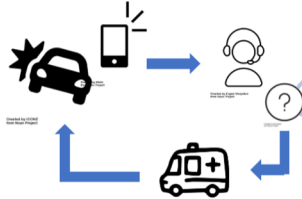# Graphical Abstract

Ambulance Dispatch Recommendation System based on Automated Crash Reports from Cell Phones

J. Bradford Burkman,Chee-Hung Henry Chu,Miao Jin,Xiaoduan Sun

Automated Dispatch Recommendation System Based on Automatic Crash Reports from Cell Phones

Car crash detection on modern mobile phones can automatically notify emergency services, who then decide on dispatching an ambulance

We develop a machine learning-based policy that balances cost against timely medical attention

A machine learning model predicts the need for immediate medical care after a crash. We derive thresholds based on three criteria: ambulance call increase, dispatch success rate, and minimum probability of needing each dispatched ambulance

Highlights

Ambulance Dispatch Recommendation System based on Automated Crash Reports from Cell Phones

J. Bradford Burkman,Chee-Hung Henry Chu,Miao Jin,Xiaoduan Sun

- Supports transferability and benchmarking of different approaches on a public large-scale dataset. We have made public (on GitHub) the code we used to perform the analysis on data from the Crash Report Sampling System (CRSS) and all of the output data.

- Novel Application motivated by Emerging Technology: Machine learning classification models for dispatching ambulances based on automated crash reports from cell phones.

- New Use of Dataset: We used the Crash Report Sampling System (CRSS), which has imputed missing values for some features, but not for all of the features 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, then compared the results with other imputation methods.

- Explicit Incorporation of Imbalanced Costs: False Positives give additional cost, False Negatives give delayed medical care.

- Explicit Incorporation of Political Dimensions: Framing the decision threshold as a government budget question.

- Consideration of Marginal Effects of Threshold Shifting

- Perennial Machine Learning Challenge: Imbalanced Datasets

Ambulance Dispatch Recommendation System based on Automated Crash Reports from Cell Phones

J. Bradford Burkman[a,b,*,1], Chee-Hung Henry Chu[a], Miao Jin and Xiaoduan Sun[c]

[a]*School of Computing and Informatics, University of Louisiana at Lafayette, 301 E. Lewis St, Lafayette LA, 70503, USA*
[b]*Louisiana School for Math, Science, and the Arts, 715 University Pkwy, Natchitoches LA, 71457, USA*
[c]*Department of Civil Engineering, University of Louisiana at Lafayette, 131 Rex St, Lafayette LA, 70504, USA*

ARTICLE INFO

ABSTRACT

Some new cell phones can detect the deceleration profile of a car crash and automatically notify emergency services who will immediately send police, but should they also immediately dispatch an ambulance? Getting medical attention sooner could be life-saving, but most crashes do not require an ambulance. Using historical data, we build an AI recommendation system to predict which ambulances to immediately dispatch within set budgetary constraints. Setting those budgetary constraints is a public policy decision, and we consider three options. We measure the success of the system by its recall, the proportion of needed ambulances that it correctly recommends for immediate dispatch.

The recall depends heavily on the features of the input data. Does the notification come with only a location, or does the cell service provider also share what it knows about the primary user of the phone, like name, age, and sex? Can the system instantaneously correlate the name with vehicle registration records to give the make and model of the car likely involved? We consider three sets of features and show that the recall increases significantly with better information.

Considering the three sets of features, three metrics, and thirteen models, the best result we achieve is that if we have the full feature set, using the budgetary constraint that we can only afford to increase the number of ambulances sent to crashes with automated notifications by 5%, the best model correctly recommends immediately dispatching ambulances to 12% of those who need one.

We trained the models on the Crash Report Sampling System (CRSS) 2016-2021, a publicly available dataset of 700,000 US crashes. We have adapted new methods (new for this dataset in the literature) to handle missing data. To promote discussion and future research, we have posted all of the code we used in our work.

1. Introduction

1.1. Scenario

In the (fictitious) city of Springfield, the city council and mayor are debating whether to immediately dispatch ambulances based on automated notifications from cell phones. Many residents have cell phones (iPhones and Google Pixels) whose accelerometers will detect the deceleration profile of a crash and automatically notify the emergency call center, which immediately dispatches a police officer. The city leaders are pleased that, because of the automated notifications, the police response to the crash scene is faster. Should they also immediately dispatch an ambulance, making the medical response faster?

Traditionally, the emergency call center did not know about a crash until an eyewitness called, and the eyewitness could say whether the crash persons needed an ambulance, but that information does not come with an automated crash notification from a cell phone. The notification will come with a location, the emergency dispatcher already has some information - such as time of day, day of week, weather, urbanicity - and the cell service provider may provide some information about the primary user of the cell phone (age, sex). With that information, the emergency dispatcher has three options:

Always immediately dispatch an ambulance, most of which will not be needed

---

✉ bradburkman@gmail.com (J.B. Burkman)
🌐 http://www.github.com/bburkman/Ambulance_Dispatch (J.B. Burkman)
ORCID(s):

Never immediately dispatch an ambulance; instead, wait for a call from an eyewitness. Many of the ambulances eventually sent to crashes had a cell phone notification and could have been sent sooner.

Sometimes Utilize an AI recommendation system to decide which to send immediately, reserving the option to send an ambulance later based on information from an eyewitness.

In Springfield today, without immediate ambulance dispatch based on automated crash notifications from cell phones, 50% of dispatched ambulances go to automobile crashes and 10% of crash persons need an ambulance. Twenty percent of the crashes first have an automated notification from a cell phone before a call from an eyewitness telling whether or not the crash person needs an ambulance. The other 80% of crashes only have an eyewitness call. Of the crashes with automated notifications from cell phones, 15% will need an ambulance, and 85% will not. In Figure 1 we have scaled the numbers per 100 ambulances sent before implementation of immediate ambulance dispatch.

[We chose these numbers for clarity of illustration; an actual implementation would use local data. For details on the 85/15 split, see §2.2 Dataset and §5 Simplifying Assumptions.]
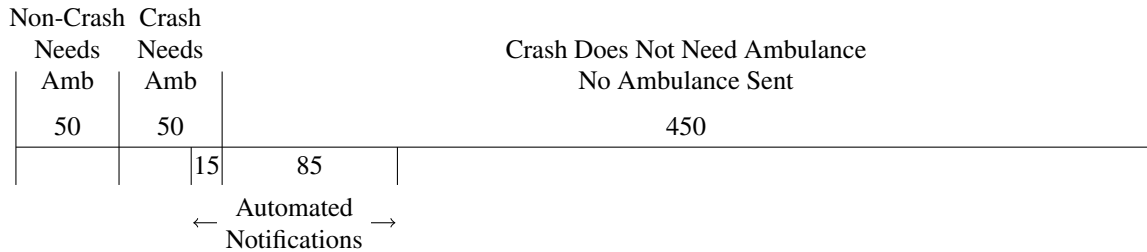

Figure 1: Springfield before implementing immediate dispatch of ambulances. Figure accompanies §1.1

If Springfield were to implement an AI recommendation system to immediately dispatch ambulances based on automated calls from cell phones, the recommendations would not perfectly predict which crash persons need an ambulance. See Figure 2, where we have zoomed in on the left side of Figure 1. In our per-100-ambulances-currently-sent proportions, the recommendation system would classify each of the automated notifications as needing or not needing an ambulance.

Of the fifteen automated crash notifications that need an ambulance, the system would correctly classify some of them as needing an ambulance (True Positive, TP), and those crash persons would get medical attention more promptly, which is the goal and benefit of the recommendation system. The rest of those fifteen would be incorrectly classified as probably not needing an ambulance with a recommendation to wait for a call from an eyewitness before sending one (False Negative, FN). Note that the false negatives get an ambulance just as quickly under the new system as under the old, with an ambulance dispatched upon call from an eyewitness.

Of the 85 automated notifications that do not need an ambulance, some would be correctly classified (True Negative, TN), but some would be incorrectly classified and we would immediately dispatch an unneeded ambulance (False Positive, FP). Besides the cost of administration, those additional ambulance runs are the cost of immediately dispatching ambulances. In the short term those additional ambulance runs could be more than current resources (ambulances and their teams) could handle, and in the long term could be unacceptably expensive.
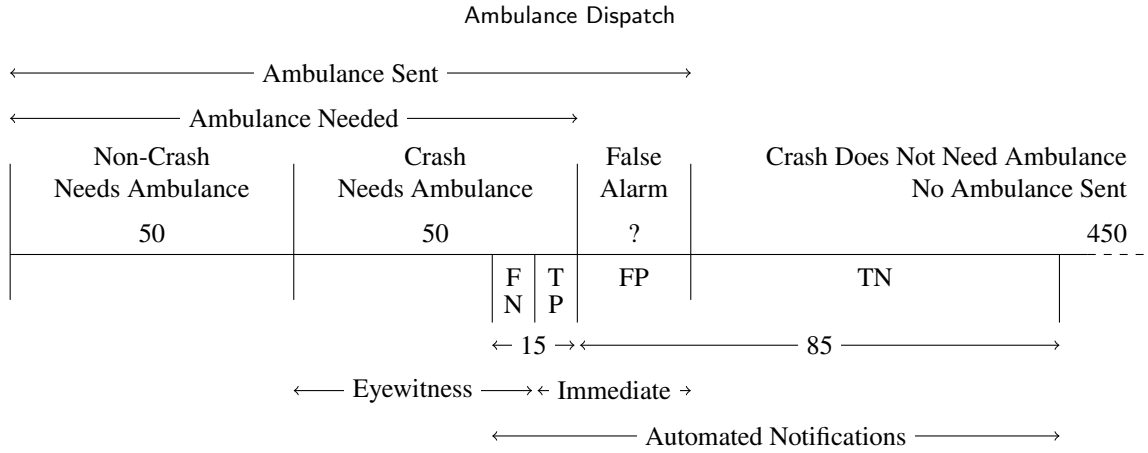
Ambulance Dispatch

←———————— Ambulance Sent ————————→

←———— Ambulance Needed ————→

Non-Crash Needs Ambulance	Crash Needs Ambulance	False Alarm	Crash Does Not Need Ambulance No Ambulance Sent
50	50	?	450

| | | F N | T P | FP | TN |

← 15 →←———————— 85 ————————→

←——— Eyewitness ———→← Immediate →

←———————— Automated Notifications ————————→

Figure 2: Springfield after implementing immediate dispatch of ambulances. Figure accompanies §1.1

The leaders of Springfield need to choose a balance between the benefit of more prompt medical attention and the cost of sending more ambulances. The tradeoff of lives and money is not ethically or morally comfortable, but that is the choice governments make when they set budgets for health care and emergency services. In the confusion matrices in Figure 12, Springfield would love to increase TP without increasing FP, but the recommendation system will not give perfect predictions.

		Prediction PN	PP		Recommendation		Wait for Call from Eyewitness	Immediately Dispatch
Actual	N	TN	FP		Needs	No	Correct	Increased Cost
	P	FN	TP		Ambulance?	Yes	Normal Delay	Prompt Medical Help

Figure 3: Confusion matrix for ambulance dispatch. Figure accompanies §1.1

Building Springfield's AI recommendation system starts with an historical dataset with the features the emergency dispatchers will have at the time of the automated notification - such as time of day, weather, maybe age and sex, and possibly other information - and whether that historical crash person needed an ambulance. This labeled dataset can be the basis of a supervised learning system. A supervised machine learning algorithm learns a model of the data, and when an automated crash notification comes in, given the data available, the model returns a value $p \in [0, 1]$ that increases with the probability that the crash person needs an ambulance. The city council and mayor need to choose a decision threshold θ such that if, for a particular crash notification, $p > \theta$, then immediately dispatch an ambulance; if $p < \theta$, wait for a call from an eyewitness. Choosing a decision threshold of $\theta = 1$ would mean never immediately dispatching an ambulance, and choosing $\theta = 0$ would be always; the town leaders want to know how to choose a θ between the two extremes that fits their values and their budget.

The histogram in Figure 4 shows typical model output. The model generally gives lower p values to crash persons who do not need an ambulance (Neg) and higher p values to crash persons who do need an ambulance (Pos), but there is significant overlap. The most obvious feature of the histogram is the class imbalance, that there are many more Neg than Pos, in fact $85/15 \approx 6$ Neg for each Pos.

Given a choice of decision threshold θ, Springfield would immediately dispatch ambulances to all of the crashes with a p value to the right of θ. The Pos (Needs ambulance) to the right of θ would get more prompt medical attention (TP), but the Neg (Does not need ambulance) to the right of θ would be wasted ambulance runs (FP) . At $\theta = 0.8$, TP and FP are about equal, but as we consider smaller θ the number of TP increases by smaller and smaller amounts while the number of FP grows dramatically.
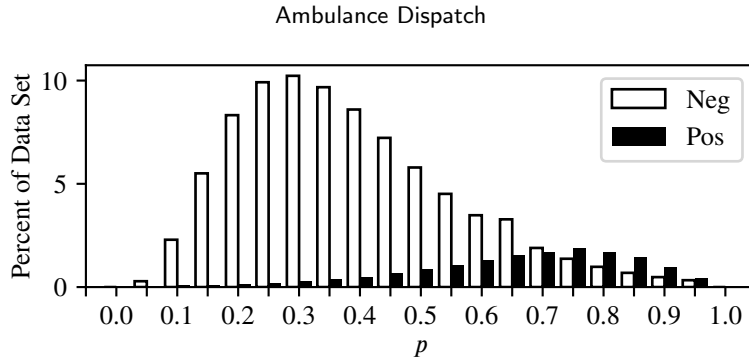
Figure 4: Example model test results. Figure accompanies §1.1

We consider three ways the leaders of Springfield can think about how to choose θ, three metrics for policy decision thresholds, detailed in §2.1. The three metrics are:

1. Percent increase in number of ambulance calls

2. Percent of immediately dispatched ambulances that are actually needed

3. Minimum probability that an immediately dispatched ambulance is actually needed

1.2. Overview

In this paper we explore building a machine learning model to recommend whether to immediately dispatch an ambulance upon an automated notification from a cell phone rather than waiting for a call from an eyewitness.

