COST-SENSITIVE MULTI-CLASS ADABOOST FOR UNDERSTANDING DRIVING BEHAVIOR BASED ON TELEMATICS

BY

BANGHEE SO, JEAN-PHILIPPE BOUCHER AND EMILIANO A. VALDEZ

ABSTRACT

Using telematics technology, insurers are able to capture a wide range of data to better decode driver behavior, such as distance traveled and how drivers brake, accelerate, or make turns. Such additional information also helps insurers improve risk assessments for usage-based insurance, a recent industry innovation. In this article, we explore the integration of telematics information into a classification model to determine driver heterogeneity. For motor insurance during a policy year, we typically observe a large proportion of drivers with zero accidents, a lower proportion with exactly one accident, and a far lower proportion with two or more accidents. We here introduce a cost-sensitive multi-class adaptive boosting (AdaBoost) algorithm we call `SAMME.C2` to handle such class imbalances. We calibrate the algorithm using empirical data collected from a telematics program in Canada and demonstrate an improved assessment of driving behavior using telematics compared with traditional risk variables. Using suitable performance metrics, we show that our algorithm outperforms other learning models designed to handle class imbalances.

KEYWORDS

Vehicle telematics, usage-based insurance, cost-sensitive learning, AdaBoost, `SMOTE`, `SAMME`, `SAMME.C2`.

JEL codes: C10, C51, C52

1. INTRODUCTION

Telematics refers to the use of telecommunication devices and technology to transmit and store information. Currently, there is a growing list of applications of telematics technology in diverse fields. Examples include radio receivers installed in drones for journalism reporting or private investigation;

Astin Bulletin 51(3), 719-751. doi:10.1017/asb.2021.22 © The Author(s), 2021. Published by Cambridge University Press on behalf of The International Actuarial Association.

smart home systems that remotely control temperature, lighting, appliances, or even alarms for security; electronic systems used for better communication between healthcare professionals and patients in health telematics (Orphanoudakis *et al.* (1998)); and, predominantly, devices installed in cars or mobile apps to remotely monitor driving habits to determine insurance rating.

In the insurance industry, the use of telematics-based insurance is slowly maturing. It is also becoming an attractive option for drivers, as it gives them the opportunity to save on premiums in exchange for permission to monitor their driving behavior. Accelerations in technology advancement have led to variations in usage-based insurance (UBI) with similar self-descriptive names that include, for example, pay as you drive (PAYD), pay how you drive (PHYD), pay as you drive as you save (PAYDAYS), pay per mile, and pay as you go (PASG). UBI programs offer a number of potential benefits to consumers, insurers, and society. For one, vehicle telematics has the positive social effect of encouraging better driving behavior. It also allows insurers to put a price tag on insurance that is more directly linked to individual driving habits, helping to increase the predictability of profit margins and giving drivers the opportunity for more affordable premiums. Consumers can also control premium costs by adopting safer driving habits or reducing their driving frequency. UBI programs also benefit society, as safer driving and having fewer drivers on the road reduce accidents, congestion, and car emissions.

The growing literature on telematics in actuarial science and insurance has produced a variety of evidence that this additional acquired information allows for better claim prediction, risk classification, and premium assessments. Interestingly, the work by Ayuso *et al.* (2016) uses survival models to conclude that gender discrimination is unnecessary in the presence of sufficient telematics information on driving behavior. Boucher *et al.* (2017) suggests the benefits of using generalized additive models (GAM) to gain additional insight into to how premiums can be more dynamically assessed for PAYD policies based on telematics information. For tariff determination, Ayuso *et al.* (2019) uses classical frequency models to incorporate significant information drawn from telematics metrics based on data from a portfolio of PAYD policies issued by a Spanish insurance company. In addition, Gao *et al.* (2019) demonstrates the relevance of telematics covariates extracted from speed-acceleration heatmaps in claim frequency models. Additional research that further reveals the benefits of telematics metrics for improving the understanding of driving behavior includes, but is not limited to, Constantinescu *et al.* (2018), Verbelen *et al.* (2018), Pérez-Marín *et al.* (2019), Pesantez-Narvaez *et al.* (2019), and Guillen *et al.* (2020).

For most portfolios of motor insurance, it is rare to observe one claim from a policyholder, let alone two or more claims. This fact presents challenges regarding the development of learning algorithms to handle sparse information. As will be discussed, this is even more relevant in the case of telematics data due to the perceived self-selection for drivers with UBI. Because of the attractiveness of potential premium savings, policyholders of UBI believe they

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

are and actually tend to be more careful drivers, making sparsity an even more challenging issue. The early work of Pednault *et al.* (2000) suggests addressing highly imbalanced classification using a tree-based model with the log-likelihood used as an impurity function for identifying the splitting of the regions. The most common measure of impurity used for classification trees is the Gini index. However, this previous work lacks empirical evidence to support the method. We distinguish our work from the previous literature by building a classification model for accident frequency and thereby addressing the sparsity of recorded claims for datasets containing telematics information resulting from highly imbalanced observed frequencies.

We here employ an extended multi-class classification algorithm, referred to as SAMME.C2, to handle class imbalances. The proposed algorithm has its origin in the work by Zhu *et al.* (2009), which introduced SAMME (Stagewise Additive Modeling using Multi-class Exponential loss function), a multi-class adaptive boosting (AdaBoost) classification model. AdaBoost, introduced by Freund and Schapire (1997), is an iterative classification algorithm that combines several weak and inaccurate learners to improve prediction accuracy. In contrast with other similar AdaBoost techniques, SAMME.C2 uses a cost-sensitive learning mechanism to rebalance and tilt the class distribution by accounting for the costs of prediction errors. This integration of the multi-class AdaBoost algorithm with the cost-sensitive learning concept is one of the main innovations of our paper.

