. Is a "Compact Pickup," a designation that only existed in 2016, more like a sedan or a light pickup? To draw the lines, we looked at how that attribute correlated with `HOSPITAL`. The rate at which crash persons in compact pickups were transported to the hospital was (statistically) indistinguishable from sedans (15%), but very different from light pickups (11%).

For `AGE_IM`, age with missing values imputed, we had values 0-120 that we wanted to assign to bins. The obvious way would have been to do it by decade, but when we looked at the hospitalization rates, patterns emerged.

In most states in the US, teens can get a driver's license at age 16, but in most states they can get a learner's permit earlier, so in the chart below, some of the 14 and 15 year olds may be the driver.

In the chart, we see a big shift between 14 and 16, both in the number of crash people and the hospitalization rate. There is also a big shift between 18 and 21 in hospitalization rate. Where exactly to draw the line is open for debate, but we made 16-18 a separate group. The next big shift was in the 50's, and we made 19-52 the next group.

Age	Percent of Total Crash Persons	Hospitalization Rate (%)
10	0.53	16.58
11	0.51	15.51
12	0.52	16.54
13	0.54	16.72
14	0.63	17.56
15	0.88	15.21
16	1.65	13.46
17	2.18	14.25
18	2.65	14.41
19	2.67	15.47
20	2.58	14.91
21	2.51	15.65
22	2.56	15.43
23	2.44	15.57
24	2.38	16.01
25	2.37	15.50

Full chart in the technical paper.

4.4. Feature Engineering

From the features in the dataset, we can create new features that may be relevant, like “Rush Hour,” combining hour with day of week. During rush hour, the number of crashes increases, but their average severity goes down. We drew the lines distinguishing “Rush Hour” from "Not Rush Hour” based, again, on the correlation to hospitalization. Interestingly, the hospitalization rates varied by age, and varied by sex, but varied differently by age within sex. Hospitalization rates for young children are low, perhaps due to child seat requirements, but after young childhood, as age increases, the hospitalization rate increases for both men and women, but younger women have a lower hospitalization rate than men, and older women have a higher rate. We created an `AGE_SEX` feature.

Crash Person Count

Age	Female	Male	Total
00-06	12331	12639	24970
07-15	16623	16726	33349
16-18	20158	21622	41780
19-52	175528	219931	395459
53-70	49750	65228	114978
71+	15972	17766	33738
Total	290362	353912	644274

Hospitalization Count

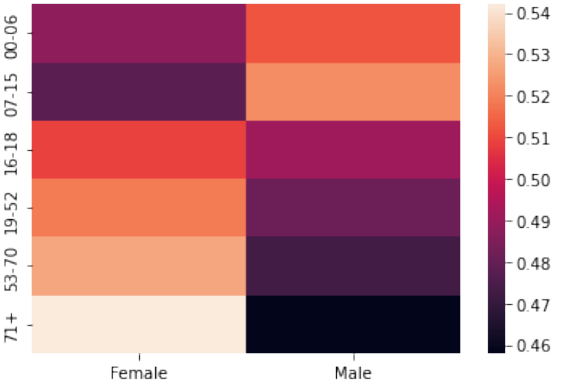
Age	Female	Male	Total
00-06	1448	1560	3008
07-15	2529	2779	5308
16-18	2900	2998	5898
19-52	28843	33514	62357
53-70	9302	10947	20249
71+	3503	3287	6790
Total	48525	55085	103610

Hospitalization Rate

Age	Female	Male	Total
00-06	0.117	0.123	0.240
07-15	0.152	0.166	0.318
16-18	0.144	0.139	0.283
19-52	0.164	0.152	0.316
53-70	0.187	0.168	0.355
71+	0.219	0.185	0.404
Total	0.983	0.933	1.916

Rate Normalized by Row

Age	Female	Male	Total
00-06	0.488	0.512	1.0
07-15	0.478	0.522	1.0
16-18	0.509	0.491	1.0
19-52	0.519	0.481	1.0
53-70	0.527	0.473	1.0
71+	0.542	0.458	1.0
Total	3.063	2.937	6.0



Hospitalization Rate Normalized by Age

More details on feature engineering in the technical paper.