Such a model would be trained on historical data. We have used the Crash Report Sampling System (CRSS) 2016-2021 data from the US National Highway Traffic Safety Administration (NHTSA), and looked at the data at the level of a person involved in a crash ("crash person") for a total of 713,566 samples. In §2.2 we describe the benefits and limitations of this dataset. The dataset has imputed missing values for some, but not all, of the features we wanted to use, so, in ways that we have not seen in the literature, we recreated the method used by the authors of the dataset to impute missing values (IVEware) and compared it against other methods, deciding on a round-robin random forest method. We briefly describe other aspects of the preprocessing, including binning and engineering features, but interested readers can get full details by reading the code.

Some data features are easily available, like the weather and time of day, but some would be challenging (expensive) to have available in real time, and having some available in real time would pose privacy and data security concerns, so we investigated three levels of data availability that we call Easy, Medium, and Hard, but one can also think of as Free, Expensive, and Problematic. We describe the three levels of data in §2.3 and compare the results in §3.

We used eight supervised learning binary classification algorithms with a variety of hyperparameters, for a total of thirteen models. Binary classification algorithms return, for each sample (crash person, feature vector), a value $p \in [0, 1]$ that increases with the probability that the crash person needs an ambulance, and we have for each sample in the training data set a value $y \in \{0, 1\}$ that tells us whether the crash person actually needed an ambulance. The eight algorithms returned different shapes of output, which we explore in §2.4.1.

The results themselves do not tell us whether we should immediately dispatch an ambulance. We need to choose, for each model, a decision threshold θ such that if for a new automated crash report $p > \theta$, then the system recommends immediately dispatching an ambulance. The criteria for the decision threshold are a matter of government policy, not technical, decisions, and in §2.1 we posit three budgetary criteria: Capping the percentage increase in ambulances sent to crashes with automated notifications (increase because of false alarms), setting a floor under the percentage of ambulances immediately dispatched that are actually needed (precision), and setting a minimum likelihood that the last ambulance sent is actually needed. For each criterion we have created a metric to determine the decision threshold on the p. Two metrics measure the entire dataset on either side of the threshold, and one measures marginal effect at the threshold. We develop a method for measuring such marginal effects using the model output. For each criterion we compared results of different models and sets of features using the simple measure of, at the decision threshold, which model immediately dispatches the largest number of needed ambulances (TP, which is proportional to the recall).

We found that the first criterion's metric is a nondecreasing function of p, so choosing a decision threshold is straightforward. The second and third criteria's metrics, however, are not smooth functions of p. They generally decrease from $p = 0$ to $p = 1$, but not at a finely granular level. In §2.5.1 we consider some methods for smoothing the results to choose the decision threshold.

Models built on the Easy features do not do nearly as good a job at separating the negative and positive classes as do the models built on the Medium features. The models built on the Hard features have the best results. In §3 we quantitatively compare the models and feature sets in terms of false alarms. For some metrics the Easy decision threshold may exist only in the far tail of the distribution, too far out to be actionable. Since the cost of having the Medium or Hard features available may be mostly a fixed cost, and the number of false positives recommended by the system is proportional to the size of the city or region implementing the system, the economies of scale of such a recommendation system may be considerable.

To support transferability, we have used a public dataset and made public all of the code we used in our investigation with all of the output data. This paper gives some results to support the conclusions and invites the interested reader to explore the full results, tinker with the code, and use any of it to investigate further topics. Simplifying assumptions and opportunities for future research are given in §5.

1.3. Literature Review
1.3.1. Smartphone
(Liu, Racz, Vaillancourt, Michelman, Barnes, Mellem, Eastham, Green, Armstrong, Bal, O'Banion and Guo, 2023), *Smartphone-based hard-braking event detection at scale for road safety services*, **Transportation Research Part C: Emerging Technologies**,

Use hard braking as an indicator of crash risk at a location.

Overcomes sparsity of crashes in data.

Data from Google Maps and Android Auto

1.3.2. Location Allocation
(Liu, Li, Liu and Patel, 2016) Liu 2016, "A double standard model for allocating limited emergency medical service vehicle resources ensuring service reliability."

32 sources

Distribution of EMS vehicle depots

Allocation of EMS vehicles to incident sites

Nomenclature section

Average response time is key measure to assess effectiveness of emergency response.

(Nadar, Jha and Thakkar, 2023), *Adaptive variable neighbourhood search approach for time-dependent joint location and dispatching problem in a multi-tier ambulance system*, **Computers & Operations Research**,

55 Sources

Multi-tier ambulance system?

Three key decisions

    Location of ambulance stations,

    Allocation of ambulances to locations,

    Preference order of locations for dispatching

A mixed-integer non-linear programming model is formulated with a survival probability-based objective function

Data from city of Kolkata in India

1.3.3. OnStar
(He, Zhang and Wang, 2019), *Crash Telemetry-Based Injury Severity Prediction is Equivalent to or Out-Performs Field Protocols in Triage of Planar Vehicle Collisions.*, **Prehospital & Disaster Medicine**,

"Vehicle telemetry technology"

"Advanced Automatic Crash Notification (AACN)"

"Injury Severity Score (ISS)," and they looked at ISS 15+ injuries

Constructed an Injury Severity Prediction (ISP) model.

Used sensitivity (recall) and specificity ($TN/(TN+FP) = TN/N = TNR$)

To construct the logistic regression model, used NASS-CDS, National Automotive Sampling System Crashworthiness Data System.

Used OnStar data, hospital records to validate the model.

Variables:	Principal direction of force (PDOF)
	Change in Velocity (δV) = deceleration
	Multiple impacts
	Presence of any older occupant (≥ 55)
	Presence of any female occupant
	Presence of any right-sided passenger
	Seat belt use
	Vehicle type

(Kononen, Flannagan and Wang, 2011), *Identification and validation of a logistic regression model for predicting serious injuries associated with motor vehicle crashes*, **Accident Analysis & Prevention**

2010 version of paper above, (He et al., 2019)

1.4. AACN, not OnStar

(Plevin, Kaufman, Fraade-Blanar and Bulger, 2017), *Evaluating the Potential Benefits of Advanced Automatic Crash Notification.*, **Prehospital & Disaster Medicine**, 2017

Did not use AACN data, but if we had that data we could predict whether the emergency responders would need specialized equipment to extricate the crash person from the vehicle.

(Bose, Crandall, McGwin, Goldman, Foster and Fine, 2011), *Computational methodology to predict injury risk for motor vehicle crash victims: A framework for improving Advanced Automatic Crash Notification systems*, **Transportation Research Part C: Emerging Technologies**,

Predict injury in specific body regions based on specific characteristics of the crash, occupant, and vehicle.

(Jagtenberg, Bhulai and van der Mei, 2015), *An efficient heuristic for real-time ambulance redeployment*, **Operations Research for Health Care**,

"Advanced automatic crash notification (AACN)"

Societal benefits were assessed by comparing correctly triaged motor vehicle crash occupants using the AACN algorithm against real-world decisions.

(Weaver, Talton, Barnard, Gaffley, Doud, Schoell, Petty, Martin, Meredith and Stitzel, 2022), *Advanced Automatic Crash Notification Algorithm for Children*, **Academic Pediatrics**,

Pediatric version of her (Weaver's) articles for adults

Algorithm inputs included: delta-v, rollover quarter-turns, belt status, multiple impacts, airbag deployment, and age.

1.4.1. Injury Prediction Models

(Kong, Lee, Kim, Lee, Kang, Choi, Kim, Jeong, Kang and Sung, 2023), *Machine learning-based injury severity prediction of level 1 trauma center enrolled patients associated with car-to-car crashes in Korea*, **Computers in Biology and Medicine**,

To handle class-imbalanced clinical datasets, we used four data-sampling techniques (i.e., class-weighting, resampling, synthetic minority oversampling, and adaptive synthetic sampling). Machine-learning analytics based on logistic regression, extreme gradient boosting (XGBoost), and a multilayer perceptron model were used for the evaluations. Each model was executed using five-fold cross-validation to solve overfitting

Age was a significant predictor

(Candefjord, Sheikh Muhammad, Bangalore and Buendia, 2021), *On Scene Injury Severity Prediction (OSISP) machine learning algorithms for motor vehicle crash occupants in US*, **Journal of Transport & Health**,

In this study, we evaluate if methods employing machine learning and variables **that can be assessed on the scene of accident** has potential to amend field triage

1.5. General Dispatch Algorithms, not particularly for Crashes

(Zhang, He, Gou and Li, 2016) Zhang 2016 TRpC "Performance measure for reliable travel time of emergency vehicles"

33 sources

Travel time reliability. Apparently a big topic.

(Bertsimas and Ng, 2019), *Robust and stochastic formulations for ambulance deployment and dispatch*, **European Journal of Operational Research**,

54 Sources

Set of locations for ambulances

(**?**), *Traffic emergency vehicle deployment and dispatch under uncertainty*, **Transportation Research Part E: Logistics and Transportation Review**,

1.5.1. Privacy

(Geuens and Dumortier, 2010), *Mandatory implementation for in-vehicle eCall: Privacy compatible?*, **Computer Law & Security Review**,

An in-vehicle eCall would reduce road fatalities with about 2500 deaths every year and the severity of accidents with 15% by reducing the response time of the emergency services in urban areas with 40% and in rural areas with 50%. This response time is reduced because the emergency services are notified instantly in case of an accident and are given the exact location of the accident.

There is however strong resistance against in-vehicle eCall, especially if it would be made mandatory on all new cars in the Community. This resistance stems from fear for invasion in private life and risks regarding personal data.

Possible text:

A legal analysis of the possibility of mandating advanced automatic crash notification (AACN) systems in new cars sold in the EU reflected the concern that the vehicle speed data in the notification could be used to prosecute the driver and void the insurance. (Geuens and Dumortier, 2010)

1.5.2. Coordinating Traffic Signals

× (Su, Zhong, Chow, Dey and Jin, 2023) "EMVLight: A multi-agent reinforcement learning framework for an emergency vehicle decentralized routing and traffic signal control system"

Synching emergency vehicle routing with traffic signal control.

85 sources

(Qin and Khan, 2012) Qin 2012 *Control strategies of traffic signal timing transition for emergency vehicle preemption*,

32 sources

Traffic signal preemption
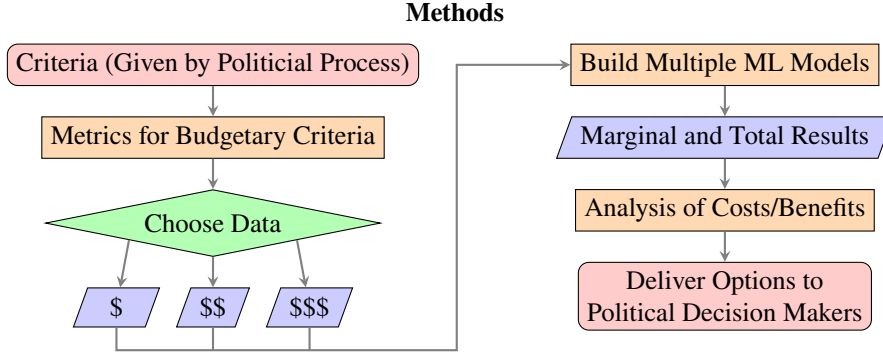
1.5.3. Other

2. Methods

Methods

Figure 5: Methods Graphical Abstract

2.1. Budgetary Decision Thresholds and Corresponding Metrics

Saying that we trade off lives for money makes us uncomfortable, but that is what governments do when they set budgets for health care and emergency services. All budgets are finite, and spending more money has diminishing returns, so we have to choose some criteria for our decision and accompanying metrics that let us quantify the criteria to choose an appropriate decision boundary for our recommendation system.

Our Springfield scenario illustrates the tradeoffs. We consider three ways to set the decision threshold.
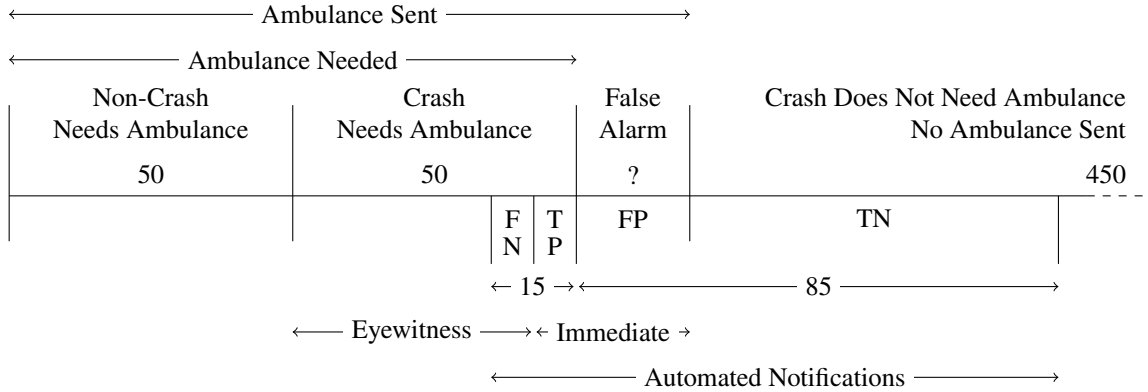
Figure 6: Springfield after implementing immediate dispatch of ambulances. Figure accompanies §1.1

2.1.1. Budgetary Decision Metric I: Percent Increased Number of Ambulance Calls

When a city or region implements immediate dispatch of ambulances, the number of ambulance dispatches increases by FP. Within the automated notifications, P = FN + TP ambulance runs becomes P + FP = FN + TP + FP ambulance runs, an increase by a rate of FP/P. The increase does not include the true positives (TP), because those ambulances would go eventually with or without immediate dispatch; the increase is the number of false positives (FP).