To some extent, the proposed algorithm is motivated by a telematics dataset drawn from a UBI program offered by a Canadian-owned insurance cooperative. We here focus on the number of accidents as the response variable during the observation period. For the training data, 97.1% of observations had zero accidents, 2.8% had exactly one accident, and only 0.1% had two or more accidents. These observations are highly imbalanced, and we investigated the predictive power of telematics metrics apart from traditional metrics to understand the heterogeneous characteristics of drivers with UBI. The telematics information drawn from this data was not raw but had been pre-engineered. This should not affect the number of accidents observed, though it may affect other pre-engineered feature variables. Feature variables that are highly correlated, including those that are compositional in nature, were addressed by applying a principal component analysis (PCA).

Using both simulated and real datasets, we found that SAMME.C2 outperformed other multi-class classification models that handle class imbalances, including SAMME, SAMME with SMOTE, SAMME with SVMSMOTE, SAMME with KMeansSMOTE, RUSBoost, and SMOTEBoost. Each of these methods is briefly explained in the subsequent section. While performance statistics, such as recall and precision, are not unreasonable measures for comparison, it is more practical to use the macro average geometric of recall (MAvG) metric for comparison in situations with highly imbalanced classes. This performance metric is merely a geometric average of the recall for all classes. In addition,

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

we found that telematics variables provide more information than traditional variables in terms of measuring accident frequency.

The rest of the paper is organized as follows. Section 2 provides an overview of related works on AdaBoost suitable to handle class imbalances. In Section 3, we introduce our novel SAMME.C2 algorithm. It is largely based on an integration of the SAMMEand Ada.C2 methods, which are both also described in detail. Section 4 presents simulation results to assess the performance of SAMME.C2. We also briefly explain various performance metrics used to compare classification models. Section 5 presents the results based on our telematics dataset, and Section 6 concludes the paper.

2. Related work

This section provides an overview of related work and techniques, which can be categorized based on three levels of focus: (a) the data level; (b) the algorithm level; and (c) the cost-sensitive learning level.

2.1. Handling minority classes

As pointed out by Yang and Wu (2006), one of the challenges of data mining is dealing with observations that may suffer due to the presence of imbalanced classes. At the data level, the class distribution may be inherently imbalanced, creating classes that are considered majority (too much data) or minority (lacking data), as is the case with our telematics dataset. It is difficult to predict classes belonging to the minority since there are only a few samples to learn from in terms of the features to predict from this class. One obvious approach to this issue is to resample from the dataset so that the class distribution is rebalanced by oversampling (or undersampling) from the underrepresented (or overrepresented) classes. A technique developed by Chawla *et al.* (2002) called SMOTE (synthetic minority oversampling technique) is becoming increasingly popular as an oversampling approach. In contrast to oversampling with replication, the SMOTE algorithm creates synthetic samples for the minority class. These are generated by drawing samples using the k-nearest neighbors (KNN) method, which are linearly connected to produce the synthetic samples. As stated by Chawla *et al.* (2002), it "works to cause the classifier to build larger decision regions that contain nearby minority class points." While SMOTE is therefore used to increase prediction accuracy over minority classes, we examine uses of this algorithm in combination with boosting methods aiming to increase the accuracy over the entire dataset. In addition, we consider recent approaches that combine SMOTE algorithm with the use of support vector machines (SVMs; Tang *et al.* (2009)) or k-means clustering (Douzas *et al.* (2018)) for handling class imbalances.

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

2.2. Boosting

At the algorithm level, boosting techniques are considered some of the most powerful learning algorithms discovered in recent years. Boosting is based on the principle of combining several weak learners to produce a strong learner to achieve more accurate predictions. Beginning with equal observation weights, the algorithm is an iterative process that involves fitting classifiers at each step and adjusting the weights in the subsequent steps according to the results of the classification. More weights are given to observations that have been misclassified. It should be noted that, as pointed out by Hastie *et al.* (2009), while boosting was originally intended for classification problems, the concept has also been expanded to regression problems.

The first practical boosting algorithm was introduced by Freund and Schapire (1997) and is referred to as `AdaBoost.M1`. Today, AdaBoost now refers to a whole class of AdaBoost algorithms. However, `AdaBoost.M1` is still a well-known iterative boosting algorithm. Let us assume that we are given a dataset denoted by $(\boldsymbol{x}_i, y_i)$ for $i = 1, \ldots, N$, where $\boldsymbol{x}_i$ is a set of feature variables and y_i is a binary variable. Beginning with a distribution of equal weights to the observations, this is updated after each iteration based on α_t, which is a function of the weighted classification error:

$$\epsilon_t = \frac{\sum_{i=1}^{N} D_i I(y_i \neq h_t(\boldsymbol{x}_i))}{\sum_{i=1}^{m} D_i}, \tag{2.1}$$

where D_i is the distribution of the weights and $h_t(\boldsymbol{x}_i)$ is the classifier at step $t \in \{1, 2, \ldots, T\}$. This weighted error tends to increase during the iteration process, while the final classifier's training error gradually decreases. It has been shown that `AdaBoost.M1` is equivalent to an additive model with a minimization of an exponential loss function (Friedman *et al.* (2000) and Hastie *et al.* (2009)) and that it therefore belongs to the traditional statistical family of forward stagewise additive models. Considering this viewpoint, it can be seen that the algorithm is efficient and has a straightforward statistical interpretation. Several variant AdaBoost algorithms have appeared in the literature (Ferreira and Figueiredo (2012)). See also Ferrario and Hämmerli (2019) for a review of boosting for actuarial applications.

AdaBoost algorithms have gained widespread popularity, and several works have demonstrated their advantages. Schapire and Singer (1999) shows that the training error of the final classifier is bounded, while Freund and Schapire (1997) demonstrates that each weak classifier is slightly better than random, and the training error drops exponentially in relation to T, the number of weak classifiers. Some scholars, such as Friedman *et al.* (2000), have also reported that AdaBoost is robust to overfitting by demonstrating that test error consistently decreases and then levels off as more classifiers are added. In terms of practicality, many empirical applications in machine learning have shown that AdaBoost algorithms are superior classifiers. For instance, Friedman *et al.*

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

(2000) called AdaBoost with decision trees the “best off-the-shelf classifier in the world.”