4.5. Handling Data Imbalance

In the Literature Review section we enumerated many methods for dealing with data imbalance, most of which are relevant to our problem. We will try all of the relevant methods, individually and in combinations, to see which combinations of methods give the most effective model.

4.6. Model Building

To build our model, we chose the Keras/Tensorflow (**?**) library over scikit-learn (**?**) because of the ease of writing a custom loss function, although we did use the sklearn library for many data preparation functions.

Since our focus is on levels of data availability and handling data imbalance, we did not build a sophisticated model. We adapted the model from the Tensorflow tutorial on imbalanced data, which uses credit card fraud detection as its application, but works very similarly. (**?**)

Funding Statement

Conflict of Interest

The authors have no relevant financial or non-financial interests to disclose.

Acknowledgements

George Broussard contributed to this work in the NSF Research Experiences for Undergraduates program.

Data Availability

The CRSS data is publicly available at the link in the references. The Louisiana crash data is not publicly available.

Technical Paper

The technical paper with more detail and the code used for the CRSS data can be found at the corresponding author's GitHub page, https://www.github.com/bburkman.

9.

References

Aloul, F., Zualkernan, I., Abu-Salma, R., Al-Ali, H., Al-Merri, M., 2014. ibump: Smartphone application to detect car accidents. 2014 International Conference on Industrial Automation, Information and Communications Technology, Industrial Automation, Information and Communications Technology (IAICT), 2014 International Conference on , 52 – 56URL: https://ezproxyprod.ucs.louisiana.edu:2443/login?url=https://search.ebscohost.com/login.aspx?direct=true&AuthType=ip,cookie,uid,url&db=edseee&AN=edseee.6922107&site=eds-live&scope=site.

Amini, M., Bagheri, A., Delen, D., 2022. Discovering injury severity risk factors in automobile crashes: A hybrid explainable ai framework for decision support. Reliability Engineering & System Safety 226, 108720. URL: https://www.sciencedirect.com/science/article/pii/S0951832022003441, doi:https://doi.org/10.1016/j.ress.2022.108720.

Basso, F., Pezoa, R., Varas, M., Villalobos, M., 2021. A deep learning approach for real-time crash prediction using vehicle-by-vehicle data. Accident Analysis & Prevention 162, 106409. URL: https://www.sciencedirect.com/science/article/pii/S0001457521004401, doi:https://doi.org/10.1016/j.aap.2021.106409.

BlinkApp LLC, . Blink! URL: https://www.blinkapp.net.

Breiman, L., 1996. Machine Learning 24. URL: https://doi.org/10.1007/BF00058655.

Cai, Q., Abdel-Aty, M., Yuan, J., Lee, J., Wu, Y., 2020. Real-time crash prediction on expressways using deep generative models. Transportation Research Part C: Emerging Technologies 117, 102697. URL: https://www.sciencedirect.com/science/article/pii/S0968090X20306124, doi:https://doi.org/10.1016/j.trc.2020.102697.

Chawla, N., Bowyer, K., Hall, L., Kegelmeyer, W., 2002. Smote: Synthetic minority over-sampling technique. JOURNAL OF ARTIFICIAL INTELLIGENCE RESEARCH 16, 321 – 357.

Chen, T., Lu, Y., Fu, X., Sze, N., Ding, H., 2022a. A resampling approach to disaggregate analysis of bus-involved crashes using panel data with excessive zeros. Accident Analysis & Prevention 164, 106496. URL: https://www.sciencedirect.com/science/article/pii/S0001457521005273, doi:https://doi.org/10.1016/j.aap.2021.106496.

Chen, T., Shi, X., Wong, Y.D., Yu, X., 2020. Predicting lane-changing risk level based on vehicles' space-series features: A pre-emptive learning approach. Transportation Research Part C: Emerging Technologies 116, 102646. URL: https://www.sciencedirect.com/science/article/pii/S0968090X20305611, doi:https://doi.org/10.1016/j.trc.2020.102646.