In the short term (too short to buy more ambulances and hire more teams), the existing budget can support an increase in the number of ambulance runs to crash persons by some small percentage. In the longer term, the city is willing to increase the budget to increase the number of ambulances going to crashes with automated notifications by a larger, but still fixed, percentage. We use 5% as our example of how to implement this policy, setting the decision threshold θ where the number of false positives is 5% of the positive class, giving our first budgetary decision metric,

$$\frac{FP}{P} = \frac{FP}{FN + TP} \le 0.05 \tag{1}$$

In our Springfield scenario, scaling to 100 currently sent ambulances to crash or non-crash, the total number of ambulance runs goes from 100 to 100 + FP, a rate of increase of FP/100. For any city or region, if we knew the proportion of crashes with automated notifications from cell phones and the proportion of ambulances going to crashes, we could choose an FP/P threshold to match a budgetary decision criterion based on the increase in total number of ambulances sent to crashes or total number of ambulances sent to any situation.

2.1.2. Budgetary Decision Metric II: Percent of Immediately Dispatched Ambulances Actually Needed

A city or region is willing to immediately dispatch ambulances based on automated crash reports, but only up to the point where a certain proportion of the ambulances they immediately dispatch (PP = FP + TP) are actually needed (TP). This proportion, TP/(FP + TP) is called the *precision* of a machine learning model. In this paper we use the term *precision* in this sense and *numerical precision* to describe the confidence we can have in a certain number of decimal places of a result.

To illustrate the method, we choose Precision = 2/3, being willing to immediately dispatch one unnecessary ambulance for each two necessary ones, giving our second budgetary decision metric,

$$\frac{\text{TP}}{\text{PP}} = \frac{\text{TP}}{\text{FP} + \text{TP}} \geq \frac{2}{3} \tag{2}$$

2.1.3. Budgetary Decision Metric III: Minimum Probability that Each Immediately Dispatched Ambulance is Needed

The previous two decision criteria let the city leaders choose a specific dollar amount of increase in the annual ambulance budget, but for ethical reasons they may decide that, while they cannot afford to immediately dispatch an ambulance to every crash notification, they should immediately dispatch an ambulance to a crash notification with some probability (like 50% or 80%) of needing medical attention, and consider the total cost later. From our model results, can we find a decision threshold θ that corresponds to such a probability?

Our recommendation system uses a supervised-learning binary classification model trained on historical data. The models do not actually return a probability but return, for each sample, a value p that generally increases with the probability. For each sample we also know whether that historical crash person actually needed an ambulance (whether that sample is in the negative or positive class).

Consider a small band of values of p; the samples in that band are either in the negative or positive class. Call the number of negative and positive samples in the band Neg and Pos, to distinguish from the total number of samples in the negative and positive classes, N and P. The probability that a crash person in that band of p needs an ambulance is given by Pos/(Neg + Pos), which we call "marginal probability," or "mProb."

To illustrate the method, we choose the minimum marginal probability to be 50%, meaning that each ambulance we immediately dispatch has at least a fifty percent chance of being needed, giving our third budgetary decision metric,

$$m\text{Prob} = \frac{\text{Pos}}{\text{Neg} + \text{Pos}} \geq 0.5 \tag{3}$$

We can relate this metric to a familiar metric if we rephrase the probability as the ratio of needed to unneeded ambulances immediately dispatched, *i.e.* Pos:Neg. For instance, a 50% probability is a 1:1 ratio of needed to unneeded, and an 80% probability is a 4:1 ratio of needed to unneeded. The proportion of needed to unneeded ambulances sent in a neighborhood of p is proportional to a widely used metric, the slope of the ROC curve, with the constant of proportionality being the class ratio, as shown here:

$$\frac{\text{Pos}}{\text{Neg}} = \frac{\Delta \text{TP}}{\Delta \text{FP}} = \frac{\text{P}}{\text{N}} \cdot \frac{\Delta \text{TP/P}}{\Delta \text{FP/N}} = \frac{\text{P}}{\text{N}} \cdot \frac{\Delta \text{TPR}}{\Delta \text{FPR}} = \frac{\text{P}}{\text{N}} \cdot m\text{ROC} \tag{4}$$

2.2. Dataset

Ideally, we would use a dataset of crashes that spawned an automated notification, but we have not found such a dataset that is publicly available. Working with such a private dataset would be an important avenue of future research. (See §5 for a list of simplifying assumptions and opportunities for future research.)

We use the Crash Report Sampling System (CRSS) data from 2016 to 2021. The CRSS, from the US National Highway Traffic Safety Administration (NHTSA), is a curated sample of crashes in the US, weighted to more serious crashes such that 17% of the crash persons needed an ambulance, significantly more than the proportion of all reported crashes needing an ambulance. Since many low-speed crashes would have a crash profile similar to hard braking, they would not spawn an automated notification, so it is reasonable to assume that the set of crashes with automated notifications would have a higher percentage of persons needing an ambulance.

We make some simplifying assumptions (see §5) using this dataset, including that the class ratio (N:P) in the automated crash notification from cell phones is close to that in the CRSS data, 5:1, and that the crash persons in the CRSS data are representative of the future crash persons whose cell phones send a crash notification. We use the CRSS as a proxy for the set of crashes with automatic crash notifications, acknowledging that we do not know how good of a proxy it is. The primary merit of CRSS for our work is that it is publicly available so that our work can be critiqued, adapted, and expanded by others.

To prepare the data we had to bin (discretize) some features and to impute missing data. Some features in CRSS have both the original data with values signifying "Missing" or "Unknown" and a new feature with missing values imputed using IVEware (Raghunathan, Solenberger, Berglund and van Hoewyk); nevertheless, not all of the features we wanted to use had imputed values. CRSS has a very useful document on the history of its imputation methods going back to 1988 with the predecessors of the current data set (Herbert, 2019). We debated the proper order of operations for binning and imputing, tried both, and decided to bin first, then impute. We tried several methods of imputation, including the IVEware used by the CRSS authors, but got better results using a round-robin random forest method. Full details and analysis are in the code, listed below.

We removed all crashes involving a pedestrian because deceleration profile of such a crash would be more like hard braking than hitting a large immovable object like a car or tree, so less likely to trigger an automated notification.

The dataset is slightly imbalanced, with five elements of the negative class for each element of the positive class. We considered several methods to handle the imbalance, including resampling, class weights, focal loss (Lin, Goyal, Girshick, He and Dollár, 2017), and balanced metrics. A related study predicting injury severity using a Korean dataset used class weights, resampling, the popular SMOTE oversampling method, and adaptive synthetic sampling (ADASYN) (Kong et al., 2023). We cannot use the SMOTE because our data is categorical (Chawla, Bowyer, Hall and Kegelmeyer, 2002). We tried undersampling with Tomek Links, but the resulting model results were not significantly different. The best results came from using the model algorithms from Imbalanced-Learn (Lemaître, Nogueira and Aridas, 2017), some of which apply bagging on top of algorithms from Scikit-Learn (Pedregosa, Varoquaux, Gramfort, Michel, Thirion, Grisel, Blondel, Prettenhofer, Weiss, Dubourg, Vanderplas, Passos, Cournapeau, Brucher, Perrot and Duchesnay, 2011).

After removing those pedestrian crashes we had 713,566 samples, each representing a crash person, and 78 relevant features.

For details, see these sections of code in the `Keras` folder at
`www.github.com/bburkman/Ambulance_Dispatch`.

```
Ambulance_Dispatch_01_Get_Data.ipynb
Ambulance_Dispatch_02_Correlation.ipynb
Ambulance_Dispatch_03_Bin_Data.ipynb
Ambulance_Dispatch_04_Impute_Missing_Data.ipynb
Ambulance_Dispatch_05_IVEware_Order_of_Operations.ipynb
Ambulance_Dispatch_06_Build_Models_Tomek_Links.ipynb
```

2.3. Choosing Features

The `Accident`, `Vehicle`, and `Person` files of the CRSS dataset 2016-2021 have 170 unique features.

First we narrowed the features to those that are relevant, of good quality, and knowable at the time of the automated notification (before any eyewitness reports). Some features, like Vehicle Identification Number (VIN), have no predictive value. Other features have missing data for more than 20% of samples. Some features, like drug and alcohol test results, are unknowable at the time of the automated notification.

Having data available for instantaneous analysis when the crash notification arrives is not free, and some features are more expensive than others. A city thinking of implementing a recommendation system for immediate dispatch would need to decide how much to spend to have the data available, and whether the more expensive features increase the

Table 1

Easy, Medium, and Hard Features. Table accompanies §2.3

Source	Easy Features	Additional Features in Medium Set	Additional Features in Hard Set
Accident .csv	Day of week Hour Month Police jurisdiction Primary Sampling Unit (Geographical region) Region Urbanicity Weather Year	Interstate highway? Relation to road Within junction? Type of Intersection	Lighting conditions Number of persons in all vehicles in accident Number of parked or working vehicles Number of moving motor vehicles Total number of motor vehicles Location within junction School bus? Work zone?
Vehicle.csv		Roadway alignment Total lanes in roadway Roadway grade Speed limit Traffic control device Trafficway description	Body type Being used as a bus? Emergency use? Make Number of occupants Roadway surface condition
Person.csv		Age Sex	Driver/Passenger
Engineered Features		Age × Sex	Vehicle age (Model year and Year) Age × School bus

quality of the models enough to be worth the cost. We categorized the features as "Easy," "Medium," and "Hard," which can also be called "Free," "Expensive," and "Problematic." The "Easy" features are those the dispatcher already has, like day of week, time of day, weather, and urban/rural. The "Medium" features add details about the location (intersection, speed limit, interstate highway) and information the cell service company probably has about the primary user of the phone (age and sex). To have the medium features instantaneously available would require coordination of many resources and be expensive to set up. The "Hard" features are much more problematic, requiring more coordination of public and private records, and having that data readily available introduces privacy and data security concerns. Hard features include whether the location is a work zone, the likely vehicle driven by the primary user of the cell phone, and, if there are multiple automated notifications from the same location, how many crash persons are likely to be involved.

See Table 2.3 for a list of features from each of the `Accident`, `Vehicle`, and `Person` CRSS files and some features we engineered, for the Easy features, additional features in the Medium features set, and more features in the Hard features set.

A similar study in 2011, reprised with new data in 2019, (Kononen et al., 2011; He et al., 2019), asked whether emergency dispatchers can use data from Advanced Automatic Collision Notification (AACN) systems like GM OnStar to predict whether a crash had a serious injury. See Table 2.3 for the features used in those studies with a comparison to the features we used for a system based on notifications from cell phones.

We note here our simplifying assumption (§5) that an actual implementation would have complete and accurate data for each automated notification. Also, we test three combinations of features but have not done more detailed testing to see which individual features or other groupings of features are most or least useful in predicting whether a crash person needs an ambulance.

See `Ambulance_Dispatch_01_Get_Data.ipynb` for a list of the excluded features.

Table 2

Comparison with Features in Similar Papers on Vehicle-based Advanced Automatic Collision Notification systems like GM OnStar. (Kononen et al., 2011; He et al., 2019) Table accompanies §2.3

Feature	Vehicle-based v/s Cell Phone
Principal direction of force (PDOF) Change in Velocity (δV) Multiple impacts	Whether cell phone collects this data depends on sensitivity of accelerometer and amount of memory reserved for this seldom-used function.
Presence of any older occupant (≥ 55) Presence of any female occupant	Are these features knowable by a vehicle-based AACN? The cell service provider may have data about the primary user; we put Age and Sex in our Medium features
Presence of any right-sided passenger Seat belt use	Knowable by AACN from sensors in seats and seat belts, but not knowable to cell phone
Vehicle type	Knowable to the AACN installed in the vehicle, but only knowable with a cell phone notification if the system correlates the phone user's ID with vehicle registration records; we put information about the vehicle in our Hard features

See `Ambulance_Dispatch_03_Bin_Data.ipynb` for a list of the features we used for imputation of missing data in CRSS.

See `Ambulance_Dispatch_07_Build_Models.ipynb`. for the complete list of the features used in the Easy, Medium, and Hard model building with their CRSS names, like `VALIGN` for roadway alignment. See the CRSS Analytical Users's Manual for complete details on all of the features. (NHTSA, 2016-2020)

2.4. Models

See `Ambulance_Dispatch_07_Build_Models.ipynb` for more details.

2.4.1. Binary Classification Algorithms and Hyperparameters

For each of the three sets of features we used eight binary classification algorithms, three of which take class weight α and one of which takes the focal loss parameter γ. (See Table 2.4.1) We learned models for various values of the hyperparameters, giving $3 \times 13 = 39$ different models. The $\alpha = 0.5$ class weight is the default, and the $\alpha = 0.85$ class weight balances the effect of the negative and positive class in the loss function, as 85% of the samples are in the negative class. Focal loss (Lin et al., 2017) puts more weight in the loss function on the samples that are badly classified, much like least squares regression puts more weight on the points furthest from the line. Setting $\gamma = 0.0$ has no effect; Lin's paper tested from $\gamma = 0.5$ to $\gamma = 5.0$ and recommended $\gamma = 2.0$.