AdaBoost.M1 has also been extended to multi-class classification problems in which y_i belongs to the set $\{1, 2, \ldots, K\}$. The extension AdaBoost.M2 presented by Freund and Schapire (1997), which is based on a pseudo-loss function instead of the error rate, is also suitable for handling multi-class problems. AdaBoost.MH, developed by Schapire and Singer (1999), is a multi-class AdaBoost algorithm based on the Hamming loss function. Because this loss function is applied to create a set of binary problems, the procedure may be slow and thereby inefficient. These are only a few such extensions. However, in this paper, we compare our proposed method to the multi-class extension of the forward stagewise additive model, which is based on a generalization of exponential loss to multiple classes given by:

$$L(\boldsymbol{U}_i, \boldsymbol{f}(\boldsymbol{x}_i)) = \exp\left(-\frac{1}{K}\sum_{k=1}^{K} U_{ki} f_k(\boldsymbol{x})\right) = \exp\left(-\frac{1}{K}\boldsymbol{U}_i' \boldsymbol{f}(\boldsymbol{x}_i)\right), \tag{2.2}$$

for observation i. Here, $\boldsymbol{U}_i$ is a recoding of y_i, and $\boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_K(\boldsymbol{x}))'$ is a set of classifier functions that solve the optimization routine. All entries in the vector $\boldsymbol{U}_i$ will be equal to $-1/(K-1)$ except for a value of 1 in position k if the observation $y_i = k$. In effect, we have $\mathbf{U}_i = (U_{1i}, U_{2i}, \ldots, U_{Ki})'$, where:

$$U_{ki} = \begin{cases} 1, & \text{if } y_i = k \\ -\frac{1}{K-1}, & \text{if } y_i \neq k \end{cases}$$

Such code, inspired by a similar one used for SVM algorithms, gives a one-to-one correspondence between $\mathbf{U}_i$ and y_i, with each referring to the class the observation i belongs to. It can be seen that $\sum_{j=1}^{K} U_{ji} = 0$ for all $i = 1, 2, \ldots, N$. Developed by Zhu *et al.* (2009), the code has been referred to as the SAMME algorithm, and the detailed steps of the algorithm are summarized as Algorithm 2 in Appendix A.

As it is common to combine the benefits of resampling and boosting, we also considered the following algorithms:

- SAMME with SMOTE sampling;
- SAMME with SVMSMOTE sampling;
- SAMME with kMeansSMOTE sampling;
- SMOTEBoost (described in Chawla *et al.* (2003); an approach for learning from minority classes based on a combination of SMOTE and AdaBoost.M2); and
- RUSBoost (described in Seiffert *et al.* (2010); an algorithm with the same goal as SMOTEBoost but that replaces SMOTE sampling with random undersampling).

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

2.3. Cost-sensitive learning

Cost-sensitive learning adds an additional layer of complexity to our algorithm to further improve its prediction accuracy. In particular, it takes into account misclassification costs by adding a penalty to predictions that lead to incorrect classification. While the added costs are primarily those accounting for misclassification, it has been identified that cost adjustments may be made to reflect costs of other types, such as those associated with computational efficiency/complexity, data collection, or model evaluation. The primary objective is to minimize the total costs of the model. Cost-sensitive algorithms that minimize misclassification costs in classification problems first appeared in Pazzani *et al.* (1994). The cost matrix is an additional input to the learning procedure and is also used to evaluate the ability of the learned procedure to reduce misclassification costs. In the context of AdaBoost, the cost adjustment function is used to modify the updating of the weights at each iteration (see Galar *et al.* (2012)). Zhang (2020) developed CS-KNNs, an extension to the KNN classification method that applies cost-sensitive learners to handle class imbalances. Shon *et al.* (2020) developed COST-HDL, an extension of a hybrid deep learning algorithm with a cost-sensitive loss function that can help to classify imbalanced kidney cancer data. The most familiar method combining AdaBoost with cost-sensitive learning is `Ada.C2`, which inspired the algorithm suggested in this paper. In the following sections, we discuss this algorithm in detail, as it was a precursor to our proposed multi-class cost-sensitive algorithm.

3. `SAMME.C2` ALGORITHM

The `SAMME.C2` algorithm combines the benefits of boosting and cost-sensitive algorithms to handle class imbalances in multi-class classification problems. Boosting algorithms are generally considered advantageous because their implementation is straightforward, they are statistically justified and generally suitable for many kinds of classification problems, and they lead to relatively few issues in terms of overfitting. By directly penalizing misclassified samples, cost-sensitive learning algorithms provide the added benefit of high prediction accuracy, especially for minority classes. The cost-sensitive component of our proposed algorithm was inspired by the `Ada.C2` method.

3.1. `Ada.C2` method

AdaBoost models treat samples of different classes equally. In particular, the weights of misclassified samples from different classes are increased by an identical ratio, while the weights of correctly classified samples from different classes are decreased by another identical ratio. A desirable boosting strategy for an imbalanced dataset is one that is able to distinguish different classes

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

of samples and provide a greater boost to the weights of the samples associated with higher costs. The concept of Ada.C2 was introduced by Sun *et al.* (2007), and this method adds a cost item to each sample as follows. The input data to the algorithm consists of $(\boldsymbol{x}_i, y_i, C(y_i))$ for $i = 1, 2, \ldots, N$, where i is the index used for each individual driver, and N is the total number of individual drivers in the training set. The differences between this algorithm and the original AdaBoost algorithm are as follows:

1. The updating of the distribution of the dataset at each iteration has the form:

$$D_{t+1}(i) = \frac{C(y_i)\, D_t(i) \exp\left(-\alpha_t I(y_i = h_t(\boldsymbol{x}_i))\right)}{\sum_{j=1}^{N} C(y_j)\, D_t(j) \exp\left(-\alpha_t I(y_j = h_t(\boldsymbol{x}_j))\right)}.$$

2. The weight of each classifier is

$$\alpha_t = \frac{1}{2} \log\left(\frac{\sum_{i=1}^{N} C(y_i)\, D_t(i)\, I(y_i = h_t(\boldsymbol{x}_i))}{\sum_{i=1}^{N} C(y_i)\, D_t(i)\, I(y_i \neq h_t(\boldsymbol{x}_i))}\right).$$

AdaBoost is accuracy-oriented, and its weighting strategy may therefore still tilt toward the majority class, since it contributes more to the overall classification accuracy. However, Ada.C2 uses a strategy for updating the data distribution based on assigning higher costs to minority classes. On the one hand, when minority classes are misclassified, the weights increase more so than when majority classes are misclassified. On the other hand, when minority classes are correctly classified, the weights decrease less than when majority classes are correctly classified. The steps used in the algorithm are summarized as Algorithm 3 in Appendix A.

