Highlights

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

First Author,Second Author,Third Author,Fourth Author

- 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

First Author[a,b], Second Author[a], Third Author[a,c] and Fourth Author[c]

[a] *School, University,*
[b] *Other School,*
[c] *Other Department, University,*

ARTICLE INFO

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

ABSTRACT

New Google Pixel phones can automatically notify police if the phone detects the deceleration profile of a crash. From the data available from such an automatic notification, can we build a machine-learning model that will recommend whether police should immediately, perhaps automatically, dispatch an ambulance? If the injuries are serious, time to medical care is critical, but few crashes result in serious injuries, and ambulances are in limited supply and expensive. Such a model will not be perfect, with many false positives (sending an ambulance when one is not needed) and some false negatives (not sending an ambulance when one is needed), but better than random. How much better depends on several things that we will investigate.

A key idea underlying this analysis is that the costs of the false positives and false negatives are very different. The cost of sending an ambulance when one is not needed is measured in dollars, but the cost of not sending an ambulance when one is needed is measured in lives. We propose a way to interpret class weights as the ethical rate of tradeoff.

We will show that the quality of the model depends mostly on what information is available to inform the decision of whether to immediately dispatch an ambulance. Whether a model is "good" is partly a political question, weighing prompt medical care against its high cost, but given a parameter p of how to weigh those costs, we can build that tradeoff into the model.

We used the data of the Crash Report Sampling System (CRSS).This data is freely available online. We have applied new methods (for this dataset in the literature) to handle missing data, and we have investigated several methods for handling the data imbalance. To promote discussion and future research, we have included all of the code we used in our analysis.

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 are very rare, about seventy-seven fatalities annually in the US in 2010-2018, (Spicer, Bahouth, Vahabaghaie and Drayer, 2021) of the about 35,000 crash fatalities per year in the same time period. (NHTSA, 1975-2020)

A much larger group who could benefit from faster ambulance response 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 (Evanco, 1996), 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.

`FirstAuthor@gmail.com` (F. Author)
ORCID(s):

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.

[Insert cost differential discussion]

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.* 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.

Feature Selection 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?

Feature Engineering 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?

Binning Some features have many values that we can usefully combine into bins or bands. 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.

Missing Data As with all real data, many samples (records) have missing values. The CRSS authors imputed missing values in some but not all features, for historical reasons going back to 1982. (Herbert, 2019) We compared their method with two others and imputed missing values in all of the features we used.

Imbalanced Data Only a small proportion of crashes require immediate medical attention. In the CRSS data, about 15% 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 85% 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).

1.4. 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.

1.5. Novel Aspects of this Work

Application We applied existing methods to an emerging application, automated cell phone crash reports.

Imputing Missing Data Other authors have imputed missing values in the CRSS data set, but as far as we know, we are the first to try the method the CRSS authors used (IVEware).

Cost-Sensitive Analysis This method has appeared in the crash literature, but we made it central to our analysis.

Open Science To promote transferability, we have attached all of the code we used.

2. 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 (NHTSA, 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.

2.1. 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 (National Center for Statistics and Analysis, 2022). 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.

2.2. 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.

2.3. 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" (Raghunathan, Solenberger, Berglund and van Hoewyk).

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 (Herbert, 2019). 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.

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

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

- Topuz and Delen (2021) 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.

- Cox and Cicchino (2021) 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."

- Amini, Bagheri and Delen (2022) 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.

- Spicer et al. (2021) used CRSS but did not mention missing or imputed data.

- Villavicencio, Svancara, Kelley-Baker and Tefft (2022) 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."

- Mueller and Cicchino (2022) 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."

- Kaplan, Caetano, Giesbrecht, Huguet, Kerr, McFarland and Nolte (2017) uses the phrase, "restricted access database." I should use that for the Louisiana crash database.

- Gong, Fu, Sun, Guo, Cong, Hu and Ling (2022) just dropped samples with missing values.

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

3. Methods

We used the Crash Report Sampling System (CRSS) NHTSA (2016-2020) data to train and test our model. After preparing the data, we built a variety of models with different combinations of techniques for handling imbalanced data, model types, loss functions, and sets of features, with the overall goal of finding an optimal combination of methods to give the most useful model.

3.1. Model Building

To build our model, we chose the Keras/Tensorflow (Chollet et al., 2015) library over scikit-learn (Pedregosa, Varoquaux, Gramfort, Michel, Thirion, Grisel, Blondel, Prettenhofer, Weiss, Dubourg, Vanderplas, Passos, Cournapeau, Brucher, Perrot and Duchesnay, 2011) 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. (Tensorflow Authors, 2019)

3.2. Class Weights as Ethical Tradeoff Rate

Deciding how to trade off dollars and lives is not a technical decision, but a political and ethical one. Given a tradeoff rate, $FP/TP < r$, we can incorporate it into our model. **For our study we will choose $r = 2$ and optimize our models based on that rate of ethical tradeoff.** We have no basis for recommending that arbitrary choice tradeoff rate for actual applications, but we decided to choose something. A very small rate, even $r = 1$ may be ethically justifiable, since our model is recommending whether to send an ambulance *now*, without waiting for more information from eyewitnesses, and police can reassess when they have more information.

We can use the class weight hyperparameter to incorporate into the model our value judgement about, at the margin, how many ambulances we are willing to send on a wild goose chase in order to send one that is needed, which is a judgement about how many dollars a life is worth.

Note we're using r for two things

Many loss functions incorporate a class weight hyperparameter, here given by α. One of its uses is to accommodate class imbalance, described above, where we let $1/r$ be the proportion of samples in the minority class.

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

The loss function for each sample is given by L, and the total loss is the sum of those sample losses, J.

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

$$J(y,p,\alpha) = -\sum_{i=1}^{N} \left(\alpha y_i \log\left(p_i\right) + (1-\alpha)\left(1-y_i\right)\log\left(1-p_i\right)\right)$$

Let us recall the confusion matrix, in terms of y_i and p_i. We will use it to switch between binary and continuous versions of the loss functions.

	Do Not Send Ambulance $p_i \leq 0.5$	Send Ambulance $p_i > 0.5$
Ambulance Not Needed $y_i = 0$	TN	FP
Ambulance Needed $y_i = 1$	FN	TP

In the (unweighted) binary cross-entropy loss function,

$$J = -\sum_{i=1}^{N} y_i \log(p_i) + (1-y_i)\log(1-p_i)$$

the y_i are binary, $y_i \in \{0,1\}$, but the model predictions, p_i, are a probability, $p_i \in (0,1)$.

If we treat the model predictions as binary, replacing

$$\log(p_i) \to \begin{cases} 0 & \text{if } p_i \leq 0.5 \\ 1 & \text{if } p_i > 0.5 \end{cases} \qquad \text{and} \qquad \log(1-p_i) \to \begin{cases} 0 & \text{if } 1-p_i \leq 0.5 \\ 1 & \text{if } 1-p_i > 0.5 \end{cases}$$

then

$$TP = \sum_{i=1}^{N} y_i \log(p_i) \qquad \text{and} \qquad TN = \sum_{i=1}^{N}(1-y_i)\log(1-p_i)$$

and the loss function becomes $J = -(TP + TN)$.

We use the continuous version when building the model, because we want the predictions to be robust, so that when we use the model on unseen data we can be more certain that it will correctly classify new instances. We use the binary when we evaluate the model on unseen data, because then we only care about whether the model gets the classification right or wrong. The binary is also easier to explain.

If the medical ethicists and politicians decide on a tradeoff threshold, r such that, at the margin, we are willing to automatically dispatch r ambulances when they aren't needed in order to send one ambulance when it is needed, then we want

$$\frac{\Delta FP}{\Delta TP} \leq r$$

which makes the binary version of our loss function $FP - r \cdot TP$, and the continuous version equivalent to the α-weighted cross-entropy loss function.

$$J = -\sum_{i=1}^{N} \alpha y_i \log(p_i) + (1-\alpha)(1-y_i)\log(1-p_i), \quad \alpha = \frac{r}{r+1}$$

Why are these equivalent?