Table 3

Models Tested for Recommendation System. Table accompanies §2.4.1

Model	Abbreviation	Source	Class Weight α	Focal Loss γ
AdaBoost Classifier	AdaBoost	Scikit-Learn		
Balanced Bagging Classifier	Bal Bag	Imbalanced-Learn		
Balanced Random Forest Classifier	Bal RF	Imbalanced-Learn	0.5	
			0.85	
Easy Ensemble Classifier with AdaBoost Estimator	Easy Ens	Imbalanced-Learn		
KerasClassifier with the	Keras	Keras	0.5	0.0
Binary Focal Crossentropy loss function			0.5	1.0
			0.5	2.0
			0.85	0.0
Logistic Regression Classifier	Log Reg	Scikit-Learn	0.5	
			0.85	
Random Forest Classifier	RF	Scikit-Learn		
RUSBoost Classifier	RUSBoost	Imbalanced-Learn		

2.4.2. Hyperparameter Effects

Our experience with varying the class weight and focal loss parameters was that they shifted the entire p distribution, both the positive and negative class, but did not do a better job of separating the positive and negative class, as measured by the area under the curve (AUC) of the receiver operating characteristic (ROC) curve, as illustrated in Figure 7.

The KerasClassifier with the Binary Focal Crossentropy loss function takes both a class weight hyperparameter α and a focal loss parameter γ. Varying these hyperparameters gave us different shapes of distributions of p, as shown in Figure 7, which also gives the ROC AUC for two runs of each model with different random seeds. That the model on the left has both the least and the greatest ROC AUC illustrates that, between models of the same algorithm with different class and focal weights, the effectiveness of the model in separating the positive and negative classes depends as much on the random seed as on the choice of weights.

If one is using the default decision threshold $\theta = 0.5$, then these hyperparameters are useful for shifting the distribution to meet the threshold, but since we are taking the liberty to move the threshold, varying the hyperparameters may be of little use. Since the ROC AUC quantifies how well the algorithm separates the positive and negative classes over the whole range of p, and we are most interested in a small range of p on the right end, the weights may yet have some useful effect, a topic for future investigation (§5).
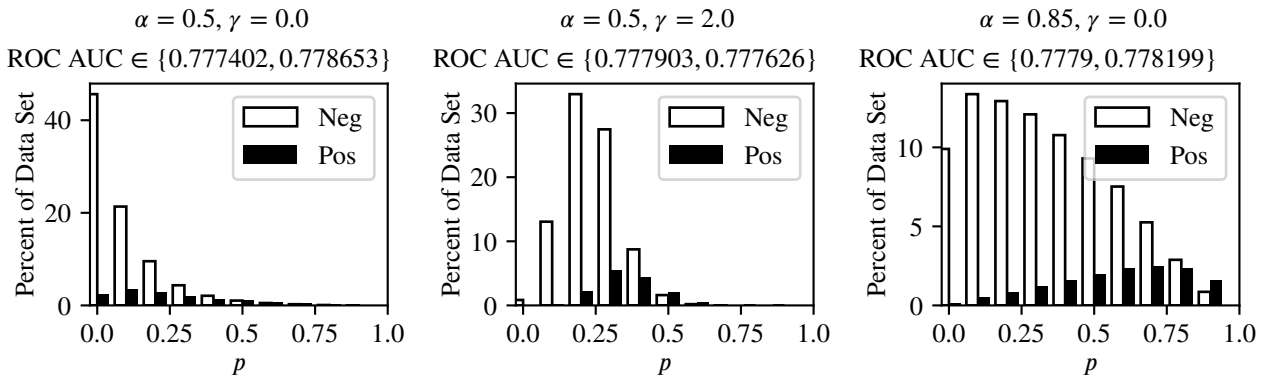

Figure 7: Histograms of p results for KerasClassifier with Different Hyperparameters, with ROC AUC values for Two Model Runs with Different Random Seeds. Figure accompanies §2.4.2

2.4.3. Five-Fold Cross Validation

As mentioned in §2.1.3, we need enough samples in each band of p to smooth out the randomness. If we had more samples total, we could use smaller bands and have more numerical precision in our specification of the decision threshold θ. If we used a typical 70/30 train/test split, we would only have p values for the 30% samples in the test set. Instead we used five-fold cross validation, having an 80/20 split five times, giving us p values for all of the samples.

2.4.4. Interpreting Supervised Learning Binary Classification Results

In supervised learning binary classification, a model predicts, for each sample in the test set, whether the sample is in the negative or positive class. The model returns a value $p \in [0, 1]$, that increases with the probability that the sample is in the positive class. Additionally in supervised learning, we already know the answer to the question of whether the sample was in the negative or positive class, with $y \in \{0, 1\}$ given in the dataset but hidden from the model during the test phase. In the code, p is often called y_proba and y is called y_test. Using these two numbers, p and y, we can study, quantify, and illustrate how well the model predicts the actual values.

A perfect model would entirely separate the negative and positive classes, but the ideal we can hope for is that most of the negative elements are towards the left and most of the positive elements are towards the right of the distribution. In Figure 8, the data has the same class ratio as our CRSS data, with 85% in the negative class and 15% in the positive class. If we choose discrimination threshold $\theta = 0.5$, the value of p for which most models algorithms are optimized, the elements of the negative class with $p < \theta$ are True Negatives (TN), the elements of the negative class with $p > \theta$ are False Positives (FP), the elements of the positive class with $p < \theta$ are False Negatives (FN), and the elements of the positive class with $p > \theta$ are True Positives (TP).

If this ideal model were our recommendation system with $\theta = 0.5$, then 38.5% of the ambulances we immediately dispatched would be needed (Precision), and 73% of the needed ambulances would be immediately dispatched (Recall). If we chose a higher value of θ, we would increase TN, decrease FP, increase FN, and decrease TP. Recall would decrease, but the effect on precision is uncertain as FP and TP would both decrease.

The ROC (Receiver Operating Characteristic) curve is a parameterized curve showing the True Positive Rate (TP/P) versus the False Positive Rate (FP/N) as p varies from 0 (upper right) to 1 (lower left). The area under the curve (AUC) is widely used to compare the quality of models in terms of how well they separate the negative and positive classes over the entire range of $p \in [0, 1]$. For our work, however, given real-life budgetary constraints on expanding ambulance fleets, we are only interested in a small range of p on the right side of the distribution, so the ROC AUC is not the primary measure we use to choose the best model.

	Prediction	
	N	P
Actual N	66.5%	18.5%
Actual P	3.4%	11.6%

0.385 Precision
0.773 Recall
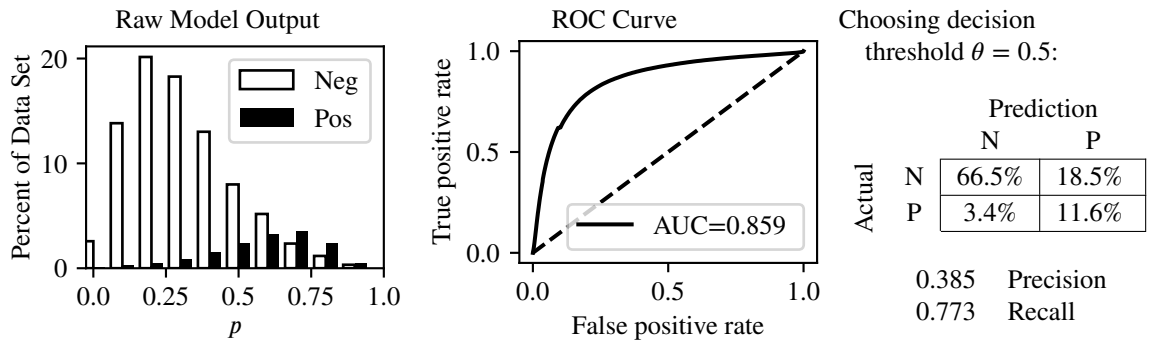
Figure 8: Example Model Results. Figure accompanies §2.4.4

2.4.5. Comparing Outputs of Different Models

The eight model algorithms not only give different results, but different kinds of results, and we find a way to compare them. See Figure 9. For the illustrations we have used models built on the Hard features with no class balance nor focal loss.

The ranges and shapes of the distributions are significantly different. The Balanced Bagging and Balanced Random Forest classifiers gives a nice spread from 0.0 to 1.0, but the AdaBoost, Easy Ensemble, and RUSBoost results are clustered in the middle, and the Random Forest on the left. If we used the Random Forest results with the default

decision threshold $\theta = 0.5$, we would never immediately dispatch an ambulance. The KerasClassifier and Logistic Regression Classifier tend towards the left with long tails to the right.

To give appropriate results for Balanced Bagging with a p range of width 1.0 and RUSBoost with a range of width 0.002, we built into the code a variable for numerical precision that depends on the range of p for that model, num_prec $= -\lceil \log_{10}(\max(p) - \min(p)) \rceil$, to which we can then add the number of digits appropriate for the purpose.

The numerics in the model results are also different. Most of the models give, for the 713,566 samples, at least 100,000 unique values of $p \in [0, 1]$, making the discrete results nearly continuous. The AdaBoost, KerasClassifier, Logistic Regression, and RUSBoost algorithms are fairly consistent with about 140,000 unique values of p for results on the Easy features, 640,000 for the results on the Medium features, and 700,000 on the Hard features. The Random Forest classifier gives about half as many unique results, but proportionally for the three features sets.

The outliers in the numerics are the Balanced Random Forest, Balanced Bagging, and Easy Ensemble. The Balanced Random Forest method on the Hard features gives fewer than 4,000 unique values of p on the Hard features because 93% of its p values are rounded to two decimal places. The Balanced Bagging results on the Hard features have fewer than 300 unique values because 99% of its p values are rounded to one decimal place. While the other algorithms had fewer unique values for the Medium features and far fewer for the Easy, the Balanced Random Forest and Balanced Bagging have more for the Medium and far more for the Easy, with the number of unique Easy results being comparable to those of the other algorithms. The Easy Ensemble results for all three features sets have about 50% of the results rounded to two decimal places. For full details see `Analyze_Proba/Value_Counts_y_proba.csv`.
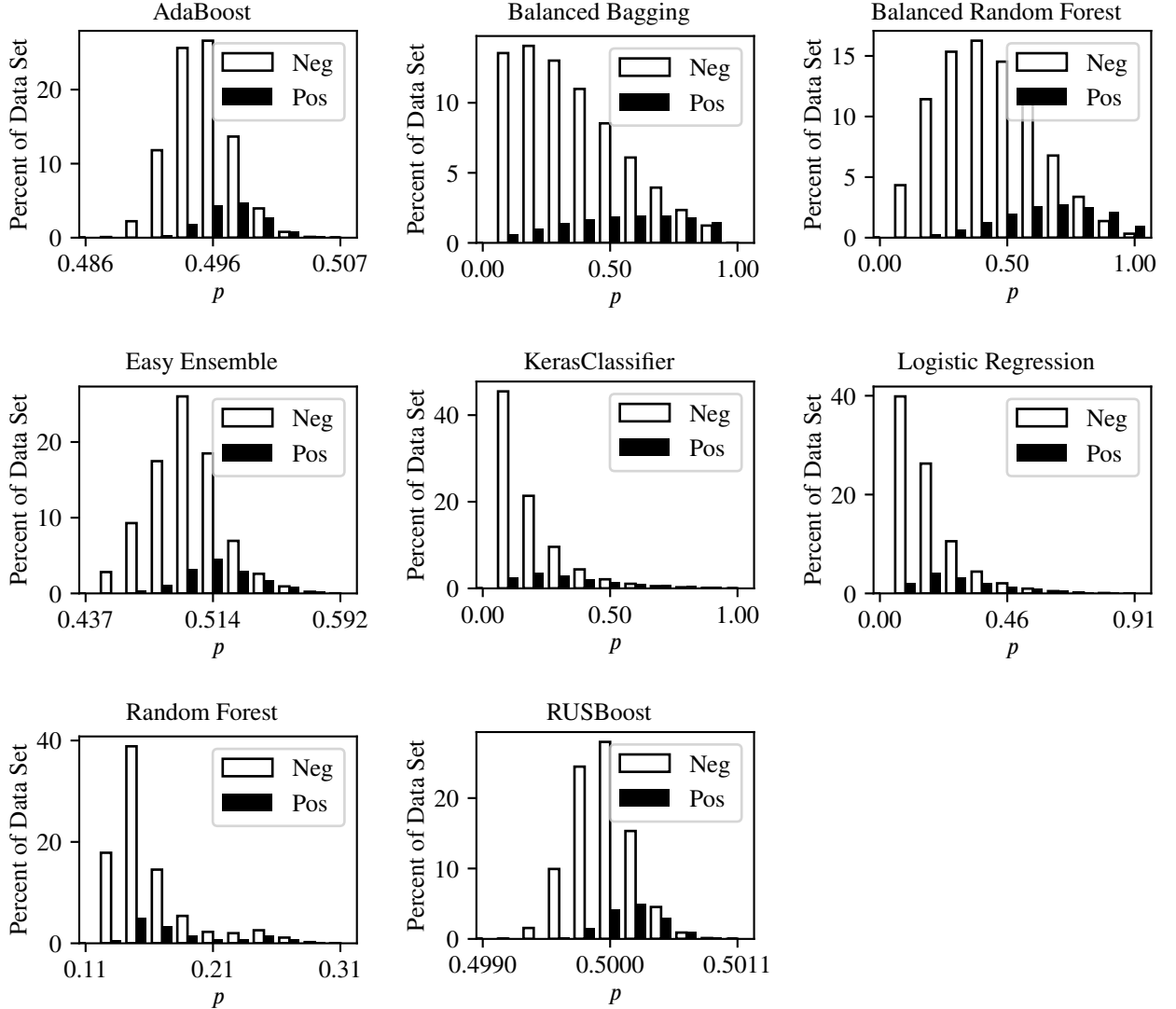
Figure 9: Raw Model Outputs. Figure accompanies §2.4.5

2.5. Interpreting Model Results
2.5.1. Finding θ for Each Budgetary Decision Metric

For each budgetary metric and each model we need to find a corresponding decision threshold θ. For each new crash notification the model outputs a prediction value p, and if $p \geq \theta$ the system recommends immediately dispatching an ambulance.