3.2. The proposed algorithm

The proposed algorithm, which we call SAMME.C2, is a blend of SAMME and Ada.C2. Although we adopt all of the algorithmic steps from Ada.C2, there are a number of important differences. First, we use the calculation formula of α_t from the SAMME algorithm at the iterative step t, which includes the addition of $\log(K-1)$. As pointed out by Zhu *et al.* (2009), this adjustment term is crucial for multi-class classification problems as it helps to ensure that "the accuracy of each weak classifier is better than random guessing." It can be shown that the presence of the term $\log(K-1)$ is a consequence of the solution to the optimization based on the extended muti-class exponential loss function. Second, in our algorithm, the calculation of the weighted classification error, ϵ_t, is not adjusted with the cost values. This was necessary to prove our algorithm follows the form of a forward stagewise additive model. Details of this proof are omitted in this paper.

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

The steps of the proposed algorithm are summarized as follows:

Algorithm 1: SAMME.C2: cost-sensitive multi-class AdaBoost

```
Input: Training dataset $\boldsymbol{x}_i \in X, y_i \in Y = \{1, 2, \ldots, K\}, C(y_i) \in (0, 1], T$
Output: Final classifier $H(\boldsymbol{x}_i)$
Set initial distribution of dataset equally distributed:
$D_1(i) = \frac{1}{N}, \quad i = 1, 2, \ldots, N$ ;
for $t = 1, \ldots, T$ do
    Train weak classifier using the distribution $D_t$;
    Get weak classifier $h_t : X \to k \in \{1, 2, \ldots, K\}$ ;
    Compute $\epsilon_t = \frac{\sum_{i=1}^{N} D_t(i) I(y_i \neq h_t(\boldsymbol{x}_i))}{\sum_{i=1}^{N} D_t(i)}$ ;
    Choose $\alpha_t = \log\left(\frac{1 - \epsilon_t}{\epsilon_t}\right) + \log(K - 1)$ ;
    Update $D_{t+1}(i) = \frac{C(y_i)\, D_t(i) \exp(-\alpha_t I(y_i = h_t(\boldsymbol{x}_i)))}{\sum_{j=1}^{N} C(y_j)\, D_t(j) \exp(-\alpha_t I(y_j = h_t(\boldsymbol{x}_j)))}$ ;
end
Return the final classifier: $H(\boldsymbol{x}_i) = \underset{k}{\operatorname{argmax}} \sum_{t=1}^{T} \alpha_t I(h_t(\boldsymbol{x}_i) = k)$ ;
```

Figure 1 presents a comparative visual representation regarding how the algorithm correctly classifies (or misclassifies) majority and minority classes. For the SAMME algorithm, there is an even redistribution of correct classification (or misclassification) for majority and minority classes. In contrast, with the addition of a cost-sensitive learning mechanism, the redistribution is uneven and gives a heavier weight to minority classes. In effect, after a sufficiently large number of iterations, weak classifiers are modeled with a heavy emphasis on minor and misclassified instances.

4. Performance evaluation and simulation

4.1. Performance metrics

For classification problems, the most common measurement of performance is accuracy, which is the proportion of all observations that are correctly classified. For obvious reasons, this is an irrelevant measure for imbalanced datasets. Let us consider three performance statistics: recall, precision and F1-score. We can compute performance statistics for each class, $i = 1, 2, \ldots, K$ and aggregate them with an average. The recall for class i, R_i is defined as the proportion of observations in class i that were correctly classified. The recall is also sometimes referred to as the sensitivity. The precision, P_i, is the proportion of predictions in class i that were correctly classified. The F1-score,

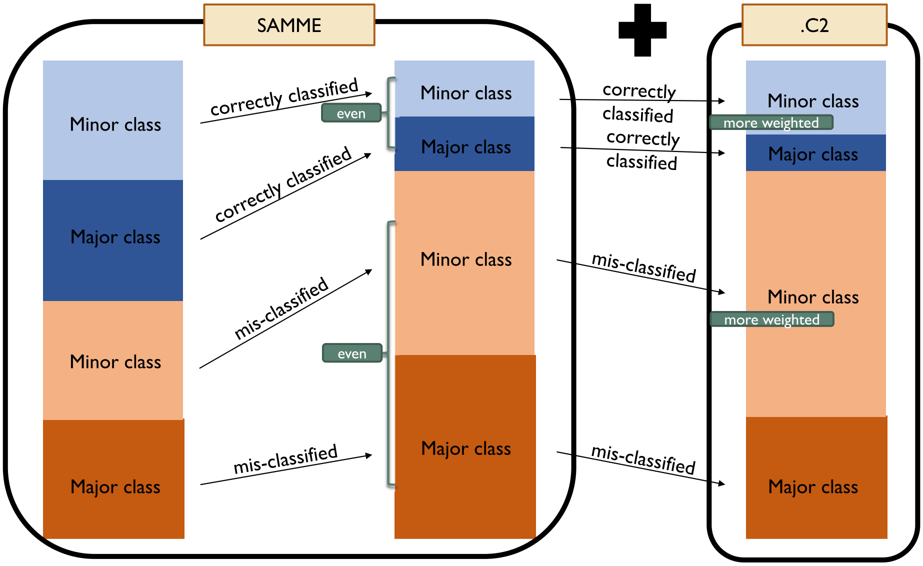


FIGURE 1: Visualizing the effect of the `SAMME.C2` algorithm on classifying majority/minority classes.

$F1_i$, is the harmonic average of recall and precision and is therefore equal to $2 \cdot (R_i \times P_i)/(R_i + P_i)$. For visualization, receiver operating characteristic curves are popular for binary classifiers but are less meaningful for multi-class classification problems, especially those with imbalanced classes.