Adding a constant to the loss function, or multiplying it by a positive constant, does not change its effect, because in comparing one iteration of the model to another, the algorithm is only concerned with which has the smaller loss.

The binary loss function $FP - r \cdot TP$ is equivalent to $FP - r \cdot TP - (TN + FP)$, because $TN + FP$ is the number of negative samples in the dataset (thus constant), so as a loss function, $FP - r \cdot TP$ is equivalent to $-(r \cdot TP + TP)$.

$$FP - r \cdot TP$$
$$-(r \cdot TP + TN)$$

Multiplying by $\frac{1}{r+1}$ gives an equivalent loss function, because $\frac{1}{r+1} > 0$.

$$-\frac{r \cdot TP + TN}{r+1}$$

$$-\left(\frac{r}{r+1}TP + \frac{1}{r+1}TN\right)$$

$$-\left(\frac{r}{r+1}TP + \left(1 - \frac{r}{r+1}\right)TN\right)$$

$$-(\alpha TP + (1-\alpha)TN)$$

The continuous versions of TP and TN are $\sum_{i=1}^{N} y_i \log(p_i)$ and $\sum_{i=1}^{N}(1-y_i)\log(1-p_i)$, so we get the α-weighted binary cross-entropy loss function,

$$J = -\sum_{i=1}^{N} \alpha y_i \log(p_i) + (1-\alpha)(1-y_i)\log(1-p_i), \quad \alpha = \frac{r}{r+1}$$

3.3. ROC Slope and Ethical Tradeoff Rate

The ROC (Receiver Operating Characteristic) curve is a parameterized curve that shows, for values of $p \in (0,1)$, the values for the True Positive Rate (TPR) versus the False Positive Rate (FPR). The ethical tradeoff rate above is inversely proportional to the slope of the ROC curve that has been used in other literature for cost-sensitive analysis [CITATION]

$$FPR = \frac{FP}{N} \rightarrow FP = N \cdot FPR$$

$$TPR = \frac{TP}{P} \rightarrow TP = P \cdot TPR$$

$$\frac{FP}{TP} = \frac{N \cdot FPR}{P \cdot TPR} = \frac{N}{P} \cdot \frac{FPR}{TPR}$$

and because N and P are constant,

$$\frac{\Delta FP}{\Delta TP} = \frac{N}{P} \cdot \frac{\Delta FPR}{\Delta TPR}$$

$$\frac{\Delta FP}{\Delta TP} = \frac{N}{P} \cdot \frac{1}{\text{slope of ROC}}$$

$$\frac{\Delta FP}{\Delta TP} = \frac{1}{\frac{P}{N} \cdot (\text{slope of ROC})}$$

The P/N is the proportion of positive to negative class used to balance the classes in the balanced accuracy metric, so our ethical tradeoff rate is the reciprocal of the product of the class balancing ratio and the slope of the ROC curve.

3.4. Model Evaluation: Baselines for Comparison

What do good results look like, what do bad results look like, how do we measure it, and when we compare two results, how much of the difference could be due to randomness?

In the supervised learning method we used here, for each of the $\approx 600,000$ samples (people) in the dataset, we know the answer (the *label* or *ground truth*) to the question, whether the person needed an ambulance, $y = 0$ for "no" and $y = 1$ for "yes." We are trying use historical data to build a model to predict the label for new data (incoming automated crash notifications).

Explain this better. I confused the internal workings of the model building with the results.

We split the data 70/30 into a training set and a test set, making sure to keep the same proportion of positive and negative samples in both. The binary classification models we used take the training data and training labels (`X_train` and `y_train`) and build a model, then apply the model to the test data (`X_test`) and returns `y_proba` that gives, for each sample, a continuous probability $p \in (0, 1)$ that the sample belongs in the positive class. If a sample has $p = 0.1$, the model is 90% confident that this sample is in the negative class. We then pick a threshold, usually but not necessarily $threshold = 0.5$, and make a binary prediction, that samples with $p > threshold$ need an ambulance, and those with $p < threshold$ do not.

While building the model, the algorithm picks a starting point, measures how badly the model predicts the training data using the *loss function*, tweaks the model, measures again, and either keeps or rejects the candidate model based on the loss function. The loss function used by the model is the sum not of how many binary predictions were incorrect, but how strongly incorrect the continuous predictions were. If two negative samples ($y = 0$) had $p = 0.1$ and $p = 0.4$, both correct classifications if $threshold = 0.5$, then the $p = 0.4$ sample would add much more to the loss value.

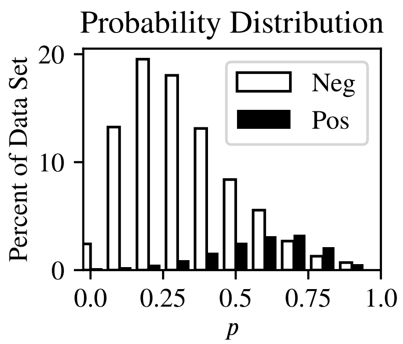
A perfect model would not only predict each sample's label correctly, but would do it with perfect certainty. In the real world, with interesting questions about real data, we will have false positives ($y = 0$ and $p > threshold$) and false negatives ($y = 1$ and $p < threshold$), but we hope those are few, and that the predictions are strongly correct, meaning the predictions are close to their labels.

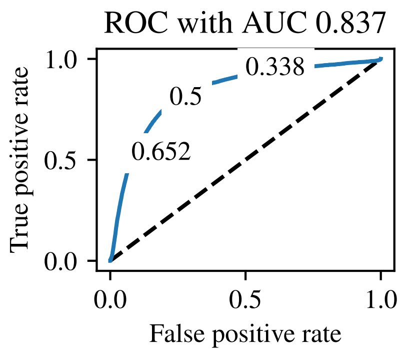
When we get results for our models based on crash data, we need some frame of reference for what is "good" and "bad," so we have created some sets of entirely artificial results using a gamma distribution for ideal results and a uniform distribution for awful results.

The histogram below of the percent of samples with predictions p in each range illustrates the best results we can hope for in the real world. The positive class is small because the data is imbalanced. about 15% of the dataset, as in our CRSS data. There are some false positives and negatives, but the overwhelming majority of the predictions are correct, and most with strong confidence.

The Receiver Operating Characteristic (ROC) is a parameterized curve following the probability threshold from $p = 0$ to $p = 1$, plotting the true positive rate (TPR) versus the false positive rate (FPR). The Area Under the ROC curve (AUC) is often used to compare two models, with AUC of 1 indicating perfect prediction and AUC of 0.5 indicating no discernable pattern.

We have added to the typical ROC curve labels for the medians of the probabilities in the negative and positive classes (0.338 and 0.655) and the default decision threshold $thr = 0.5$.





Incorporate $thr = 0.5$ into the discussion.

The *confusion matrix* for this ideal data set, here given as percentages of the entire dataset, shows few false positives and false negatives. The metrics below are the ones we will watch when evaluating models. Each of them tells a different story about what the model does well.

$Precision = \frac{TP}{PP} = \frac{TP}{TP + FP}$ tells what proportion of the ambulances we sent were needed.

$Recall = \frac{TP}{P} = \frac{TP}{TP + FN}$ tells what proportion of ambulances we needed were sent.