For each of the 713,566 samples we use from the CRSS dataset we have a target value $y \in \{0, 1\}$ that tells whether the sample is in the Neg or Pos class. Each model returns for each of those historical samples a value $p \in [0, 1]$ that indicates whether the model predicts that the sample is in the positive and negative class. For each model we need to find the value of p that corresponds to each of the budgetary decision criteria to set as the decision threshold θ.

To illustrate our methods for finding θ for each budgetary decision metric and each model we use the results of the Random Forest Classifier on the hard features, which returns values $p \in [0.10619, 0.311074]$. Table 4 shows some of the results. We have rounded p to four decimal places and sorted model results by increasing p, giving the number of elements of the negative and positive classes for each p. From Neg and Pos we can calculate the TN, FP, FN, and

Table 4

Metrics on p Output of Random Forest Classifier on the Hard Features. Table accompanies §2.5.1

p	Neg	Pos	mProb	TN	FP	FN	TP	Prec	Rec	FP/P
0.1062	1	0	0.0000	1	605,609	0	107,956	0.1513	1.0000	5.6098
0.1063	86	2	0.0227	87	605,523	2	107,954	0.1513	1.0000	5.609
0.1064	51	0	0.0000	138	605,472	2	107,954	0.1513	1.0000	5.6085
⋮	⋮	⋮	⋮		⋮	⋮	⋮	⋮	⋮	⋮
0.2579	33	40	0.5479	600,161	5,449	102,100	5,856	0.518	0.0542	0.0505
0.258	39	26	0.4000	600,200	5,410	102,126	5,830	0.5187	0.0540	0.0501
0.2581	39	25	0.3906	600,239	5,371	102,151	5,805	0.5194	0.0538	0.0498
0.2582	42	31	0.4247	600,281	5,329	102,182	5,774	0.5200	0.0535	0.0494
⋮	⋮	⋮	⋮		⋮	⋮	⋮	⋮	⋮	⋮
0.2858	4	8	0.6667	605,179	431	106,858	1,098	0.7181	0.0102	0.0040
0.2859	1	9	0.9000	605,180	430	106,867	1,089	0.7169	0.0101	0.0040
0.286	4	11	0.7333	605,184	426	106,878	1,078	0.7168	0.0100	0.0039
0.2861	1	10	0.9091	605,185	425	106,888	1,068	0.7153	0.0099	0.0039
0.2862	5	6	0.5455	605,190	420	106,894	1,062	0.7166	0.0098	0.0039
⋮	⋮	⋮	⋮		⋮	⋮	⋮	⋮	⋮	⋮
0.3097	0	1	1.0000	605,610	0	107,953	3	1.0000	0.0000	0.0000
0.3108	0	1	1.0000	605,610	0	107,954	2	1.0000	0.0000	0.0000
0.3111	0	2	1.0000	605,610	0	107,956	0	nan	0.0000	0.0000

TP we would have if we chose θ to be that value of p. The number of negative and positive samples are constant at $N = 605,610$ and $P = 107,956$. TN is the cumulative sum of Neg, and FP = N − TN; similarly for FN and TP.

In Table 4, we have shown the results for four bands of p. The first and fourth are the beginning and end of the range of p. The second range shows where FP/P ≈ 0.05, the first budgetary decision metric. The third range illustrates that while Precision generally increases from 0.15 (the class ratio) to 1, it is not perfectly nondecreasing, even with p rounded to four decimal places. Note in the fourth column, mProb, the values generally increase from 0 to 1 but not in a smooth way. We have to significantly smooth that metric to find the value of p where it equals 50%. The full table is at `Analyze_Proba/RFC_Hard_Run_0_Round_4_Rolling_Intervals`.

Our **first budgetary decision metric**, FP/P, is a nonincreasing function of p because P is constant and FP is P minus the cumulative sum of nonnegative integers Neg. Because FP/P is a nonincreasing function of p, we can choose θ to be the value of p where FP/P is closest to 0.05. For this run of the Random Forest Classifier with the hard features we can choose $\theta = 0.2580$, and a second run with a different random seed gave $\theta = 0.2583$, so we can say with some confidence that we should choose $\theta = 0.258$ for this model.

Table 4 shows that our **second budgetary decision metric**, Precision = TP/(FP + TP), generally increases from 0.15 to 1 (0.15 is the class ratio P/(N + P)), but is not an increasing function of p at that level of granularity.

To go from one p to the next, or from one band of p to the next,

$$\frac{\Delta \text{TP}}{\Delta p} = -\text{Neg} \qquad \text{and} \qquad \frac{\Delta \text{FP}}{\Delta p} = -\text{Pos} \qquad (5)$$

For large changes in p, Prec(p) is an increasing function, but what do we mean by "large"? In Equation 6 we assume that Prec(p) increases on an interval of length Δp, then we derive a relationship that tells us the proportion of Neg to Pos we need in that interval:

$$\text{Prec}(p) < \text{Prec}(p + \Delta p)$$

$$\frac{\text{TP}}{\text{FP} + \text{TP}} < \frac{\text{TP} - \text{Neg}}{(\text{FP} - \text{Pos}) + (\text{TP} - \text{Neg})}$$

$$\vdots \tag{6}$$

$$\text{FP} \cdot \text{Pos} < \text{TP} \cdot \text{Neg}$$

$$\frac{\text{FP}}{\text{TP}} < \frac{\text{Neg}}{\text{Pos}}$$

Having no elements of the negative class in a p-interval would make precision decrease, so we cut the p-values into bands with some minimum number of elements of each class. For each model, the `Keras/Analyze_Proba` directory has a spreadsheet like Table 5 for minimum number of class elements 5, 10, 25, 50, 100, 200, 400, 800, 1600, 3200, and 6400. An opportunity for future research (§5) is to develop a method to cut the p output into the smallest possible bands such that $\text{Prec}(p)$ is an increasing function.

In Table 5, we have cut p into bands such that each band has at least fifty elements of each of the negative and positive classes. The p values in the table are the average of the minimum and maximum values of p in the band. The full table is at `Analyze_Proba/RFC_Easy_Run_0_50_Slices.csv`.

At this level of granularity and in this range of p, the precision is an increasing function of p, and we can say that to get Precision = 2/3 we should set $\theta \approx 0.276$. Another run of the same model with different random seed gave $\theta \approx 0.279$.

Table 5

Metrics on p Output of Random Forest Classifier on the Hard Features with Minimum of 50 Elements of Each Class in Each Band. Table accompanies §2.5.1

p	Neg	Pos	mProb	TN	FP	FN	TP	Prec	Rec	FP/P
0.27565	50	51	0.505	604,454	1,156	105,702	2,254	0.661	0.0209	0.0107
0.27608	50	59	0.5413	604,504	1,106	105,761	2,195	0.665	0.0203	0.0102
0.27667	50	79	0.6124	604,554	1,056	105,840	2,116	0.6671	0.0196	0.0098
0.27732	56	50	0.4717	604,610	1,000	105,890	2,066	0.6738	0.0191	0.0093
0.27798	50	80	0.6154	604,660	950	105,970	1,986	0.6764	0.0184	0.0088
0.27869	50	67	0.5726	604,710	900	106,037	1,919	0.6807	0.0178	0.0083

For the the **third budgetary decision criterion**, that the minimum probability that each immediately dispatched ambulance is needed is at least 50%, with the metric mProb = Pos/(Neg + Pos) $\geq$ 0.50, equivalent to Pos $\geq$ Neg.

Table 6 shows that, if we zoom out to where each band has at least 6,400 elements in each class, cutting p into only seventeen intervals, mProb generally increases, but even at this high level the metric decreases between $p = 0.1606$ and $p = 0.1689$. This third metric is highly volatile.

Table 6

Metrics on p Output of Random Forest Classifier on the Hard Features with Minimum of 6400 Elements of Each Class in Each Band. Table accompanies §2.5.1

p	Neg	Pos	mProb	TN	FP	FN	TP	Prec	Rec	FP/P
0.1166	132,612	5,396	0.0391	132,612	472,998	5,396	102,560	0.1782	0.95	4.3814
0.1293	79,551	6,400	0.0745	212,163	393,447	11,796	96,160	0.1964	0.8907	3.6445
0.1336	57,673	6,400	0.0999	269,836	335,774	18,196	89,760	0.2109	0.8314	3.1103
0.137	42,946	6,400	0.1297	312,782	292,828	24,596	83,360	0.2216	0.7722	2.7125
0.1399	36,922	6,400	0.1477	349,704	255,906	30,996	76,960	0.2312	0.7129	2.3705
0.1432	35,947	6,400	0.1511	385,651	219,959	37,396	70,560	0.2429	0.6536	2.0375
0.1469	32,431	6,400	0.1648	418,082	187,528	43,796	64,160	0.2549	0.5943	1.7371
0.1509	30,786	6,400	0.1721	448,868	156,742	50,196	57,760	0.2693	0.535	1.4519
0.1553	25,477	6,401	0.2008	474,345	131,265	56,597	51,359	0.2812	0.4757	1.2159
0.1606	23,300	6,400	0.2155	497,645	107,965	62,997	44,959	0.294	0.4165	1.0001
0.1689	26,951	6,400	0.1919	524,596	81,014	69,397	38,559	0.3225	0.3572	0.7504
0.1806	21,160	6,400	0.2322	545,756	59,854	75,797	32,159	0.3495	0.2979	0.5544
0.1979	17,362	6,400	0.2693	563,118	42,492	82,197	25,759	0.3774	0.2386	0.3936
0.2199	16,065	6,400	0.2849	579,183	26,427	88,597	19,359	0.4228	0.1793	0.2448
0.2366	10,969	6,400	0.3685	590,152	15,458	94,997	12,959	0.456	0.12	0.1432
0.249	9,058	6,400	0.414	599,210	6,400	101,397	6,559	0.5061	0.0608	0.0593
0.2835	6,400	6,559	0.5061	605,610	0	107,956	0	nan	0	0

We tried two ways to estimate where $mProb = 0.50$, and both gave the same result.

Method 1: If we slice the p results into bands with at least 100 elements of each class, partly shown in Table 7, there is no band with $p < 0.2692$ with $mProb \geq 0.5$ and no band with $p > 0.2733$ with $mProb \leq 0.5$, so we can say with some confidence that if we chose $\theta \approx 0.271$, each ambulance immediately dispatched has at least a 50% chance of being needed.

Table 7

Metrics on *p* Output of Random Forest Classifier on the Hard Features with Minimum of 100 Elements of Each Class in Each Band. Table accompanies §2.5.1

p	Neg	Pos	*m*Prob	TN	FP	FN	TP	Prec	Rec	FP/P
0.2692	118	100	0.4587	603,564	2,046	104,738	3,218	0.6113	0.0298	0.019
0.2698	100	101	0.5025	603,664	1,946	104,839	3,117	0.6156	0.0289	0.018
0.2704	127	100	0.4405	603,791	1,819	104,939	3,017	0.6239	0.0279	0.0168
0.2711	108	100	0.4808	603,899	1,711	105,039	2,917	0.6303	0.027	0.0158
0.2718	111	100	0.4739	604,010	1,600	105,139	2,817	0.6378	0.0261	0.0148
0.2725	100	109	0.5215	604,110	1,500	105,248	2,708	0.6435	0.0251	0.0139
0.2733	100	149	0.5984	604,210	1,400	105,397	2,559	0.6464	0.0237	0.013

Method 2: We used a rolling sum of the Neg and Pos to intervals with enough Neg and Pos to smooth out the metric *m*Prob. With *p* rounded to four decimal places we still have volatility at each value of *p*, and the metric gets close to the target value (0.50) many times over a large range. If we take the rolling average of the ten or twenty rows centered at that value of *p*, we see less volatility, but even in this small interval the *m*Prob increases and decreases. With a window size of 20, however, the only values of *p* where the metric is close to the target are in this range, so we can choose $\theta = 0.272$, and our two methods agree. The full chart, up to a window of size 2,000, is in `Analyze_Proba/RFC_Hard_Run_0_Round_4_Rolling_Intervals`.

Table 8

Smoothing *m*Prob with Rolling Sums of Different Window Sizes. Table accompanies §2.5.1

Window Size	1			10			20			50		
p	Neg	Pos	*m*Prob	Neg	Pos	*m*Prob	Neg	Pos	*m*Prob	Neg	Pos	*m*Prob
0.2717	14	22	0.6111	144	166	0.4645	303	320	0.4864	754	771	0.4944
0.2718	12	18	0.6	153	159	0.4904	310	315	0.496	754	768	0.4954
0.2719	14	16	0.5333	143	164	0.4658	307	323	0.4873	758	757	0.5003
0.2720	12	13	0.52	152	156	0.4935	300	323	0.4815	756	745	0.5037
0.2721	26	13	0.3333	161	150	0.5177	296	312	0.4868	768	737	0.5103
0.2722	10	22	0.6875	159	149	0.5162	306	296	0.5083	755	726	0.5098
0.2723	17	7	0.2917	159	147	0.5196	309	291	0.515	756	717	0.5132
Range of *p* where $|mProb - 0.50| < 0.01$												
	min(*p*)	0.2472		min(*p*)	0.2718		min(*p*)	0.2718		min(*p*)	0.2711	
	max(*p*)	0.3076		max(*p*)	0.2726		max(*p*)	0.2722		max(*p*)	0.2722	

2.5.2. Choosing the Best Model for each Budgetary Decision Metric

For each budgetary constraint we want to find the model that, within the constraint, immediately dispatches the most ambulances to crash persons who need them, which is given as a proportion in Recall = TP/P. Using as an example our first budgetary constraint, FP/P = 0.05, we need to find, for each model whether there exists a value of *p* where FP/P is close to 0.05, then find the best θ interval in that neighborhood. Of those valid results, find the model that gives the highest Recall.