When aggregating the results to provide a single measure of performance for a given classifier, we define statistics, such as macro-precision and macro-recall, to represent the arithmetic averages of the respective performance statistics for each class. We define macro-F1-score as the reciprocal of the harmonic average of macro-precision and macro-recall, that is, macro-F1-score = (2 ∗ macro-precision ∗ macro-recall)/(macro-precision + macro-recall). For the recall, we use the geometric average and define

$$\text{MAvG} = (R_1 \times R_2 \times \cdots \times R_K)^{1/K}. \tag{4.1}$$

Finally, when comparing classifiers, a better performing classifier is one that gives a larger value for each of these aggregate statistics.

If we take the log of both sides of the MAvG statistics, we obtain an average of the log of all the recall statistics. For log transformation, the result provides an average of the importance of accurately classifying observations for all classes. For imbalanced datasets, this means that it is perfectly acceptable to increase misclassifications of a majority class to correctly classify more of a minority class. Indeed, as has been previously discussed (Fernández *et al.* (2018)), the recall, or sensitivity, is usually a more interesting measure for imbalanced classification. For these reasons, the MAvG is a good performance

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

measure for imbalanced datasets. The MAvG concept is credited to the work of Fowlkes and Mallows (1983).

4.2. Simulation

To investigate how the SAMME.C2 algorithm performs relative to other boosting algorithms, the first crucial step was to perform a test based on a simple simulated dataset. As mentioned, for the sake of comparison, we chose six other boosting algorithms that are potentially viable for handling imbalanced multi-class classification problems: (a) SAMME, (b) SAMME with SMOTE, (c) SAMME with SVMSMOTE, (d) SAMME with KMeansSMOTE, (e) RUSBoost, and (f) SMOTEBoost. Each of these methods has been described in the previous section.

To generate a simulated dataset, we used the *scikit-learn* Python module described in Pedregosa *et al.* (2011). This module is a user-friendly tool for applying "state-of-the-art machine learning algorithms." We used the built-in function make_classification with parameterization executed as:

```
"""Make Simulation"""
from sklearn.datasets  import make_classification

X, y = make_classification(n_samples=100000, n_features=50, n_informative=5-,
n_redundant=0, n_repeated=0, n_classes=3, n_clusters_per_class=2, class_sep=2,
flip_y=0, weights=[0.96,0.035, 0.005], random_state=16)
```

This generated 100,000 samples with 50 features, along with 3 classes. We deliberately created a highly imbalanced dataset by setting the ratios for the three classes as 96%, 3.5%, and 0.5%. Please refer to the package for an explanation of the other parameters. We used 75% of the data for training, and the rest was used for testing.

For the sake of simplicity, our analysis used a one-depth decision tree as a weak classifier, as is used with many boosting algorithms. In addition, 200 weak classifiers were linearly combined with weights. To implement the cost values for SAMME.C2, we employed a genetic algorithm (GA), which is a directed random search technique invented by Holland (1975) and described in Mühlenbein (1997). This same GA to implement cost values was also used in the empirical application. See also Bhowan *et al.* (2010).

For the GA, we first create the population set consisting of M arbitrary cost vectors, after which we run SAMME.C2 and determine the resulting MAvG, hereafter referred to as the objective function. A cost vector has three elements corresponding to each class. In the selection step used to select the two cost vectors from the M vectors, we employ the "choice by roulette" method. Since a large portion of the roulette wheel is assigned to cost vectors with a large MAvG, this method has a higher probability of choosing cost vectors with a large MAvG. In the crossover step, we combine the two selected cost vectors into a single vector using an arithmetic average. In the mutation step, we pick

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

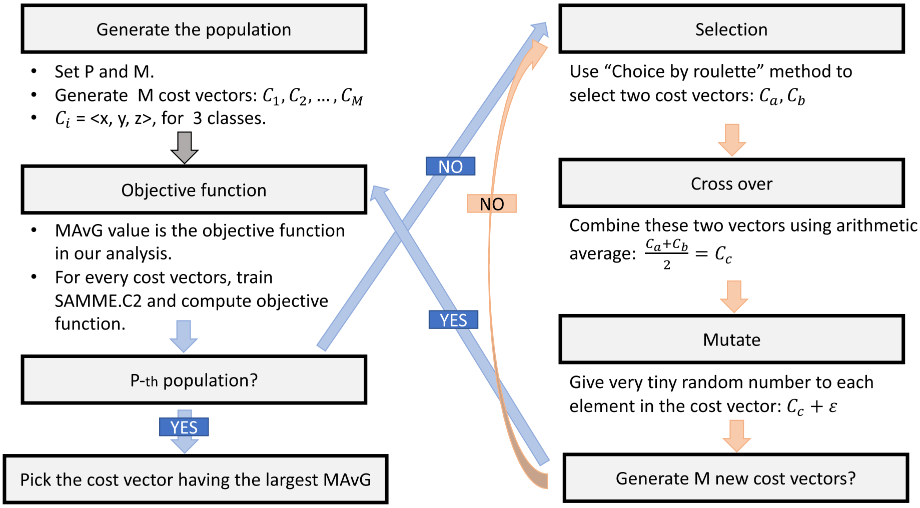

Generate the population
• Set P and M.
• Generate M cost vectors: $C_1, C_2, \ldots, C_M$
• C_i = <x, y, z>, for 3 classes.
Objective function
• MAvG value is the objective function in our analysis.
• For every cost vectors, train SAMME.C2 and compute objective function.
P-th population?
YES
Pick the cost vector having the largest MAvG
NO
NO
YES
Selection
Use "Choice by roulette" method to select two cost vectors: C_a, C_b
Cross over
Combine these two vectors using arithmetic average: $\frac{C_a+C_b}{2} = C_c$
Mutate
Give very tiny random number to each element in the cost vector: $C_c + \varepsilon$
Generate M new cost vectors?