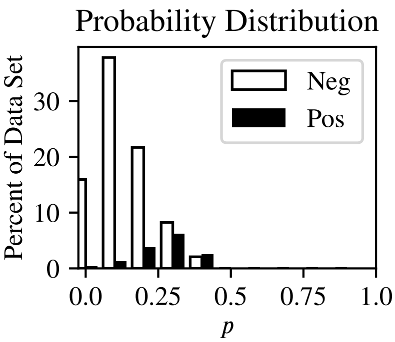
$F1 = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}}$ is the harmonic mean of precision and recall.

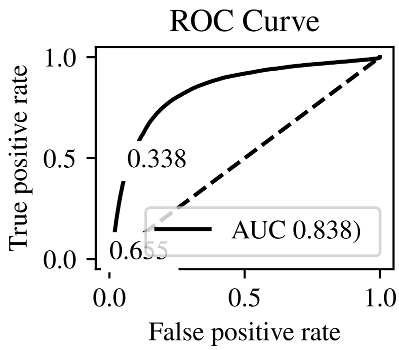
Why the harmonic mean, not the arithmetic or geometric mean? If a and b are positive numbers with $a < b$, then $a <$ harmonic $<$ geometric $<$ arithmetic $< b$. The harmonic, while being influenced by the larger number, is closest to the smaller, so the harmonic mean emphasizes what the model does poorly.

		Prediction Neg	Prediction Pos	Total
Actual	Neg	TN = 67.0%	FP = 18.7%	N = 85.7%
	Pos	FN = 3.2%	TP = 11.1%	P = 14.3%
	Total	PN = 70.2%	PP = 29.8%	

0.372	Precision
0.774	Recall
0.502	F1

If we do not address the data imbalance, the model building algorithm will maximize accuracy by classifying most (or all) of the samples as "No Ambulance" with $p < 0.5$ We built the artificial results below by multiplying the probabilities in the above results by 0.5. Note that the Area Under the Curve (AUC) did not change.

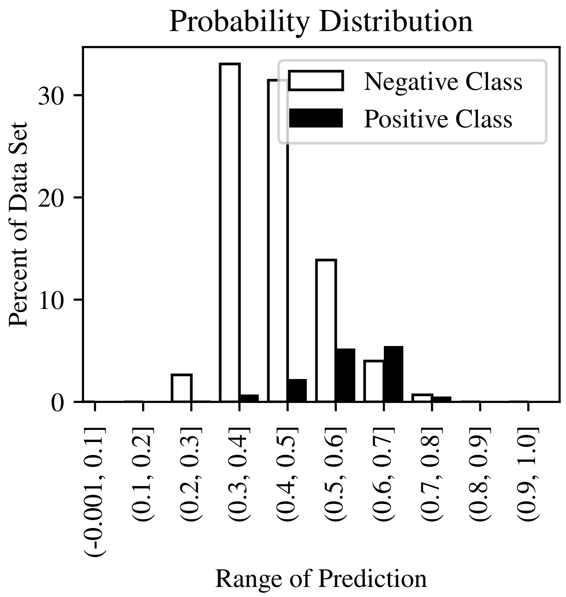


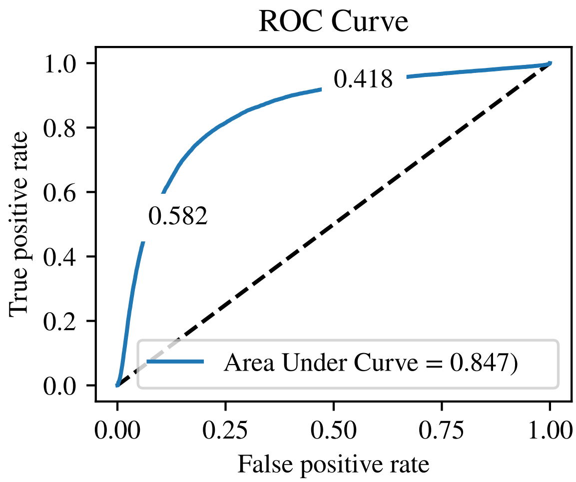


		Prediction N	Prediction P
Actual	N	85.7%	0.0%
	P	14.3%	0.0%

0.857	Accuracy
0.500	Balanced Accuracy
0.000	Precision
0.000	Balanced Precision
0.000	Recall
0.000	F1
0.000	Balanced F1
0.000	Gmean

Such a recommendation system ("Never send an ambulance") would be useless, but note that the distribution still separates the negative and positive classes, just not at $p = 0.5$. We can fix that in two ways; the first is to shift the distribution to be centered at $p = 0.5$. By "centered," we mean that the average of the medians of the negative and positive classes (the 0.107 and 0.293 on the ROC curve above) will now be 0.5. Further research can explore whether centering the distribution at the $p = 0.5$ threshold or another value of p is most useful.

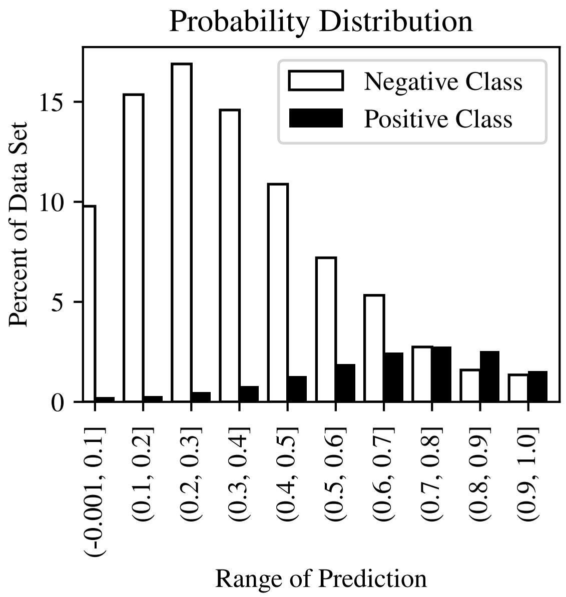


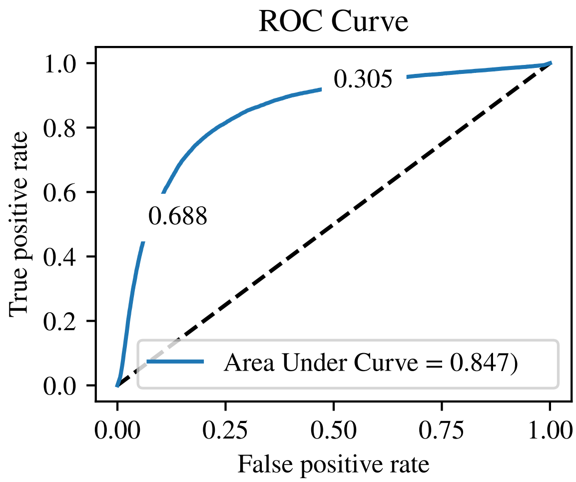


		Prediction	
		N	P
Actual	N	67.2%	18.5%
	P	3.06%	11.22%

0.784	Accuracy
0.785	Balanced Accuracy
0.377	Precision
0.784	Balanced Precision
0.786	Recall
0.510	F1
0.785	Balanced F1
0.543	Gmean

Another way is to linearly transform the probabilities. Whether the distribution was clustered to the left or right, or clustered at the center, is not necessarily relevant, so we want to see it spread out. We have arbitrarily chosen a transformation to put next the original models in our results to see if it will make a better model; tuning the transformation is an avenue for future work. We have chosen to take the 0.05 quantile of the negative class and map it to $p = 0.05$, and the 0.95 quantile of the positive class and map it to $p = 0.95$. This linear transformation gives the same metrics as the shift, and the ROC curve is the same except for the two labeled medians, now at 0.305 and 0.688.