Table 9 shows the best results for each model algorithm. Within these, the Balanced Random Forest Classifier gives the best results, immediately dispatching the largest proportion (11.67%) of needed ambulances while staying within the budgetary constraint. The KerasClassifier with the Binary Focal Crossentropy loss function is a close second, and those two are clearly better than the other six models.

The Balanced Bagging model here is an interesting case that we cover in the next section, §2.5.3.

Table 9

Comparing Models: Best results for each algorithm on the Hard features for budgetary criterion FP/P closest to 0.05. Table accompanies §2.5.2

Algorithm	α	γ	p	TN	FP	FN	TP	Recall	FP/P
Bal RF	0.5		0.86	600,464	5,146	95,359	12,597	0.1167	0.048
Keras	0.5	0.0	0.598	600,212	5,398	97,029	10,927	0.1012	0.05
Log Reg	0.5		0.549	600,212	5,398	100,764	7,192	0.0666	0.05
RUSBoost			0.5	600,212	5,398	101,041	6,915	0.0641	0.05
AdaBoost			0.501	600,212	5,398	101,131	6,825	0.0632	0.05
Bal Bag			0.9	602,185	3,425	101,272	6,684	0.0619	0.032
RF			0.258	600,212	5,398	102,135	5,821	0.0539	0.05
Easy Ens			0.549	600,155	5,455	102,322	5,634	0.0522	0.051

2.5.3. Detecting Model Failure

We have found two ways a model can fail to give a decision threshold θ that satisfies the budgetary criteria. The second one we consider is the usual problem, that the model does not sufficiently separate the positive and negative classes to give enough correct recommendations. The other cause is more subtle, and we consider it first because we just saw it in Table 9, that for the Balanced Bagging algorithm, the value of FP/P closest to 0.05 (FP/P = 0.032) is not close to 0.05, and the cause of the failure is the numerics in the algorithm.

The RUSBoost algorithm on the hard features only gives values of p in a tiny range, $p \in [0.499, 0.5011]$, but in that range of 00021 the 713,566 samples return 706,938 different values of p. That's about as close to "continuous" as a discrete function can get. Digging down into the results returned by the Balanced Bagging algorithm on the hard features, 99% of the values of p are rounded to the nearest tenth, making the results extremely discrete. In Table 10, going from $p = 0.897$ to $p = 0.9$ we add 19,052 new samples that change the FP/P value from 0.11 to 0.03. The model results give no way to find a budgetary decision threshold θ that gives us a 5% increase in the number of ambulances dispatched to automated notifications from cell phones. This model may have too much volatility to be useful.

The Balanced Random Forest algorithm on the hard features has a similar quirk, that 93% of its p values are rounded to the nearest hundredth, which explains the FP/P = 0.048 as the closest to 0.05 in Table 9. These two algorithms, Bal Bag and Bal RF, interestingly, had more detail in the Medium Features results and even more detail for the Easy features, which is the opposite of what most of the other algorithms did. The AdaBoost, Keras, Log Reg, and RUSBoost all had, from the 713,566 samples, about 700,000 unique values of p for the hard features, 650,000 for the medium features, and 150,000 for the easy features. A breakdown of those numbers is given in `Value_Counts_y_proba.csv` in the `Keras/Analyze_Proba` folder, and the value counts of p are given for each algorithm in files such as `BalBag_Hard_Value_Counts.csv`.

Table 10
Model Failure: Balanced Bagging Algorithm on the Hard Features with FP/P closest to 0.05. Table accompanies §2.5.3

p	Neg	Pos	TN	FP	FN	TP	FP/P
0.896	1	0	593,725	11,885	90,679	17,277	0.110091
0.897	1	0	593,726	11,884	90,679	17,277	0.110082
0.9	8,459	10,593	602,185	3,425	101,272	6,684	0.031726
0.908	1	1	602,186	3,424	101,273	6,683	0.031717
0.913	1	0	602,187	3,423	101,273	6,683	0.031707

The second kind of model failure is the usual problem with machine learning models, that the algorithm is not successful in detecting enough of a pattern in the data to build a robust model that separates the positive and negative classes well. There are two likely reasons for the failure. One is that the particular algorithm with its hyperparameters is not suited to the kind of patterns in the data, and it is for this reason that we have tried several models with different class and focal weights. A second possible reason is that there just isn't enough of a pattern in the dataset, and it is for this reason that we have tested the Easy, Medium, and Hard feature sets.

As an example of this kind of model failure, consider the Random Forest Classifier on the Medium features, shown in Table 11. Here the p values are cut into bands with at least one hundred elements of each of the Neg and Pos classes; the p values in the first column are the average of the minimum and maximum p value in the band.

The Precision values generally increase with p, and the table shows the end of the p range; there are no values of Precision close to 2/3, not because it doesn't theoretically exist, but the model is not robust enough to sufficiently separate the positive and negative classes. If the governmental leaders decided to change to Precision = 1/2, we could choose $\theta = 0.240$, but of the crash persons with an automated notification from a cell phone who needed an ambulance, we would only be immediately dispatching ambulances to Recall = 1% of them. Scaling that to a medium-sized city, the administrative overhead of the program might outweigh any benefit. Given a finite budget, a responsible government needs to choose the most cost-effective programs, and an immediate dispatch recommendation system might not be one of them.

We can see the problem also in a histogram of the model output, in Figure 10, where we compare the right tails of the output of the Random Forest model on the Medium features with the Example model from Figure 4. The model output has a long tail to the right, and zooming in on that tail, we see that in none of those intervals are there more elements of the positive class than the negative class.

Note that this range of p in Table 11 is also where Pos/(Neg + Pos) crosses 0.50, meaning that there is a 50% marginal probability that the last ambulance sent is needed, so we would have the same problem for that metric, that the model would recommend immediately dispatching ambulances to only about 1% of the automated notifications that needed an ambulance.

Fortunately we do have models with better results on the Medium features set.

Table 11

Metrics on Partial p Output of Random Forest Classifier on the Medium Features with Minimum of 400 Elements of Each Class in Each Band. Table accompanies §2.5.3

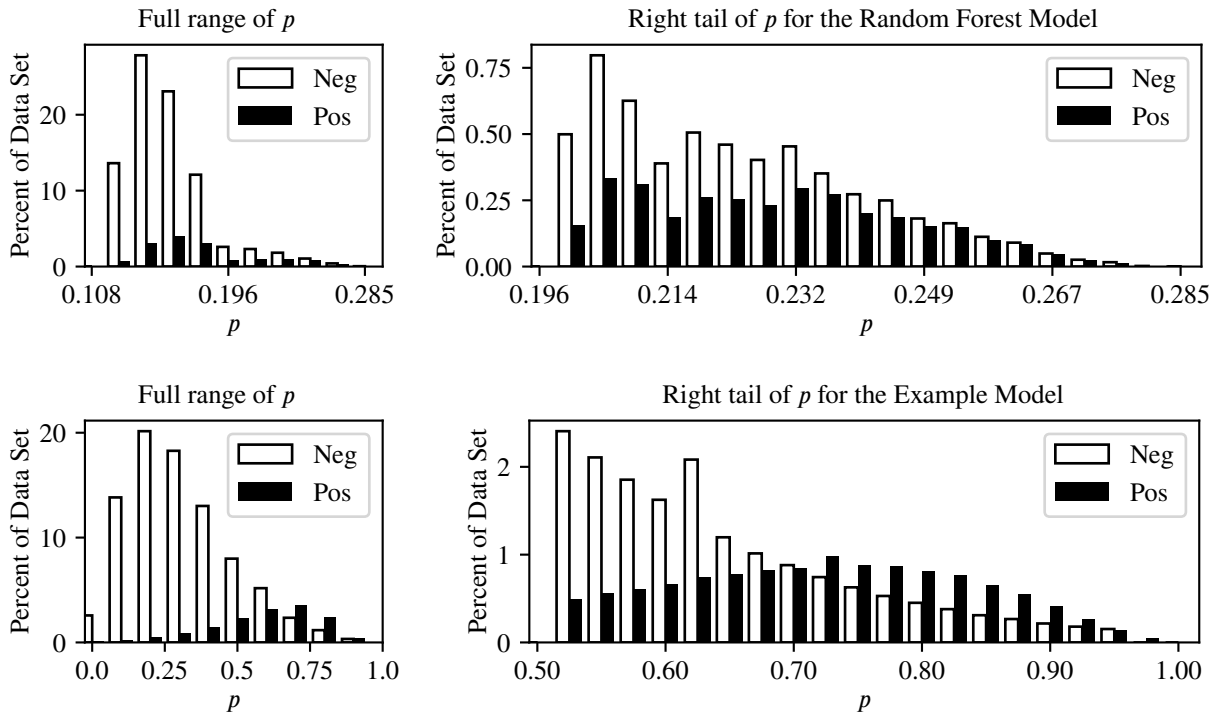| p	Neg	Pos	mProb	TN	FP	FN	TP	Prec	Rec	FP/P
0.2386	415	401	0.491	603,883	1,727	106,283	1,673	0.492	0.016	0.016
0.2405	460	400	0.465	604,343	1,267	106,683	1,273	0.501	0.012	0.0117
0.2429	466	400	0.462	604,809	801	107,083	873	0.522	0.008	0.0074
0.2465	400	421	0.513	605,209	401	107,504	452	0.53	0.004	0.0037
0.2599	401	452	0.53	605,610	0	107,956	0	nan	0	0

Figure 10: Random Forest Classifier on the Medium Features (top) and Example model (bottom). Figure accompanies §2.5.3

2.5.4. Numerical Precision

We ran all of the models twice with different random seeds. As an example of the difference in model results, consider the first budgetary decision metric, the values of p that make FP/P closest to 0.05, because that metric is the most stable. If there is a difference in those results between two runs of the same model for FP/P, not only can we only claim that much numerical precision in reporting our results for that metric, but we may need to limit ourselves to even less numerical precision in reporting results for the other budgetary decision metrics.

Table 12 shows the models with the largest changes between runs. The median Δp was 0.0003, the median Δ TP was 57, and the median Δ FP was 1, giving median Δ FP/P of 0.0000093.

Table 12
Comparison of values of p, TP, FP, and FP/P between Runs of the Same Model with Different Random Seeds. Table accompanies §2.5.3

Model	Features	Run	p	Neg	Pos	FP	TP	FP/P	Notes
Keras $\alpha = 0.5, \gamma = 1.0$	Medium	0	0.50556	1	0	5,398	6,294	0.05	$\Delta p = 0.012$
Keras $\alpha = 0.5, \gamma = 1.0$	Medium	1	0.51764	1	0	5,398	6,105	0.05	
Keras $\alpha = 0.5, \gamma = 1.0$	Hard	0	0.56429	1	0	5,398	10,946	0.05	ΔTP = 409
Keras $\alpha = 0.5, \gamma = 1.0$	Hard	1	0.55914	1	0	5,398	10,537	0.05	
Easy Ens	Easy	0	0.52846	263	107	5,286	2,682	0.049	ΔFP = 180
Easy Ens	Easy	1	0.52866	27	15	5,466	2,741	0.0506	ΔFP/P = 0.0016

3. Results

3.1. Data and Image Files

Results data and images are in the `Keras/Analyze_Proba` and `Keras/Images` folders at `www.github.com/bburkman/Ambulance_Dispatch`. The results data are in `.csv` format, and the images in both `.png` and `.pgf`.

The results of each (algorithm, hyperparameters, features, random seed) combination are in, for example,

> `LogReg_alpha_0_5_Easy_Run_0.csv` with p in its full precision,
> `LogReg_alpha_0_5_Easy_Run_0_Round_4.csv` with p rounded and grouped,
> `LogReg_alpha_0_5_Easy_Run_0_Rolling_Intervals.csv`,
> `LogReg_alpha_0_5_Easy_Run_0_Round_4_Rolling_Intervals.csv`,
> `LogReg_alpha_0_5_Easy_Run_0_Value_Counts_head_100.csv`
>   with the hundred most frequent values of p,
> `LogReg_alpha_0_5_Easy_Run_0_{0,5,10,25,50,100,200,...,6400}_Slices.csv`
>   that each have p cut into intervals with at least n samples of each of the two classes.

For each (model, hyperparameters, features set, random seed) combination there are five images, for instance,

> `BRFC_alpha_0_85_Easy_Run_0_Pred`,
> `BRFC_alpha_0_85_Easy_Run_0_Pred_Wide`,
> `BRFC_alpha_0_85_Easy_Run_0_Pred_Zoom`,
> `BRFC_alpha_0_85_Easy_Run_0_Pred_Zoom_Wide`, and
> `BRFC_alpha_0_85_Easy_Run_0_ROC`,

and each image comes in a `.png` and `.pgf` version. The `Wide` pictures are 4.5 inches wide, and the others 2.0 inches. The `Pred` and `Pred_Wide` images show p from 0 to 1 as in Figure 7, and the `Zoom` images from the minimum to maximum values of p, as in Figure 9.