Chen, Z., Liu, K., Wang, J., Yamamoto, T., 2022b. H-convlstm-based bagging learning approach for ride-hailing demand prediction considering imbalance problems and sparse uncertainty. Transportation Research Part C: Emerging Technologies 140, 103709. URL: https://www.sciencedirect.com/science/article/pii/S0968090X22001474, doi:https://doi.org/10.1016/j.trc.2022.103709.

Chollet, F., et al., 2015. Keras. https://keras.io.

Cox, A.E., Cicchino, J.B., 2021. Continued trends in older driver crash involvement rates in the united states: Data through 2017–2018. Journal of Safety Research 77, 288–295. URL: https://www.sciencedirect.com/science/article/pii/S0022437521000463, doi:https://doi.org/10.1016/j.jsr.2021.03.013.

Elamrani Abou Elassad, Z., Mousannif, H., Al Moatassime, H., 2020. A real-time crash prediction fusion framework: An imbalance-aware strategy for collision avoidance systems. Transportation Research Part C: Emerging Technologies 118, 102708. URL: https://www.sciencedirect.com/science/article/pii/S0968090X20306239, doi:https://doi.org/10.1016/j.trc.2020.102708.

Evanco, W.E., 1996. Impact Of Rapid Incident Detection On Freeway Accident Fatalities. Technical Report. Joint Program Office for Intelligent Transportation Systems. URL: https://rosap.ntl.bts.gov/view/dot/14153. 792508; PDF; Research Paper; DTFH61-95-C00040;.

Formosa, N., Quddus, M., Ison, S., Abdel-Aty, M., Yuan, J., 2020. Predicting real-time traffic conflicts using deep learning. Accident Analysis & Prevention 136, 105429. URL: https://www.sciencedirect.com/science/article/pii/S000145751930973X, doi:https://doi.org/10.1016/j.aap.2019.105429.

Freund, Y., Schapire, R.E., 1997. A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences 55, 119–139. URL: https://www.sciencedirect.com/science/article/pii/S002200009791504X, doi:https://doi.org/10.1006/jcss.1997.1504.

Gong, H., Fu, T., Sun, Y., Guo, Z., Cong, L., Hu, W., Ling, Z., 2022. Two-vehicle driver-injury severity: A multivariate random parameters logit approach. Analytic Methods in Accident Research 33, 100190. URL: https://www.sciencedirect.com/science/article/pii/S2213665721000348, doi:https://doi.org/10.1016/j.amar.2021.100190.

Google, . Get help a!er a car crash (pixel 6 pro). URL: https://partnerdash.google.com/apps/simulator/#get-help-after-a-car-crash?l=en&model=Pixel_6_Pro.

Guo, M., Zhao, X., Yao, Y., Yan, P., Su, Y., Bi, C., Wu, D., 2021. A study of freeway crash risk prediction and interpretation based on risky driving behavior and traffic flow data. Accident Analysis & Prevention 160, 106328. URL: https://www.sciencedirect.com/science/article/pii/S0001457521003596, doi:https://doi.org/10.1016/j.aap.2021.106328.

Haule, H.J., Ali, M.S., Alluri, P., Sando, T., 2021. Evaluating the effect of ramp metering on freeway safety using real-time traffic data. Accident Analysis & Prevention 157, 106181. URL: https://www.sciencedirect.com/science/article/pii/S0001457521002128, doi:https://doi.org/10.1016/j.aap.2021.106181.

Herbert, G., 2019. Crash Report Sampling System: Imputation. Technical Report DOT HS 812 795. National Highway Traffic Safety Administration.

Islam, Z., Abdel-Aty, M., Cai, Q., Yuan, J., 2021. Crash data augmentation using variational autoencoder. Accident Analysis & Prevention 151, 105950. URL: https://www.sciencedirect.com/science/article/pii/S000145752031770X, doi:https://doi.org/10.1016/j.aap.2020.105950.