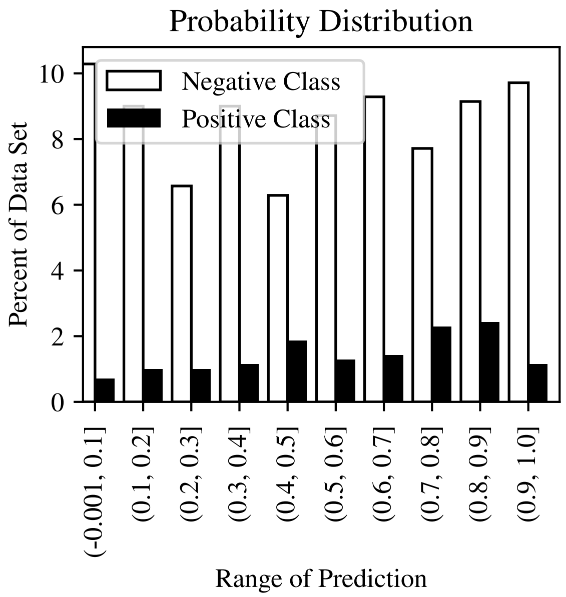


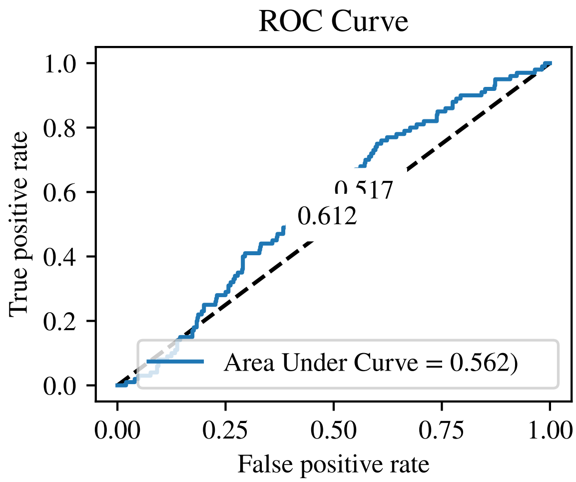


		Prediction N	Prediction P
Actual	N	67.5%	18.2%
	P	3.11%	11.2%

0.787	Accuracy
0.785	Balanced Accuracy
0.380	Precision
0.786	Balanced Precision
0.782	Recall
0.512	F1
0.784	Balanced F1
0.547	Gmean

In the ideal results above, the algorithm learned a useful model from the patterns in the data. The results below illustrate the worst case scenario, where the algorithm does not learn a good model, usually because the data does not have a pattern that predicts the target variable. In the ROC curve, the median values of the probabilities for the two classes are so close that the labels are on top of each other.





		Prediction N	Prediction P
Actual	N	41.1%	44.6%
	P	5.71%	8.57%

0.497	Accuracy
0.540	Balanced Accuracy
0.161	Precision
0.536	Balanced Precision
0.600	Recall
0.254	F1
0.566	Balanced F1
0.278	Gmean

3.5. Model Evaluation: Incorporating Ethical Tradeoffs

In evaluating our models, the considerations are not just technical. A false positive is a recommendation to send an ambulance when one is not needed, which costs money. A false negative is a recommendation to not send an ambulance when one is needed, which costs lives.

Of the metrics we are watching, F1 and AUC seem most useful in discriminating “better” from “worse” models. F1 is the harmonic mean of Precision (What proportion of the ambulances we sent were needed?) and Recall (What proportion of the ambulances needed did we send?), and reflects the discrete results. AUC (Area Under the (ROC) Curve) quantifies the predictive power of the continuous results. It is possible to introduce a parameter to weight the precision and recall parts of F1 if you can quantify how much more important one is than the other, but we had no way to choose that number.

4. Results

We tested several combinations of model inputs.

- Easy, Medium, and Hard to collect features
- Undersampling with Tomek links
- Model types
- Loss Functions
- Class weights parameters and other hyperparameters

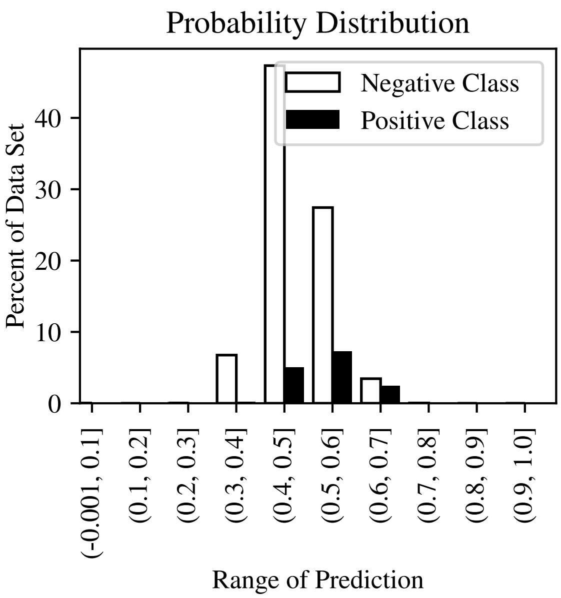
We do not hope to get a perfect model, but something better than, “Never send an ambulance,” “Always send an ambulance,” or random noise.

Each of the five models types we tested had one version in the top five, measured by both F1 and AUC.

F1	**AUC**	**Model**
0.429	0.760	Bagging
0.400	0.752	AdaBoost (Linear transformation)
0.364	0.714	Balanced Random Forest
0.354	0.697	α-weighted Binary Crossentropy with Class Weights
0.353	0.695	Binary Focal Crossentropy with Class Weights and $\gamma = 2.0$ (Linear Transformation)

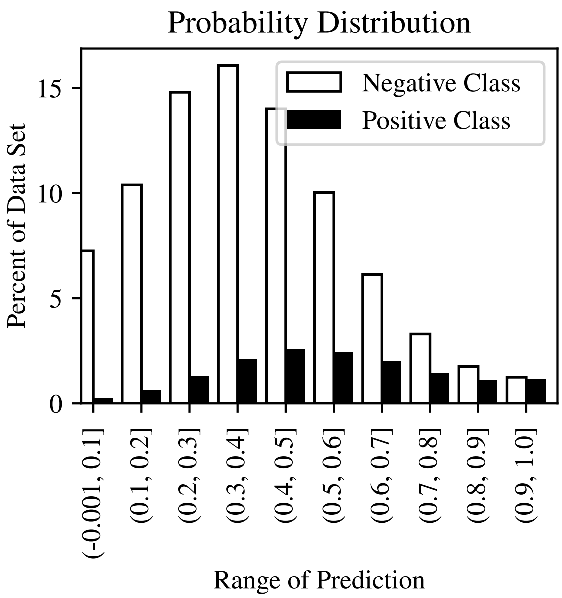
Details follow.

4.1. α-weighted Binary Cross Entropy Model with $\alpha = 0.850$, $r = 5.66$



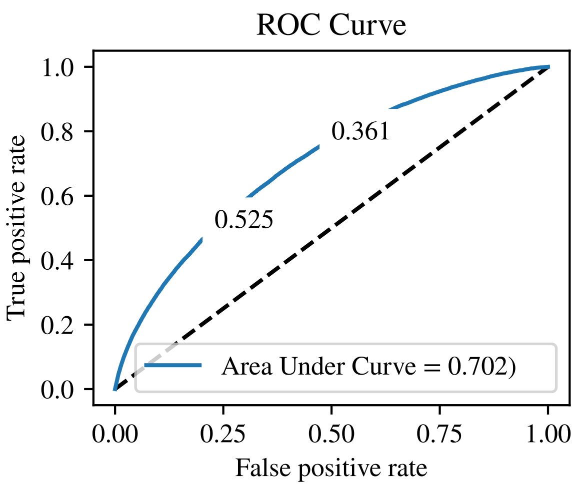
Prediction		N	P
Actual	N	109,166	41,605
	P	11,959	14,662