These summary tables, all `.csv` files, have one row for each (model, hyperparameters, features set, random seed) combination:

`FP_P_0_05`	Value of p with FP/P closest to 0.05
`FP_P_0_10`	Value of p with FP/P closest to 0.10
`Prec_Rolling_0_667`	For each model and each window length in {1, 10, 20, 50, 100, 200, 500, 1000, 2000}, the first and last value of p for which Precision is within 0.01 of 0.667, if such p exist for that window length.
`mProb_Rolling_0_500`	Same for marginal probability within 0.01 of 0.50
`ROC_AUC`	Area Under the Curve of the Receiver Operating Characteristic curve

3.2. Easy, Medium, and Hard Features

Figure 11 shows the raw model output and ROC curves for the Balanced Random Forest Classifier model trained on the Easy, Medium, and Hard feature sets. All three have significant overlap of the negative (does not need an ambulance) and positive (needs an ambulance) classes, but comparing the Hard to the Easy, the negative class in the Hard is pushed more to the left and the positive more to the right, better separating the two classes and giving a space (up to about $p = 0.8$) to choose a decision threshold θ that will send more needed than unneeded ambulances.

The ROC (Receiver Operating Characteristic) curves illustrate how well the models separate the positive and negative classes, with area under the curve (AUC) of 0.5, shown by the dashed line, being no better than random and 1.0 being perfect separation. The ROC AUC also support the idea that the models built on the Hard features set are much more predictive than the Medium, which are better than the Easy.

The thirteen models built on the Easy feature set had AUC in $[0.552, 0.665]$, Medium in $[0.640, 0.729]$, and Hard in $[0.708, 0.801]$. If we take out the worst Hard model and the worst Medium model, the ranges do not overlap.
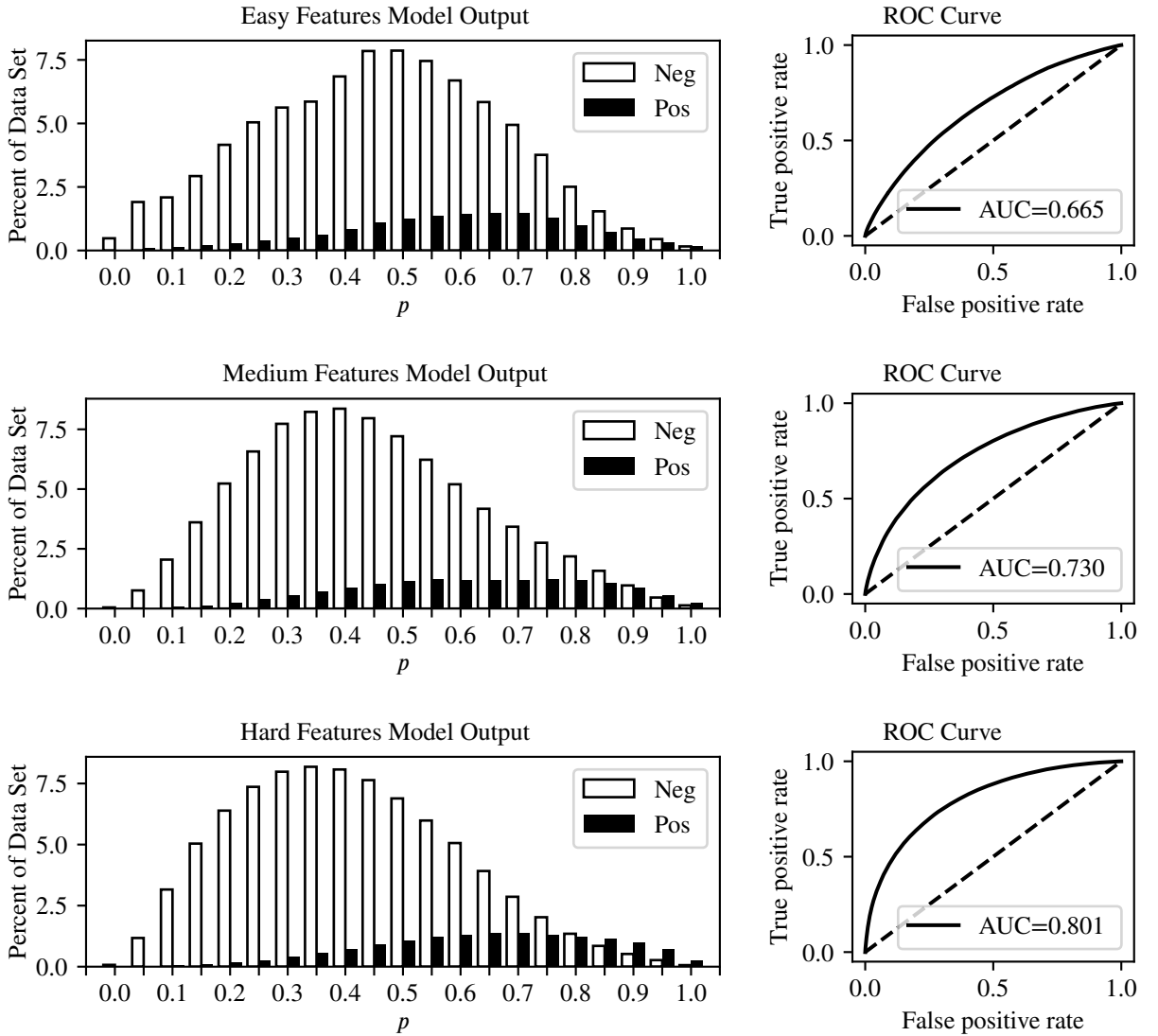

Figure 11: Balanced Random Forest Classifier with $\alpha = 0.5$ Model Results for Different Sets of Data Features. Figure accompanies §3.2

Table 13
Best models and transformations for FP/P = 0.05 for each algorithm. Table accompanies §3.3

Model	Features	p	TN	FP	FN	TP	FP/P	Recall
Bal RF, $\alpha = 0.5$	Hard	0.86	600,519	5,091	95,468	12,488	0.0472	0.1157
Keras, $\alpha = 0.5$, $\gamma = 1.0$	Hard	0.5643	600,212	5,398	97,010	10,946	0.05	0.1014
Log Reg, $\alpha = 0.5$	Hard	0.5477	600,212	5,398	100,715	7,241	0.05	0.0671
Bal RF, $\alpha = 0.5$	Medium	0.89	600,288	5,322	101,267	6,689	0.0493	0.062
Keras, $\alpha = 0.5$, $\gamma = 0.0$	Medium	0.5117	600,212	5,398	101,643	6,313	0.05	0.0585
RF	Medium	0.2581	600,212	5,398	102,135	5,821	0.05	0.0539
Bal RF, $\alpha = 0.5$	Easy	0.8887	600,213	5,397	103,717	4,239	0.05	0.0393
Bal Bag	Easy	0.9	600,476	5,134	104,345	3,611	0.0476	0.0334
Keras, $\alpha = 0.5$, $\gamma = 1.0$	Easy	0.4193	600,217	5,393	104,763	3,193	0.05	0.0296

3.3. Best Model for Each Budgetary Criterion and Feature Set

Our **first budgetary decision criterion** was that the increase in number of ambulances sent to crash persons with automated notifications from cell phones should be not more than 5%, with metric FP/P ≤ 0.05. Table 13 gives, for each feature set (Easy, Medium, Hard) the best models, with "best" meaning that the model recommends sending the most ambulances that are actually needed (TP) subject to the constraint that we choose our decision threshold θ to be the value of p such that FP/P is closest to 0.05. In Table 13 we have included the best three models for each set of data features. The numbers for the Hard features vary slightly from Table 9 because the run used a different random seed.

We have not here included similar results, like the Balanced Random Forest models with different hyperparameters, because the data shows that the differences between the results of very similar models is more about the numerics and randomness than predictive of what a similar model on a slightly different dataset with different random seeds would produce. For full results see `Keras/Analyze_Proba/FP_P_0_05.csv`.

Our **second budgetary decision criterion** was that, of the ambulances immediately dispatched, at least 2/3 should be actually needed. Table 14 gives, for each feature set the best models, with "best" meaning that the model recommends sending the most ambulances that are actually needed (TP) subject to the constraint that we choose our decision threshold θ to be the value of p such that Precision = TP/(FP + TP) is closest to 0.667 when Precision converges in rolling averages of FP and TP of increasing window size. For full results see `Keras/Analyze_Proba/Prec_Rolling_0_667.csv`.

The Hard features give robust models that immediately dispatch up to 19% of the needed ambulances while staying within the budgetary decision criterion, but the Medium models are bad and the Easy worthless. Most of the thirteen Easy models do not even get close to finding a p where Precision is 2/3. Governments that wanted to consider models built on the Easy features could consider Precision of 1/2 as an attainable target.

Table 14

Best models and transformations for Precision = TP/(FP + TP) = 2/3 for each algorithm. Table accompanies §3.3

Model	Features	p	TN	FP	FN	TP	Prec	Recall
Bal RF, $\alpha = 0.5$	Hard	0.8199	594,922	10,688	87,467	20,489	0.6572	0.1898
Keras, $\alpha = 0.5, \gamma = 1.0$	Hard	0.5631	599,135	6,475	95,560	12,396	0.6571	0.1148
Bal Bag	Hard	0.9567	602,204	3,406	101,142	6,814	0.6666	0.0631
Bal RF, $\alpha = 0.5$	Medium	0.9671	604,992	618	106,770	1,186	0.6574	0.011
Keras, $\alpha = 0.5, \gamma = 0.0$	Medium	0.749	605,478	132	107,705	251	0.6571	0.0023
Log Reg, $\alpha = 0.5$	Medium	0.727	605,596	14	107,927	29	0.6574	0.0003
AdaBoost	Easy	0.4997	605,608	2	107,952	4	0.6667	0
Keras, $\alpha = 0.5, \gamma = 1.0$	Easy	0.5445	605,608	2	107,952	4	0.6667	0
Log Reg, $\alpha = 0.5$	Easy	0.4708	605,609	1	107,954	2	0.6667	0

Our **third budgetary decision criterion** was that each ambulance immediately dispatched should have at least a 50% chance of being needed. Table 15 gives, for each feature set the best models, with "best" meaning that the model recommends sending the most ambulances that are actually needed (TP) subject to the constraint that we choose our decision threshold θ to be the value of p such that the marginal probability, $m\text{Prob} = \text{Pos}/(\text{Neg} + \text{Pos})$, is closest to 0.50 when mProb converges in rolling sums of Neg and Pos of increasing window size. For full results see `Keras/Analyze_Proba/mProb_Rolling_0_500.csv`.

Table 15
Best models for $m\text{Prob} = \text{Pos}/(\text{Neg} + \text{Pos}) \approx 0.5$ for each algorithm with length of rolling sum. Table accompanies §3.3

Model	Features	Roll	p	Neg	Pos	mProb	TP	Recall
Bal RF, $\alpha = 0.5$	Hard	50	0.5214	3,217	3,275	0.5045	23,732	0.2198
Keras, $\alpha = 0.5$, $\gamma = 1.0$	Hard	200	0.5415	2,446	2,535	0.5089	15,007	0.139
Log Reg, $\alpha = 0.5$	Hard	500	0.5005	2,518	2,613	0.5093	6,916	0.0641
RUSBoost	Hard	100	0.8858	3,980	4,142	0.51	5,529	0.0512
Bal RF, $\alpha = 0.5$	Medium	200	0.5006	2,266	2,339	0.5079	6,364	0.0589
BalBag, $\alpha = 0.85$	Medium	20	0.5195	4,727	4,864	0.5071	4,668	0.0432
Keras, $\alpha = 0.5$, $\gamma = 0.0$	Medium	500	0.5222	2,352	2,442	0.5094	4,409	0.0408
Bal RF, $\alpha = 0.5$	Easy	500	0.5522	1,610	1,601	0.4986	760	0.007
Keras, $\alpha = 0.85$, $\gamma = 0.0$	Easy	10	0.5174	12	12	0.5	33	0.0003

4. Conclusions and Discussion

4.1. Scenario Reprise

The (fictitious) city of Springfield is considering implementing immediate dispatch of ambulances to automated crash notifications from cell phones, and are considering an AI recommendation system to predict which crash persons are more likely to need an ambulance.

Of all of Springfield's crash notifications, 20% come first from automated crash notifications from cell phones. Only 10% percent of all crashes need an ambulance, but those that trigger automated notifications tend to be more serious, so of those, 15% need an ambulance. Of all of the ambulances dispatched, half go to crashes. Scaling the numbers out of 100 ambulances dispatched, 50 go to non-crashes, 50 go to crashes, and there are 450 crashes that do not need an ambulance. There are 100 automated crash notifications, and 15 of those need an ambulance. Of the 15 that need an ambulance, our AI recommendation system correctly labels some (TP), and they get more prompt medical attention, but it incorrectly labels others (FN) who still get an ambulance, but only after a later 911/119 call from an eyewitness. Of the 85 who do not need an ambulance, the AI recommendation system correctly will label most of them (TN), but incorrectly label some of them, and Springfield will send an unneeded ambulance that would not have been sent if they had waited for an eyewitness call.