Jiang, F., Yuen, K.K.R., Lee, E.W.M., 2020. A long short-term memory-based framework for crash detection on freeways with traffic data of different temporal resolutions. Accident Analysis & Prevention 141, 105520. URL: https://www.sciencedirect.com/science/article/pii/S0001457519317713, doi:https://doi.org/10.1016/j.aap.2020.105520.

Kaplan, M.S., Caetano, R., Giesbrecht, N., Huguet, N., Kerr, W.C., McFarland, B.H., Nolte, K.B., 2017. The national violent death reporting system: Use of the restricted access database and recommendations for the system's improvement. American Journal of Preventive Medicine 53, 130–133. URL: https://www.sciencedirect.com/science/article/pii/S0749379717301101, doi:https://doi.org/10.1016/j.amepre.2017.01.043.

Khan, W.A., Ma, H.L., Chung, S.H., Wen, X., 2021. Hierarchical integrated machine learning model for predicting flight departure delays and duration in series. Transportation Research Part C: Emerging Technologies 129, 103225. URL: https://www.sciencedirect.com/science/article/pii/S0968090X21002394, doi:https://doi.org/10.1016/j.trc.2021.103225.

King, G., Zeng, L., 2001. Logistic regression in rare events data. Political Analysis 9, 137 – 163.

Lack, C.D., Berkow, K.S., Gao, Y., 2021. Insights into motor carrier crashes: A preliminary investigation of fmcsa inspection violations. Accident Analysis & Prevention 155, 106105. URL: https://www.sciencedirect.com/science/article/pii/S0001457521001366, doi:https://doi.org/10.1016/j.aap.2021.106105.

Li, P., Abdel-Aty, M., Yuan, J., 2020. Real-time crash risk prediction on arterials based on lstm-cnn. Accident Analysis & Prevention 135, 105371. URL: https://www.sciencedirect.com/science/article/pii/S0001457519311108, doi:https://doi.org/10.1016/j.aap.2019.105371.

Li, Y., Li, M., Yuan, J., Lu, J., Abdel-Aty, M., 2021. Analysis and prediction of intersection traffic violations using automated enforcement system data. Accident Analysis & Prevention 162, 106422. URL: https://www.sciencedirect.com/science/article/pii/S000145752100453X, doi:https://doi.org/10.1016/j.aap.2021.106422.

Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P., 2017. Focal loss for dense object detection, in: Proceedings of the IEEE international conference on computer vision, pp. 2980–2988.

Lin, Y., Li, L., Jing, H., Ran, B., Sun, D., 2020. Automated traffic incident detection with a smaller dataset based on generative adversarial networks. Accident Analysis & Prevention 144, 105628. URL: https://www.sciencedirect.com/science/article/pii/S0001457519314150, doi:https://doi.org/10.1016/j.aap.2020.105628.

Liu, Y., Lyu, C., Liu, Z., Cao, J., 2021. Exploring a large-scale multi-modal transportation recommendation system. Transportation Research Part C: Emerging Technologies 126, 103070. URL: https://www.sciencedirect.com/science/article/pii/S0968090X21000954, doi:https://doi.org/10.1016/j.trc.2021.103070.

Man, C.K., Quddus, M., Theofilatos, A., 2022. Transfer learning for spatio-temporal transferability of real-time crash prediction models. Accident Analysis & Prevention 165, 106511. URL: https://www.sciencedirect.com/science/article/pii/S000145752100542X, doi:https://doi.org/10.1016/j.aap.2021.106511.

Mohammadi, R., He, Q., Ghofrani, F., Pathak, A., Aref, A., 2019. Exploring the impact of foot-by-foot track geometry on the occurrence of rail defects. Transportation Research Part C: Emerging Technologies 102, 153–172. URL: https://www.sciencedirect.com/science/article/pii/S0968090X18314001, doi:https://doi.org/10.1016/j.trc.2019.03.004.

Morris, C., Yang, J.J., 2021. Effectiveness of resampling methods in coping with imbalanced crash data: Crash type analysis and predictive modeling. Accident Analysis & Prevention 159, 106240. URL: https://www.sciencedirect.com/science/article/pii/S0001457521002712, doi:https://doi.org/10.1016/j.aap.2021.106240.