0.698	Accuracy
0.637	Balanced Accuracy
0.261	Precision
0.666	Balanced Precision
0.551	Recall
0.354	F1
0.603	Balanced F1
0.434	Gmean

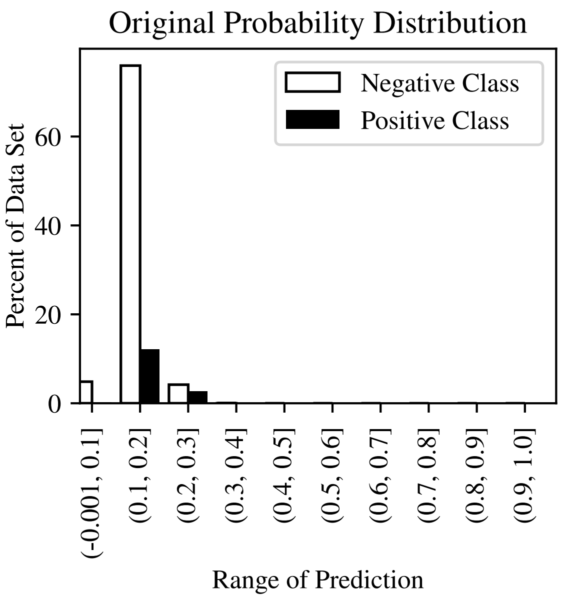


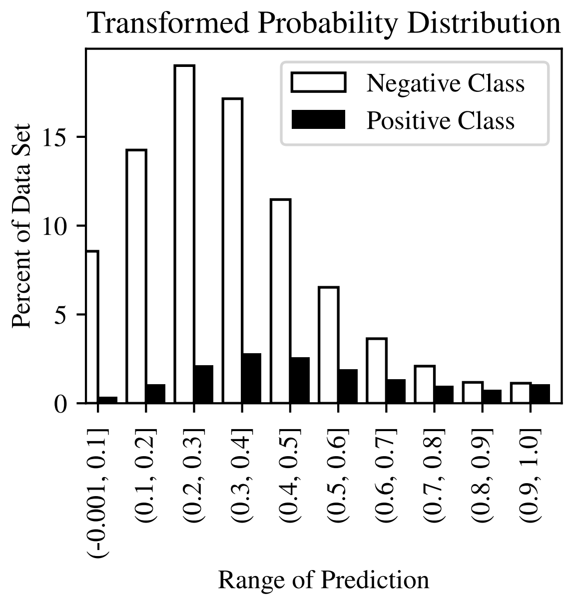
Prediction		N	P
Actual	N	112,677	38,094
	P	12,746	13,875

0.713	Accuracy
0.634	Balanced Accuracy
0.267	Precision
0.674	Balanced Precision
0.521	Recall
0.353	F1
0.588	Balanced F1
0.447	Gmean



4.2. α-weighted Binary Cross Entropy Model with $\alpha = 0.5$, $r = 1$



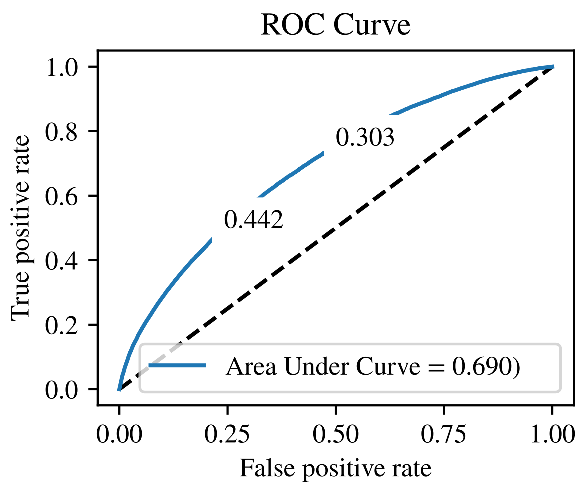


Prediction		N	P
Actual	N	150,771	0
	P	26,621	0

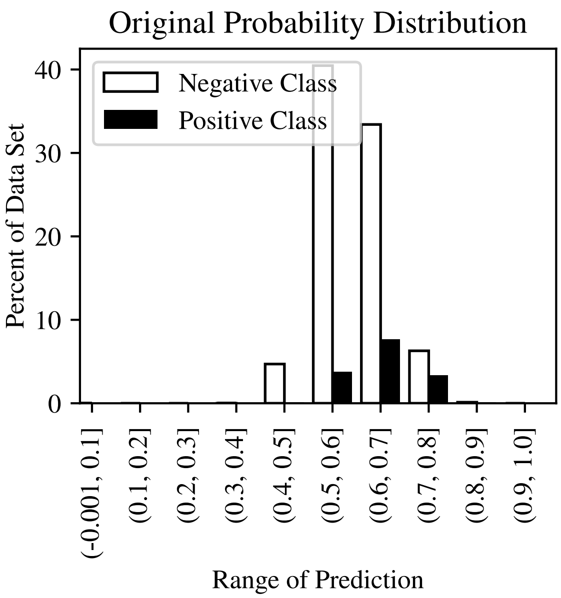
Prediction		N	P
Actual	N	124,949	25,822
	P	15,894	10,727

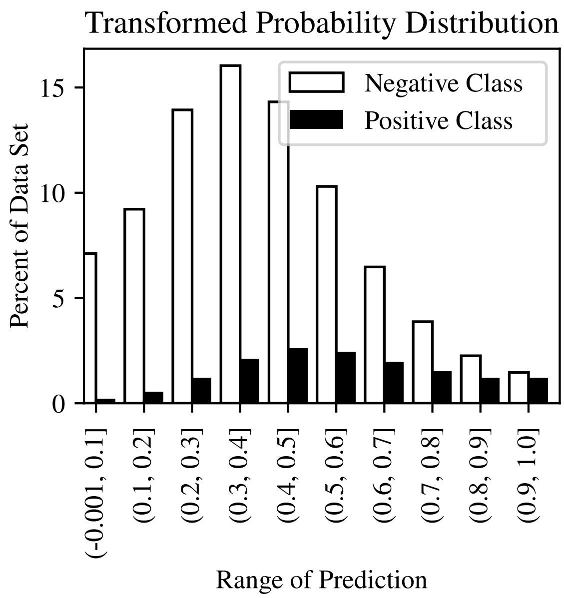
0.850	Accuracy
0.500	Balanced Accuracy
0.000	Precision
0.000	Balanced Precision
0.000	Recall
0.000	F1
0.000	Balanced F1
0.000	Gmean

0.765	Accuracy
0.616	Balanced Accuracy
0.293	Precision
0.702	Balanced Precision
0.403	Recall
0.340	F1
0.512	Balanced F1
0.493	Gmean



4.3. α-weighted Binary Cross Entropy Model with $\alpha = 0.89$, $r = 10$



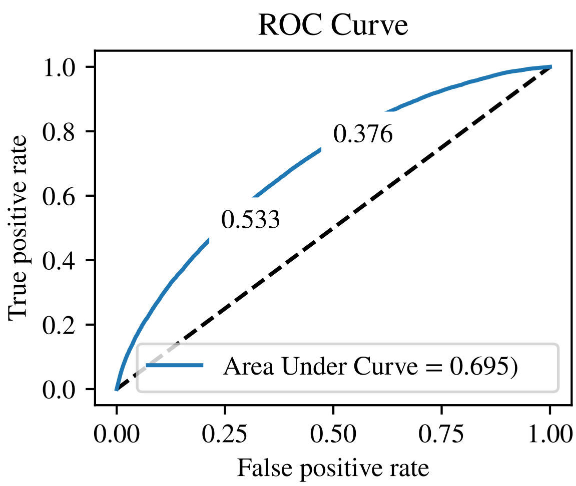


Prediction		N	P
Actual	N	8,346	142,425
	P	210	26,411

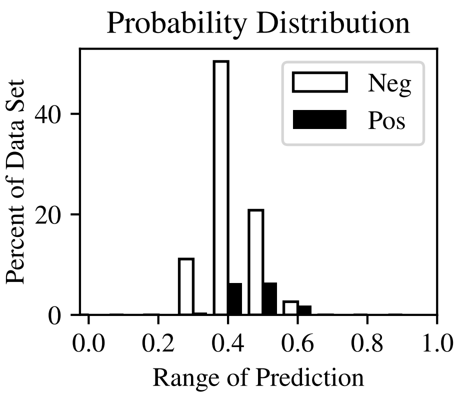
0.196	Accuracy
0.524	Balanced Accuracy
0.156	Precision
0.512	Balanced Precision
0.992	Recall
0.270	F1
0.676	Balanced F1
0.093	Gmean