		Prediction			Recommendation		Wait for Call from Eyewitness	Immediately Dispatch
		PN	PP					
Actual	N	TN	FP		Needs	No	Correct	Increased Cost
	P	FN	TP		Ambulance?	Yes	Normal Delay	Prompt Medical Help

Figure 12: Confusion matrix for ambulance dispatch. Figure accompanies §1.1

If Springfield built a recommendation system based on the Balanced Random Forest model with the Hard features, depending on which of the three budgetary decision criteria they chose, Springfield crash persons would see the following values for TN, FP, FN, and TP. With the first budgetary decision criterion, for each 50 ambulances dispatched, the model would recommend sending 1.7 sooner to crash persons who need them while sending an additional 0.7 not needed. Note that those numbers come with the caveat that getting instantaneous access to the Hard features would likely be expensive, if even possible.
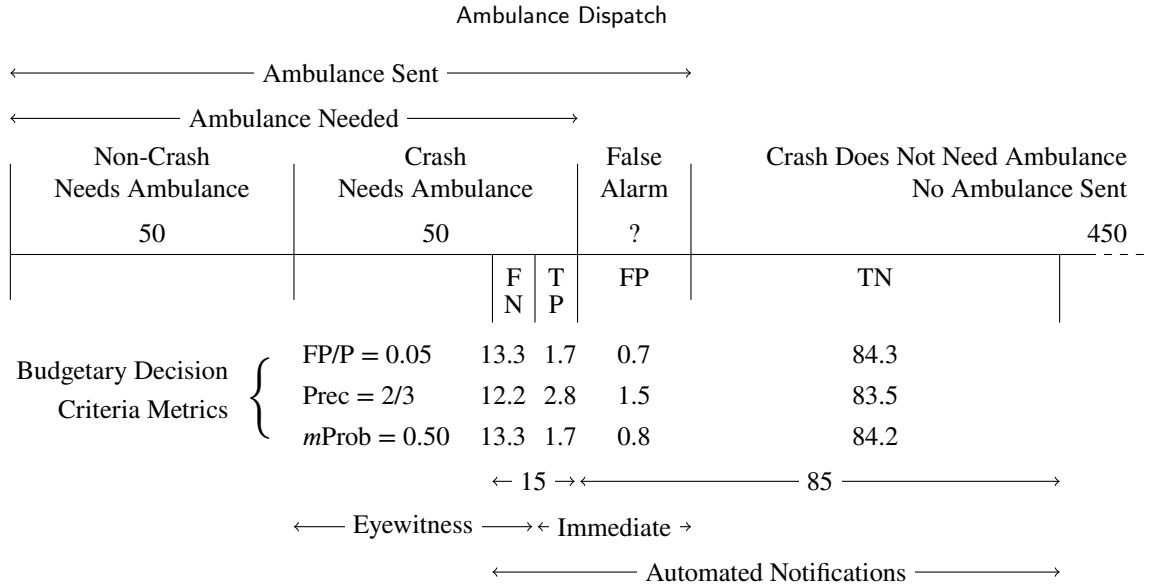
Ambulance Dispatch

	Non-Crash Needs Ambulance	Crash Needs Ambulance	False Alarm	Crash Does Not Need Ambulance No Ambulance Sent
	50	50	?	450
		FN / TP	FP	TN
Budgetary Decision Criteria Metrics	FP/P = 0.05	13.3 1.7	0.7	84.3
	Prec = 2/3	12.2 2.8	1.5	83.5
	mProb = 0.50	13.3 1.7	0.8	84.2

← Ambulance Sent →
← Ambulance Needed →
← 15 → ← 85 →
←— Eyewitness —→ ← Immediate →
←—— Automated Notifications ——→

Figure 13: Springfield after implementing immediate dispatch of ambulances. Figure accompanies §4.1

Should Springfield implement such an AI recommendation system for immediate dispatch of ambulances based on automated notifications from cell phones? If the only cost were the ambulances falsely dispatched (FP), then probably yes. Considering the fixed costs of implementing such a system, a smaller city would have other options of programs that can better improve health and save more lives with the same amount of money.

Springfield should consider changing the values of the metrics, perhaps to allow a FP/P = 10% increase in the number of ambulances sent to automated crash notifications. The best model on the Hard features, however, would give TP = 2.9 and FP = 1.5, more than doubling the cost (FP) without doubling the benefit (TP) from FP/P = 5% with TP = 1.7 and FP = 0.7. As we saw in our model output histograms, the returns diminish very quickly.

With better data and more appropriate model algorithms, of course, we might get models that do a better job of separating the negative and positive class, an opportunity for future research. (See §5)

4.2. Discussion

Choosing a tradeoff between saving lives and saving money is a fraught political question, but the decisions can be informed with models showing the likely outcomes of the possible choices.

Many cell phones can automatically notify the emergency dispatcher when they detect the deceleration profile of a crash. The dispatcher will immediately send police to investigate, but should they also immediately dispatch an ambulance, which has about a small chance of being needed, or wait for a call from an eyewitness? We have explored options for an AI recommendation system that would take the known information about the crash, a model built on historical crash data, and a decision criterion chosen by the political decision makers, and return a recommendation for whether to immediately dispatch an ambulance or to wait for a call from an eyewitness.

The main problem with such a recommendation system is that it will recommend sending some ambulances that are not needed. Budgets are limited, and priorities have to be chosen. We showed how to incorporate three kinds of decision criterion: Percent increase in number of ambulance calls, percent of immediately dispatched ambulances actually needed, and minimum probability that each immediately dispatched ambulance is needed.

We considered many challenges, including the randomness inherent in machine learning models and how the numerics severely limit the precision we can claim in our results. Another challenge is identifying with useful numerical precision the decision threshold θ that satisfies a given metric when the value of that metric is only stable after much smoothing. We found that finding θ for the first metric, a cap on increased ambulance runs, is straightforward, but Precision requires some smoothing, and marginal probability that an ambulance is needed is very unstable on small intervals.

The most significant challenge of implementing such a system is the cost and availability of the input data required to make a useful recommendation system. We considered three sets of input features that we called Easy, Medium,

and Hard, but that can also be thought of as Free, Expensive, and Problematic. Even through the randomness and numerics, the results show that the results of models built on the Hard features are clearly better than those built on the Medium features, which are clearly better than those built on the Easy features. The models built on the Easy features, however, are still better than random guessing, and a recommendation system built on just those features may be worth the expense, especially since the emergency dispatcher already has that information.

5. Simplifying Assumptions and Opportunities for Future Research

All models are simplifications, and we should acknowledge the most egregious of our simplifying assumptions. Some of the simplifying assumptions may hint at opportunities for future research.

Section	Simplifying Assumption
§2.1	All ambulances sent based on calls from eyewitnesses are actually needed.
	All automated crash notifications from cell phones refer to an actual crash, not just hard braking, *i.e.* the notifications have no false positives.
§2.2	The class ratio (P/N) in the automated crash notification from cell phones will be close to that in the CRSS data, 1/5.
§2.2	The crash persons in the CRSS data are representative of the future crash persons whose cell phones send a crash notification. (Several layers to unpack here)
§2.3	The data features we seek for each automated crash notification will be available and accurate.

Section	Opportunities for Future Research
§2.2	Find data on crashes that spawned an automated notification from a cell phone.
§2.3	We tested only three sets of features and did not test to see which features or combinations of features were most useful in predicting needing an ambulance.
§2.4.2	We found that varying the hyperparameters for class weight and focal loss did not significantly vary the ROC AUC, a measure of how well the model separates the positive and negative classes over the whole range $p \in [0, 1]$. Do those hyperparameters make a difference in how well the model separates the positive and negative classes on the right tail of the p distribution, the part relevant to our work?
§2.5.1	Find a better way to slice the p-values into intervals such that the target metrics are monotonic functions of the intervals.
§4.1	Can we find better combinations of data and model algorithms that will better separate the positive and negative classes?

Conflict of Interest

Declarations of interest: none

Data Availability

The CRSS data is publicly available at
`https://www.nhtsa.gov/crash-data-systems/crash-report-sampling-system`
All of the code and generated data, tables, and graphs are available at `http://www.github.com/bburkman/Ambulance_Dispatch`

CRediT authorship contribution statement

J. Bradford Burkman: Conceptualization, Investigation, Writing - original draft, Visualization. **Chee-Hung Henry Chu:** Supervision, Methodology, Writing - review and editing. **Miao Jin:** Supervision, Methodology. **Xiaoduan Sun:** Data curation, Writing - review and editing.

References

Bertsimas, D., Ng, Y., 2019. Robust and stochastic formulations for ambulance deployment and dispatch. European Journal of Operational Research 279, 557–571. URL: https://www.sciencedirect.com/science/article/pii/S0377221719304138, doi:https://doi.org/10.1016/j.ejor.2019.05.011.

Bose, D., Crandall, J., McGwin, G., Goldman, J., Foster, J., Fine, P., 2011. Computational methodology to predict injury risk for motor vehicle crash victims: A framework for improving advanced automatic crash notification systems. Transportation Research Part C: Emerging Technologies 19, 1048–1059. URL: https://www.sciencedirect.com/science/article/pii/S0968090X11000891, doi:https://doi.org/10.1016/j.trc.2011.06.001.

Candefjord, S., Sheikh Muhammad, A., Bangalore, P., Buendia, R., 2021. On scene injury severity prediction (osisp) machine learning algorithms for motor vehicle crash occupants in us. Journal of Transport & Health 22, 101124. URL: https://www.sciencedirect.com/science/article/pii/S2214140521001547, doi:https://doi.org/10.1016/j.jth.2021.101124.

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

Geuens, C., Dumortier, J., 2010. Mandatory implementation for in-vehicle ecall: Privacy compatible? Computer Law & Security Review 26, 385–390. URL: https://www.sciencedirect.com/science/article/pii/S0267364910000580, doi:https://doi.org/10.1016/j.clsr.2010.03.009.

He, K., Zhang, P., Wang, S.C., 2019. Crash telemetry-based injury severity prediction is equivalent to or out-performs field protocols in triage of planar vehicle collisions. Prehospital & Disaster Medicine 34, 356 – 362. URL: https://ezproxyprod.ucs.louisiana.edu:2443/login?url=https://search.ebscohost.com/login.aspx?direct=true&AuthType=ip,cookie,uid,url&db=cul&AN=137931739&site=eds-live&scope=site.

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

Jagtenberg, C., Bhulai, S., van der Mei, R., 2015. An efficient heuristic for real-time ambulance redeployment. Operations Research for Health Care 4, 27–35. URL: https://www.sciencedirect.com/science/article/pii/S2211692314200075, doi:https://doi.org/10.1016/j.orhc.2015.01.001.

Kong, J.S., Lee, K.H., Kim, O.H., Lee, H.Y., Kang, C.Y., Choi, D., Kim, S.C., Jeong, H., Kang, D.R., Sung, T.E., 2023. Machine learning-based injury severity prediction of level 1 trauma center enrolled patients associated with car-to-car crashes in korea. Computers in Biology and Medicine 153, 106393. URL: https://www.sciencedirect.com/science/article/pii/S0010482522011015, doi:https://doi.org/10.1016/j.compbiomed.2022.106393.

Kononen, D.W., Flannagan, C.A., Wang, S.C., 2011. Identification and validation of a logistic regression model for predicting serious injuries associated with motor vehicle crashes. Accident Analysis & Prevention 43, 112–122. URL: https://www.sciencedirect.com/science/article/pii/S000145751000206X, doi:https://doi.org/10.1016/j.aap.2010.07.018.

Lemaître, G., Nogueira, F., Aridas, C.K., 2017. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. Journal of Machine Learning Research 18, 1–5. URL: http://jmlr.org/papers/v18/16-365.html.

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.

Liu, L., Racz, D., Vaillancourt, K., Michelman, J., Barnes, M., Mellem, S., Eastham, P., Green, B., Armstrong, C., Bal, R., O'Banion, S., Guo, F., 2023. Smartphone-based hard-braking event detection at scale for road safety services. Transportation Research Part C: Emerging Technologies 146, 103949. URL: https://www.sciencedirect.com/science/article/pii/S0968090X2200362X, doi:https://doi.org/10.1016/j.trc.2022.103949.

Liu, Y., Li, Z., Liu, J., Patel, H., 2016. A double standard model for allocating limited emergency medical service vehicle resources ensuring service reliability. Transportation Research Part C: Emerging Technologies 69, 120–133. URL: https://www.sciencedirect.com/science/article/pii/S0968090X16300602, doi:https://doi.org/10.1016/j.trc.2016.05.023.

Nadar, R.A., Jha, J., Thakkar, J.J., 2023. Adaptive variable neighbourhood search approach for time-dependent joint location and dispatching problem in a multi-tier ambulance system. Computers & Operations Research 159, 106355. URL: https://www.sciencedirect.com/science/article/pii/S0305054823002198, doi:https://doi.org/10.1016/j.cor.2023.106355.

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.

Plevin, R.E., Kaufman, R., Fraade-Blanar, L., Bulger, E.M., 2017. Evaluating the potential benefits of advanced automatic crash notification. Prehospital & Disaster Medicine 32, 156 – 164. URL: https://ezproxyprod.ucs.louisiana.edu:2443/login?url=https://search.ebscohost.com/login.aspx?direct=true&AuthType=ip,cookie,uid,url&db=cul&AN=158577825&site=eds-live&scope=site.

Qin, X., Khan, A.M., 2012. Control strategies of traffic signal timing transition for emergency vehicle preemption. Transportation Research Part C: Emerging Technologies 25, 1–17. URL: https://www.sciencedirect.com/science/article/pii/S0968090X1200054X, doi:https://doi.org/10.1016/j.trc.2012.04.004.

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

Su, H., Zhong, Y.D., Chow, J.Y., Dey, B., Jin, L., 2023. Emvlight: A multi-agent reinforcement learning framework for an emergency vehicle decentralized routing and traffic signal control system. Transportation Research Part C: Emerging Technologies 146, 103955. URL: https://www.sciencedirect.com/science/article/pii/S0968090X22003680, doi:https://doi.org/10.1016/j.trc.2022.103955.

Weaver, A.A., Talton, J.W., Barnard, R.T., Gaffley, M., Doud, A.N., Schoell, S.L., Petty, J.K., Martin, R.S., Meredith, J.W., Stitzel, J.D., 2022. Advanced automatic crash notification algorithm for children. Academic Pediatrics 22, 1057–1064. URL: https://www.sciencedirect.com/science/article/pii/S1876285922000845, doi:https://doi.org/10.1016/j.acap.2022.02.016.

Zhang, Z., He, Q., Gou, J., Li, X., 2016. Performance measure for reliable travel time of emergency vehicles. Transportation Research Part C: Emerging Technologies 65, 97–110. URL: https://www.sciencedirect.com/science/article/pii/S0968090X16000346, doi:https://doi.org/10.1016/j.trc.2016.01.015.