FIGURE 2: Visualizing the genetic algorithm.

a random number within a tiny interval, which is used to adjust the elements in the cost vector. Repeating these selection, crossover, and mutation steps, we are able to produce a new population with new M cost vectors, for which the procedure is iteratively repeated P number of times to generate the population that produces the best cost vectors. Figure 2 provides a visualization of these steps of the GA. In our search, we found that setting $P = 10$ and $M = 10$ produced cost vectors that optimized the MAvG. For the simulation, the final cost vector produced had elements 1 for the most minority class, 0.963 for the majority class, and 0.985 for the remaining class.

The GA is a necessary tuning process for the cost vectors given our purpose. To demonstrate the sensitivity of the cost vectors to the performance measure (MAvG), we examined the effects of randomly adjusting the cost vectors with uniform distributions (−0.01, 0.01) and (−0.001, 0.001). The resulting effects are visualized in Figure 3. The top part of the graph shows the sensitivity of MAvG to uniform adjustments between −0.01 and 0.01. Here, even the range between 0.3 and 0.7 did not lead to an optimal MAvG. The bottom part of the graph shows a lower sensitivity to MAvG based on much smaller uniform adjustments between −0.001 and 0.001.

Each boosting algorithm was trained using the training data, and its performance was then evaluated on the test data. The results on the performance of the various models are summarized in Table 1. According to the results, all models except `RUSBoost` had similar Gini coefficients. In terms of MAvG, `SAMME.C2`, which had an MAvG of 0.93, outperformed all the other models, though `SAMME` with `SMOTE` was not far behind at 0.90. An examination of each class' recall statistics revealed that `SAMME.C2` outperformed all other models in terms of correctly classifying those belonging to the minority class. However,

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

TABLE 1

COMPARING THE PERFORMANCE MEASURES USING MAVG BASED ON THE SIMULATED DATASET.

	Recall statistics						
		SAMME with	SAMME with	SAMME with			
Class	SAMME	SMOTE	SVMSMOTE	KMeansSMOTE	RUSBoost	SMOTEBoost	SAMME.C2
Class 1	1.00	0.96	0.92	0.97	0.88	0.99	0.88
Class 2	0.84	0.93	0.96	0.91	0.88	0.91	0.95
Class 3	0.42	0.83	0.82	0.82	0.30	0.76	0.95
MAvG	0.71	0.90	0.90	0.90	0.62	0.88	0.93
Gini coeff.	0.96	0.96	0.96	0.96	0.80	0.95	0.96

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

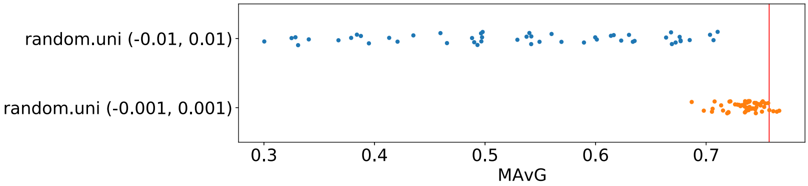


FIGURE 3: SAMME.C2 model's sensitivity of cost values to MAvG. Top: uniform adjustments to cost values from (− 0.01, 0.01). Bottom: uniform adjustments to cost values from (− 0.001, 0.001). Vertical (red) line: MAvG based on original cost vector.

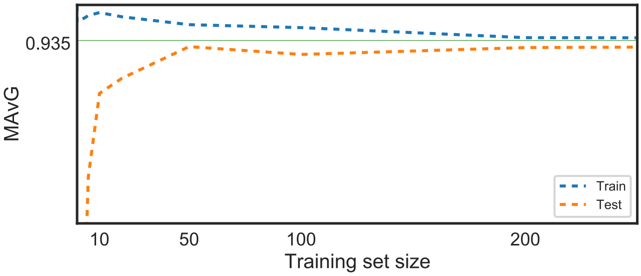


FIGURE 4: Learning curves for `SAMME.C2` based on the simulated data. Values on the x-axis, representing the size of the training dataset, are in thousands ('000).

this came at the sacrifice of having the worst recall statistics for the majority class (Class 1). The Gini coefficients were calculated according to the method presented in Hand and Till (2001).

Figure 4 shows a set of diagnostic learning curves for examining the bias and variance associated with training the `SAMME.C2` based on the performance metric MAvG. The graph demonstrates that we were able to achieve extremely low bias and low variance simultaneously due to performance convergence as the size of the training dataset increased (see Hastie *et al.* (2009)).

5. EMPIRICAL DATA: TELEMATICS

The overall goal of this paper is to analyze and understand driver behavior using telematics. In this section, we demonstrate the strength of our `SAMME.C2` algorithm as a model for analyzing and understanding the effects of the value added by the telematics information. Insurance companies have long been using traditional variables, such as age and gender, for the driver risk classification. The advancement of technology has led insurers to offer innovative products, such as UBI (which uses telematics), to better classify and price risks, with the understanding that such additional information helps to investigate driver behavior.

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

TABLE 2

SUMMARY STATISTICS OF THREE (CONTINUOUS) TRADITIONAL VARIABLES.

	Mean	Std dev	Min	Q1	Median	Q3	Max
DRIVER.AGE	51.3	16.8	16.0	38.0	51.0	65.0	103.0
VEHICLE.AGE	5.7	4.49	−2.0	2.0	5.0	9.0	20.0
CREDIT.SCORE	754.8	88.0	390.0	716.0	780.0	811.0	892.0

5.1. Description and summary of data

Our telematics dataset was acquired from a Canadian-owned cooperative that offers insurance and investment products. Its UBI program was launched in Ontario in 2013. This dataset consists of information from drivers who participated in the program and were observed during the period from 2013 to 2016, and it is based on nearly 100,000 policies. The telematics information here is not raw but has been pre-engineered for the purpose of training a statistical model for predictive purposes. This is in contrast to the work of Gao *et al.* (2021), which analyzes raw telematics data acquired on a more frequent basis. We considered feature variables as telematics when they were drawn as a result of voluntary participation in this telematics program, which required the installation of a vehicle-tracking device. At time of underwriting, it is possible that telematics-related data may have been acquired and for our purposes, they are considered classical and this includes, for example, the variable ANN.KMS.DRV.SYSTEM. The response variable of interest is accident frequency, classified according to the number of accidents observed per driver. We were able to observe 0 (no accidents), 1 (exactly one accident), or 2^+ (two or more accidents). For training, we had a total of 50,301 observations, among which 48,822 had no accidents, 1,430 had exactly one accident, and only 49 had 2 or more accidents. It can thus be seen that the classes were highly imbalanced (97.1% with no accidents, 2.8% with exactly one accident, and only 0.1% with two or more accidents). The dataset had no missing values regarding acident frequency.