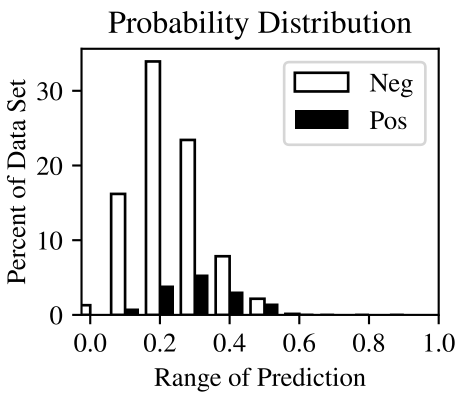
Prediction		N	P
Actual	N	107,543	43,228
	P	11,819	14,802

0.690	Accuracy
0.635	Balanced Accuracy
0.255	Precision
0.660	Balanced Precision
0.556	Recall
0.350	F1
0.603	Balanced F1
0.427	Gmean



4.4. Binary Focal Crossentropy with $\alpha = 0.850$ and $\gamma = 0.0$





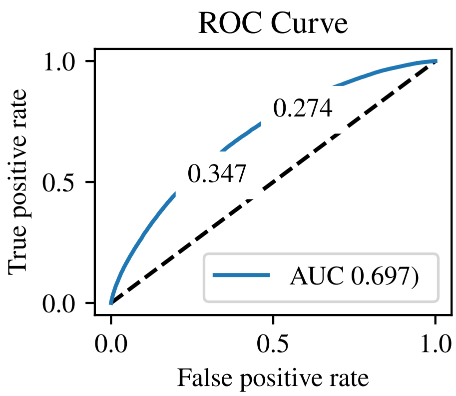
Prediction		N	P
Actual	N	150,771	0
	P	26,621	0

0.850	Accuracy
0.500	Balanced Accuracy
0.000	Precision
0.000	Balanced Precision
0.000	Recall
0.000	F1
0.000	Balanced F1
0.000	Gmean

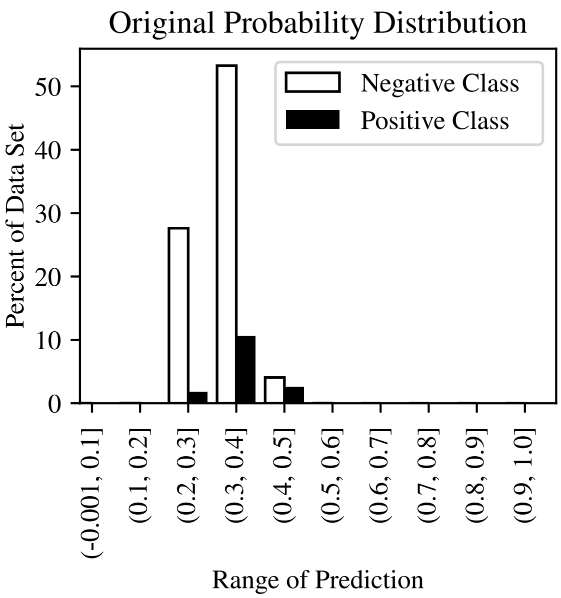
Prediction		N	P
Actual	N	126,475	24,296
	P	16,054	10,567

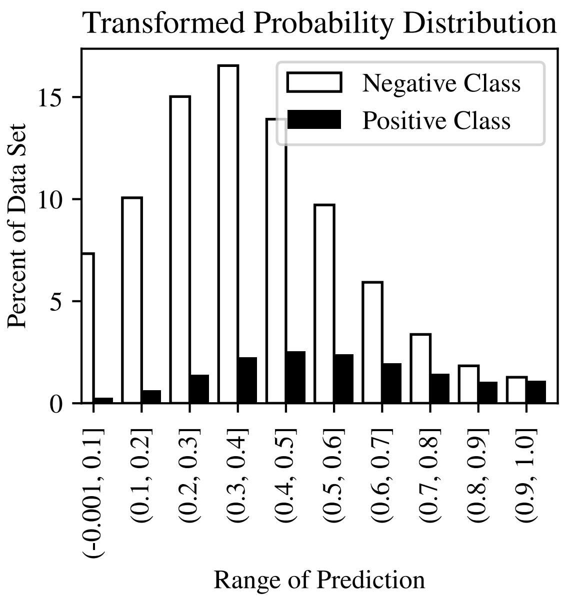
0.773	Accuracy
0.618	Balanced Accuracy
0.303	Precision
0.711	Balanced Precision
0.397	Recall
0.344	F1
0.510	Balanced F1
0.504	Gmean

The results from this model should be the same as for our original α-weighted binary crossentropy model with $\alpha = 0.850$, but they're not. Need to fix that.



4.5. Binary Focal Crossentropy with $\alpha = 0.850$ and $\gamma = 2.0$



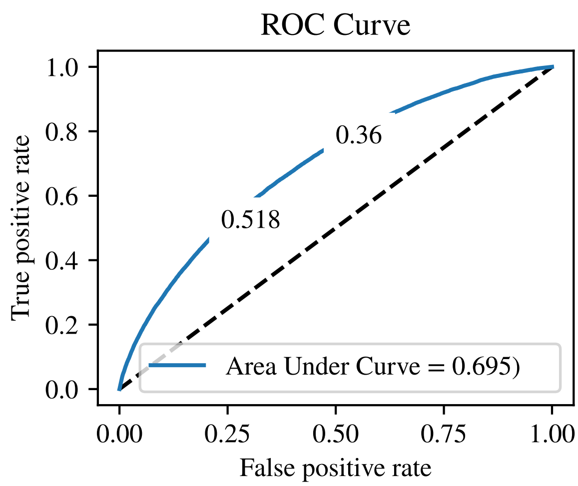


Prediction		N	P
Actual	N	150,771	0
	P	26,621	0

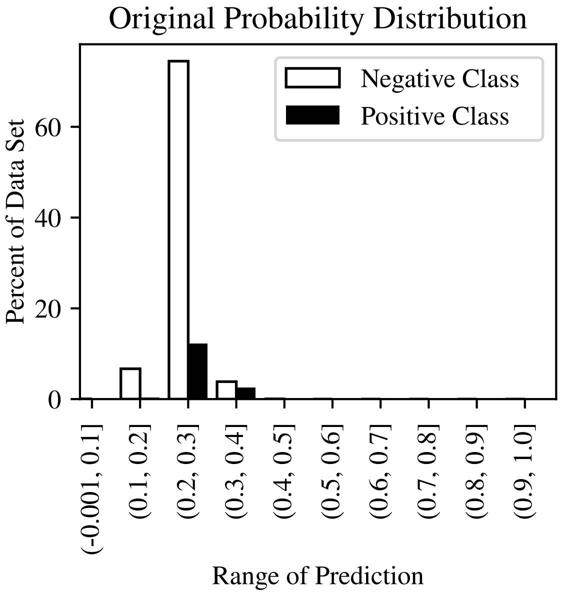
Prediction		N	P
Actual	N	111,533	39,238
	P	12,521	14,100