Mueller, A.S., Cicchino, J.B., 2022. Teen driver crashes potentially preventable by crash avoidance features and teen-driver-specific safety technologies. Journal of Safety Research 81, 305–312. URL: https://www.sciencedirect.com/science/article/pii/S0022437522000433, doi:https://doi.org/10.1016/j.jsr.2022.03.007.

National Center for Statistics and Analysis, 2022. Crash Report Sampling System analytical user's manual, 2016-2020. Technical Report DOT HS 813 236. National Highway Traffic Safety Administration.

NHTSA, 1975-2020. Fatality analysis reporting system. https://www.nhtsa.gov/research-data/fatality-analysis-reporting-system-fars.

NHTSA, 2016-2020. Crash report sampling system. https://www.nhtsa.gov/crash-data-systems/crash-report-sampling-system.

Orsini, F., Gecchele, G., Rossi, R., Gastaldi, M., 2021. A conflict-based approach for real-time road safety analysis: Comparative evaluation with crash-based models. Accident Analysis & Prevention 161, 106382. URL: https://www.sciencedirect.com/science/article/pii/S0001457521004139, doi:https://doi.org/10.1016/j.aap.2021.106382.

Park, H., Oh, C., 2019. A vehicle speed harmonization strategy for minimizing inter-vehicle crash risks. Accident Analysis & Prevention 128, 230–239. URL: https://www.sciencedirect.com/science/article/pii/S0001457519300314, doi:https://doi.org/10.1016/j.aap.2019.04.014.

Parsa, A.B., Taghipour, H., Derrible, S., Mohammadian, A.K., 2019. Real-time accident detection: Coping with imbalanced data. Accident Analysis & Prevention 129, 202–210. URL: https://www.sciencedirect.com/science/article/pii/S0001457519301642, doi:https://doi.org/10.1016/j.aap.2019.05.014.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, 2825–2830.

Peng, Y., Li, C., Wang, K., Gao, Z., Yu, R., 2020. Examining imbalanced classification algorithms in predicting real-time traffic crash risk. Accident Analysis & Prevention 144, 105610. URL: https://www.sciencedirect.com/science/article/pii/S0001457519306906, doi:https://doi.org/10.1016/j.aap.2020.105610.

Raghunathan, T., Solenberger, P., Berglund, P., van Hoewyk, J., . Iveware: Imputation and variation estimation software. URL: https://www.src.isr.umich.edu/software/iveware/.

Rahman, M., 2019. Google accidentally rolls out "personal safety" app, confirming car crash detection is coming to the pixel. URL: https://www.xda-developers.com/google-pixel-car-crash-detection/.

Schlögl, M., 2020. A multivariate analysis of environmental effects on road accident occurrence using a balanced bagging approach. Accident Analysis & Prevention 136, 105398. URL: https://www.sciencedirect.com/science/article/pii/S0001457519308516, doi:https://doi.org/10.1016/j.aap.2019.105398.

Schlögl, M., Stütz, R., Laaha, G., Melcher, M., 2019. A comparison of statistical learning methods for deriving determining factors of accident occurrence from an imbalanced high resolution dataset. Accident Analysis & Prevention 127, 134–149. URL: https://www.sciencedirect.com/science/article/pii/S0001457518307760, doi:https://doi.org/10.1016/j.aap.2019.02.008.

Shi, Q., Zhang, H., 2021. An improved learning-based lstm approach for lane change intention prediction subject to imbalanced data. Transportation Research Part C: Emerging Technologies 133, 103414. URL: https://www.sciencedirect.com/science/article/pii/S0968090X21004083, doi:https://doi.org/10.1016/j.trc.2021.103414.

Shi, X., Wong, Y.D., Li, M.Z.F., Palanisamy, C., Chai, C., 2019. A feature learning approach based on xgboost for driving assessment and risk prediction. Accident Analysis & Prevention 129, 170–179. URL: https://www.sciencedirect.com/science/article/pii/S0001457518310820, doi:https://doi.org/10.1016/j.aap.2019.05.005.