We had a total of 49 potential feature variables, 10 of which were traditional (e.g., driver age and gender), while the rest were telematics-driven. A description of each variable in our dataset is given in Table 8. To provide a preliminary understanding of the data, Table 2 shows summary statistics of three traditional continuous variables: DRIVER.AGE, VEHICLE.AGE, and CREDIT.SCORE. For example, it can be seen that the average driver age in our records was 51.3, with 16.0 and 103.0 being the youngest and oldest drivers, respectively. The cohort of drivers in our dataset with more mature driving experience tended to fall within the middle age groups. According to Figure 5, there does not seem to be a significant difference between males and females with respect to accident frequency, even after controlling for the effects of DRIVER.AGE, VEHICLE.AGE, and CREDIT.SCORE.

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

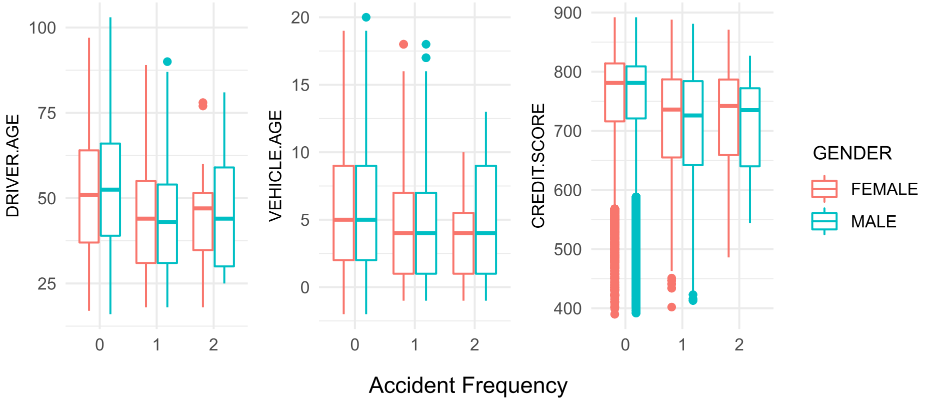


FIGURE 5: Analysis of accident frequency by gender.

Table 3 provides summary statistics of two telematics variables: DISTANCE.DRIVEN and EXPOSURE. For example, the average distance traveled was 7,555.3 km for those without accidents, 14,155.4 km for those with exactly one accident, and 12,834.89 km for those with two or more accidents. Broadly speaking, this appears to indicate that the greater the distance traveled, the more likely it is that the driver will have at least one claim.

We can broadly classify the telematics variables into those that are considered driving maneuvers (braking, acceleration, and left and right turn events) and those that do not fall into this category. Most variables that fall into the latter category are related to how much time drivers spend on the road (percentage of driving spent during each day of the week, exposure, and distance driven; see Verbelen *et al.* (2018)). According to Table 8, there are essentially four types of driving maneuver: braking, acceleration, left turn events, and right turn events. The work by Gao *et al.* (2021) similarly examines these driving maneuvers involving speed and acceleration.

BRAKE.xxKM refers to the number of sudden brakes applied at different measures of deceleration (10,000/13,000/15,000/17,000/20,000/23,000 km per h/s). The more rapid the deceleration, the faster the car is being driven when a sudden brake is applied. Figure 6 displays a boxplot of the frequency of these sudden brakes according to accident frequency. The y-axis was measured on a logarithmic scale. First, it should be noted that the application of brakes with higher deceleration is less frequently observed. Second, a more frequent application of brakes for different decelerations leads to an increase in the likelihood of an accident.

For left and right turn events (LTURN.EVENTxx and RTURN.EVENTxx), the intensity was measured on a scale of 812, which expresses the intensity of the force of gravity applied as the driver decelerates during a turn. For example, 8 refers to an intensity of 80% of (3.57/9.80665) per m/s^2, and similarly, 12 refers to the intensity of 120% of (3.57/9.80665) per m/s^2. The higher the intensity value, the more intense the movement of the turn.

TABLE 3

SUMMARY STATISTICS OF TWO TELEMATICS VARIABLES.

Variable	Acc Freq	Count	Mean	Std Dev	Min	Q1	Median	Q3	Max
DISTANCE.DRIVEN	0	48, 822	7555.3	7149.4	0.1	2374.8	5395.7	10, 592.7	76, 271.8
	1	1430	14, 155.4	8257.3	253.9	8319.7	12, 657.4	18, 161.2	58, 759.2
	2	49	12, 834.9	7925.9	2247.8	7408.1	11, 408.3	16, 621.3	46, 527.4
EXPOSURE	0	48, 822	0.49	0.31	0.00	0.24	0.50	0.73	1.08
	1	1430	0.78	0.25	0.02	0.64	0.89	1.00	1.06
	2	49	0.74	0.26	0.23	0.50	0.80	1.00	1.06