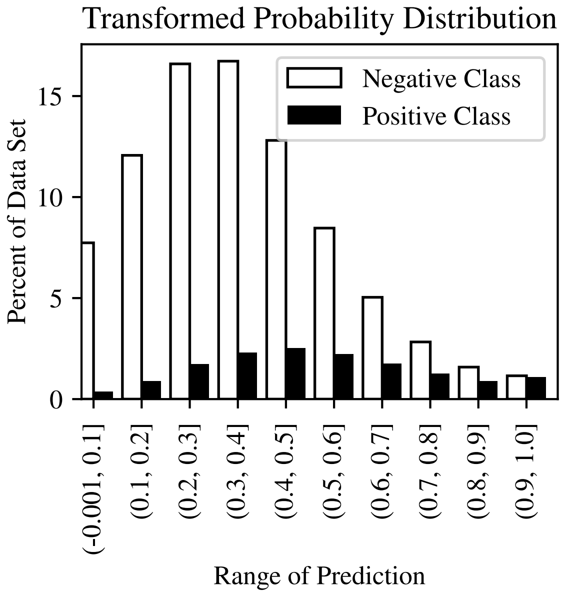
0.850	Accuracy
0.500	Balanced Accuracy
0.000	Precision
0.000	Balanced Precision
0.000	Recall
0.000	F1
0.000	Balanced F1
0.000	Gmean

0.708	Accuracy
0.635	Balanced Accuracy
0.264	Precision
0.671	Balanced Precision
0.530	Recall
0.353	F1
0.592	Balanced F1
0.442	Gmean



4.6. Binary Focal Crossentropy with Class Balancing and $\gamma = 2.0$





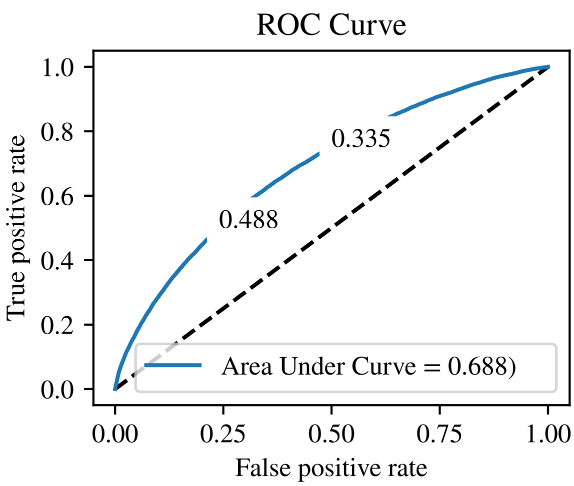
Prediction		N	P
Actual	N	150,771	0
	P	26,621	0

Prediction		N	P
Actual	N	116,943	33,828
	P	13,850	12,771

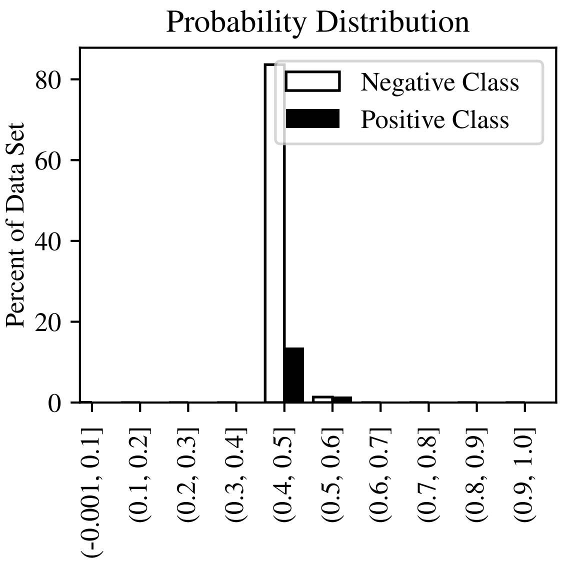
0.850	Accuracy
0.500	Balanced Accuracy
0.000	Precision
0.000	Balanced Precision
0.000	Recall
0.000	F1
0.000	Balanced F1
0.000	Gmean

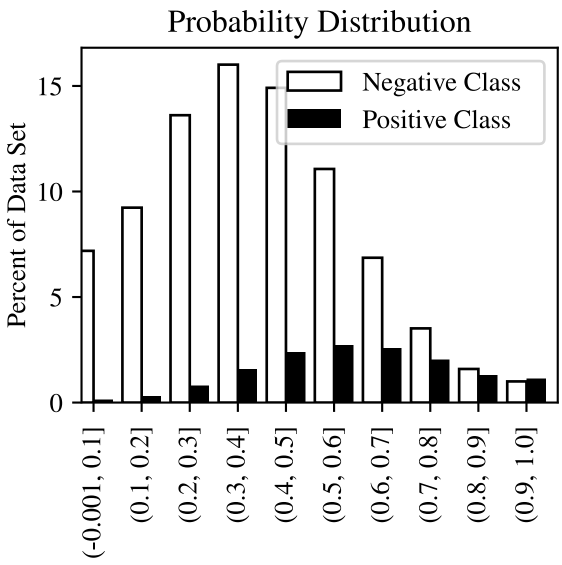
0.731	Accuracy
0.628	Balanced Accuracy
0.274	Precision
0.681	Balanced Precision
0.480	Recall
0.349	F1
0.563	Balanced F1
0.461	Gmean

For this model we took out the α parameter and set `apply_class_balancing=True`.



4.7. AdaBoost



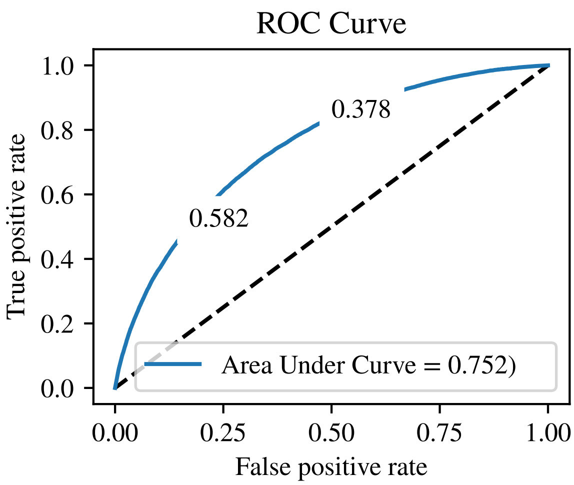


Prediction		N	P
Actual	N	148,358	2,413
	P	23,995	2,626

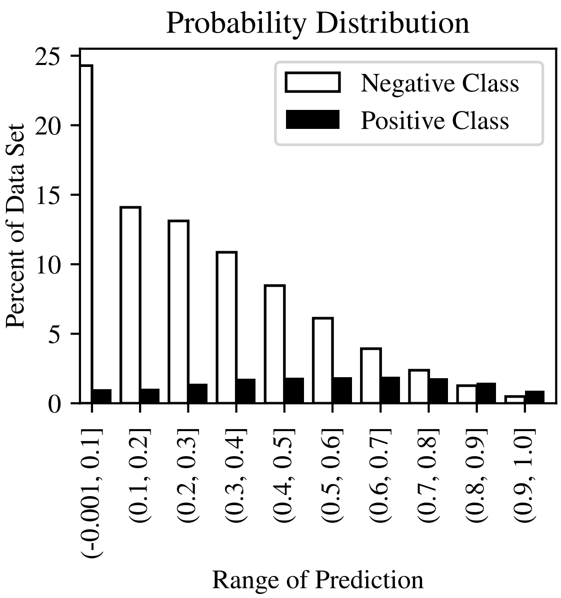
Prediction		N	P
Actual	N	108,133	42,638
	P	9,298	17,323