Sosmart SAP, . SOSmart automatic car crash detection app. URL: http://www.sosmartapp.com.

Spicer, R., Bahouth, G., Vahabaghaie, A., Drayer, R., 2021. Frequency and cost of crashes, fatalities, and injuries involving disabled vehicles. Accident Analysis & Prevention 152, 105974. URL: https://www.sciencedirect.com/science/article/pii/S0001457521000051, doi:https://doi.org/10.1016/j.aap.2021.105974.

Subramanian, R., et al., 2002. Transitioning to multiple imputation: a new method to impute missing blood alcohol concentration (BAC) values in FARS. Technical Report. National Center for Statistics and Analysis (US).

Tensorflow Authors, 2019. Classification on imbalanced data. URL: https://www.tensorflow.org/tutorials/structured_data/imbalanced_data.

Tomek, I., 1976. Two modifications of cnn. IEEE transactions on Systems, Man and Communications, SMC 6, 769–772.

Topuz, K., Delen, D., 2021. A probabilistic bayesian inference model to investigate injury severity in automobile crashes. Decision Support Systems 150, 113557. URL: https://www.sciencedirect.com/science/article/pii/S0167923621000671, doi:https://doi.org/10.1016/j.dss.2021.113557. interpretable Data Science For Decision Making.

Villavicencio, L., Svancara, A.M., Kelley-Baker, T., Tefft, B.C., 2022. Passenger presence and the relative risk of teen driver death. Journal of Adolescent Health 70, 757–762. URL: https://www.sciencedirect.com/science/article/pii/S1054139X21005759, doi:https://doi.org/10.1016/j.jadohealth.2021.10.038.

White, J., Thompson, C., Turner, H., Dougherty, B., Schmidt, D.C., 2011. Wreckwatch: Automatic traffic accident detection and notification with smartphones. MOBILE NETWORKS AND APPLICATIONS 16, 285 – 303. URL: https://ezproxyprod.ucs.louisiana.edu:2443/login?url=https://search.ebscohost.com/login.aspx?direct=true&AuthType=ip,cookie,uid,url&db=edsbl&AN=RN290973892&site=eds-live&scope=site.

Winkler, R., 2021. Apple wants iphones to detect car crashes, auto-dial 911. The Wall Street Journal Eastern Edition , 1URL: https://ezproxyprod.ucs.louisiana.edu:2443/login?url=https://search.ebscohost.com/login.aspx?direct=

true&AuthType=ip,cookie,uid,url&db=edsgao&AN=edsgcl.681029711&site=eds-live&scope=site.

Yahaya, M., Guo, R., Fan, W., Bashir, K., Fan, Y., Xu, S., Jiang, X., 2021a. Bayesian networks for imbalance data to investigate the contributing factors to fatal injury crashes on the ghanaian highways. Accident Analysis & Prevention 150, 105936. URL: https://www.sciencedirect.com/science/article/pii/S0001457520317565, doi:https://doi.org/10.1016/j.aap.2020.105936.

Yahaya, M., Guo, R., Jiang, X., Bashir, K., Matara, C., Xu, S., 2021b. Ensemble-based model selection for imbalanced data to investigate the contributing factors to multiple fatality road crashes in ghana. Accident Analysis & Prevention 151, 105851. URL: https://www.sciencedirect.com/science/article/pii/S0001457520316717, doi:https://doi.org/10.1016/j.aap.2020.105851.

Yu, R., Wang, Y., Zou, Z., Wang, L., 2020. Convolutional neural networks with refined loss functions for the real-time crash risk analysis. Transportation Research Part C: Emerging Technologies 119, 102740. URL: https://www.sciencedirect.com/science/article/pii/S0968090X20306549, doi:https://doi.org/10.1016/j.trc.2020.102740.

Zhu, S., Wan, J., 2021. Cost-sensitive learning for semi-supervised hit-and-run analysis. Accident Analysis & Prevention 158, 106199. URL: https://www.sciencedirect.com/science/article/pii/S000145752100230X, doi:https://doi.org/10.1016/j.aap.2021.106199.