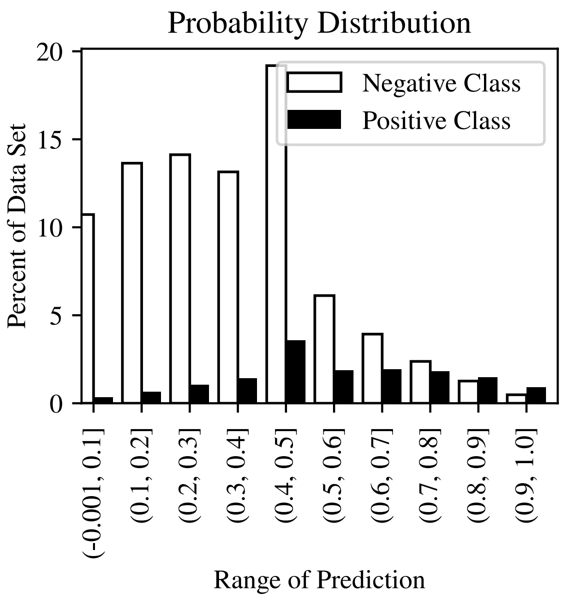
0.851	Accuracy
0.541	Balanced Accuracy
0.521	Precision
0.860	Balanced Precision
0.099	Recall
0.166	F1
0.177	Balanced F1
0.716	Gmean

0.707	Accuracy
0.684	Balanced Accuracy
0.289	Precision
0.697	Balanced Precision
0.651	Recall
0.400	F1
0.673	Balanced F1
0.455	Gmean



4.8. Bagging





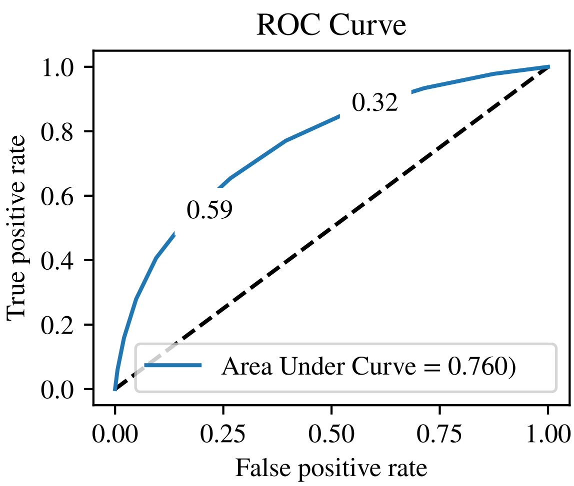
Prediction		N	P
Actual	N	125,633	25,138
	P	12,475	14,146

0.788	Accuracy
0.682	Balanced Accuracy
0.360	Precision
0.761	Balanced Precision
0.531	Recall
0.429	F1
0.626	Balanced F1
0.548	Gmean

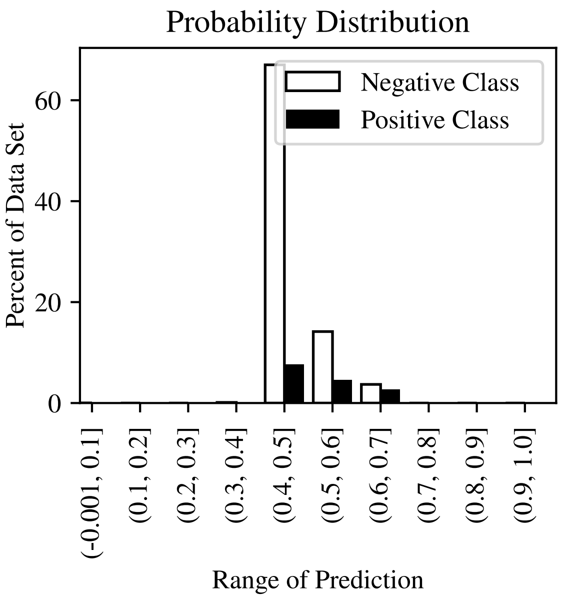
Prediction		N	P
Actual	N	125,633	25,138
	P	12,475	14,146

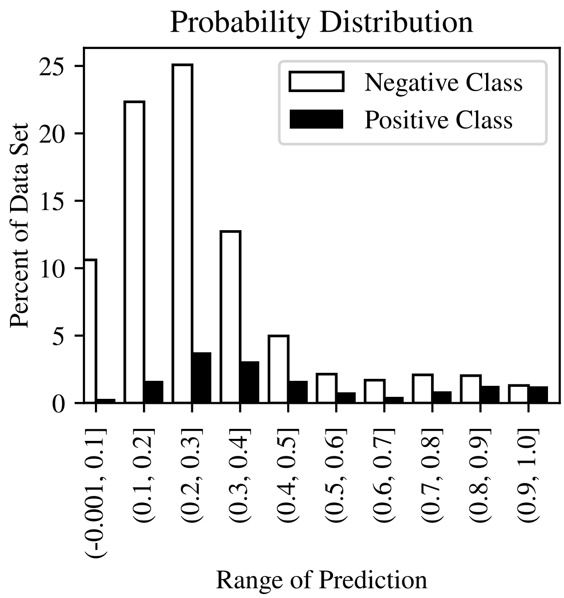
0.788	Accuracy
0.682	Balanced Accuracy
0.360	Precision
0.761	Balanced Precision
0.531	Recall
0.429	F1
0.626	Balanced F1
0.548	Gmean

Our linear transformation took $p = 0.5$ to itself, so the discrete metrics did not change.



4.9. Balanced Random Forest Classifier



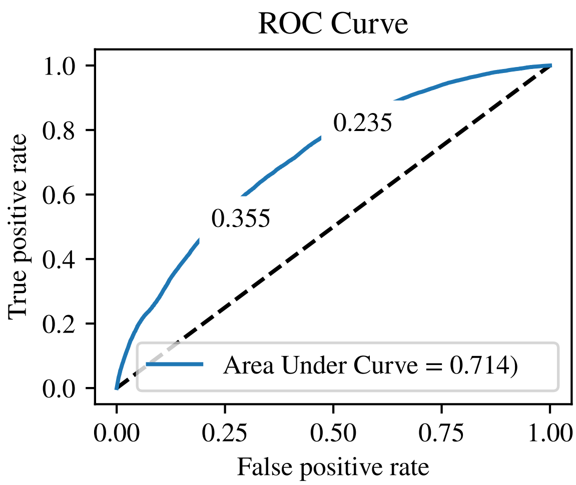


Prediction		N	P
Actual	N	119,102	31,669
	P	13,649	12,972

0.745	Accuracy
0.639	Balanced Accuracy
0.291	Precision
0.699	Balanced Precision
0.487	Recall
0.364	F1
0.574	Balanced F1
0.479	Gmean

Prediction		N	P
Actual	N	134,347	16,424
	P	18,475	8,146

0.803	Accuracy
0.599	Balanced Accuracy
0.332	Precision
0.737	Balanced Precision
0.306	Recall
0.318	F1
0.433	Balanced F1
0.544	Gmean



5. Conclusions

6. Discussion

7. Future Work

Funding Statement

Conflict of Interest

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

Acknowledgements

[STUDENT] contributed to this work in the [FUNDED PROGRAM]

Data Availability

The CRSS data is publicly available at
https://www.nhtsa.gov/crash-data-systems/crash-report-sampling-system

8.

CRediT authorship contribution statement

First Author: Conceptualization, Investigation, Writing - original draft, Visualization. **Second Author:** Supervision, Methodology, Writing - review and editing. **Third Author:** Investigation, Methodology. **Fourth Author:** Data curation, Writing - review and editing.

References

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.

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.

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;.

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.

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

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.

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.

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.

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

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.

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.