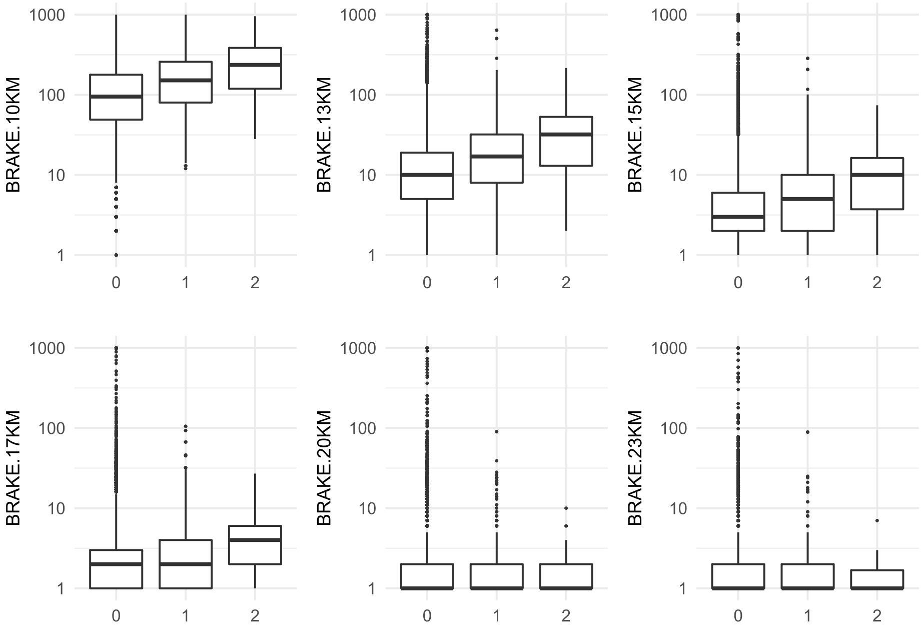


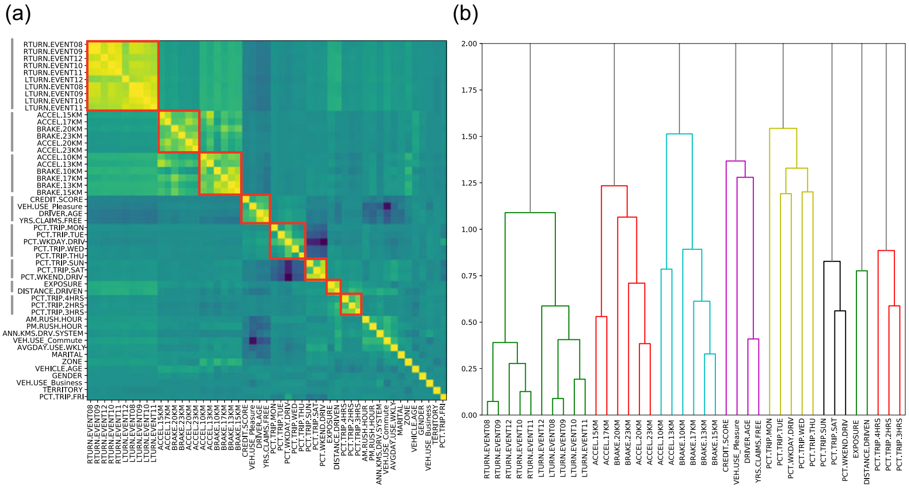
FIGURE 6: Analysis of accident frequency by exposure and distance driven.

5.2. Handling feature correlation

There are a number of clearly correlated features in our dataset. First, there are feature variables that are compositional in nature, which refers to those that describe parts of some whole and typically arise in terms of proportions or percentages. The sum of the elements of these vectors is usually fixed as a constant, such as 100%. To illustrate an example from the dataset, there are feature variables that describe the percentages of time spent driving on each day of the week (e.g., PCT.TRIP.MON and so forth). Such variables imply high feature correlations and must be handled carefully. Second, there are other feature variables that intuitively have a high correlation, such as EXPOSURE and DISTANCE.DRIVEN. If a driver has a longer driving distance, it is highly likely that the driver spends a higher percentage of time on the road.

Boosting algorithms using decision trees as weak learners are known to be robust to feature correlation. However, collinearity may affect the interpretation of certain results, such as the feature importance induced from the model. To avoid such distortion, one particularly useful approach, applied in Verbelen *et al.* (2018), is the application of log-ratio transformation for compositional features. However, we found this to be inadequate for our data due to the large number of zero occurrences related to compositional features in our dataset. At the same time, we wanted to control for the correlation of other feature variables, as earlier described. We thus applied a PCA after a preliminary investigation of the correlations of the feature variables.

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press



Heatmap of Spearman’s rank correlation

Dendogram of Ward’s hierarchical clustering

FIGURE 7: Analysis of feature correlation based on Spearman’s rank correlation with Ward’s hierarchical clustering.

The Spearman’s rank correlation coefficient was used to measure the degree of similarity between the feature variables. Based on these correlations, we then performed Ward’s hierarchical clustering and applied PCA to each cluster to create new independent feature variables according to the principal components (PCs). Figure 7(a) shows a heatmap of these features based on Spearman’s coefficients, with the features ranked according to the clustering results so that they match the clusters in Figure 7(b). The different colors indicate different clusters. The resulting new variables (PCs) in each cluster were set to explain above 90% of the group information. For example, the cluster comprising ACCEL.15KM, ACCEL.17KM, ACCEL.20KM, ACCEL.23KM, BRAKE.20KM, and BRAKE.23KM had only one PC because one PC explained 90% of the group information. The results of the eight new clustered feature variables are summarized in Table 4.

5.3. Empirical model evaluation

We next evaluated the performance of SAMME.C2 against other classification models using the test data set, as earlier described. For the test dataset, we included a total of 21,574 observations: 20,901 had no accidents, 650 had exactly one accident, and only 23 had two or more accidents. All the settings of the various algorithms were identical to the ones described in Section 4. The cost vector for SAMME.C2 was preprocessed using a GA for which the assigned cost was unique to each class in each observation. Employing the GA explained in Section 4.2, the cost values for Accident 0, Accident 1, and Accident 2^+ were found to be 0.958, 0.979, and 1, respectively.

TABLE 4

NEW VARIABLES (PRINCIPAL COMPONENTS) FROM THE EIGHT CLUSTERS.

	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	Cluster 6	Cluster 7	Cluster 8
New Variables	PC1.TURN.L PC2.TURN.R	PC1.ACCEL.H	PC1.ACCEL.L PC2.ACCEL.L PC3.ACCEL.L	PC1.AGE related PC2.AGE related PC3.AGE related	PC1.WEEK.D PC2.WEEK.D PC3.WEEK.D PC4.WEEK.D	PC1.WEEKEN.D PC2.WEEKEN.D	PC1.DISTANCE PC2.DISTANCE	PC1.TRIP.HRS PC2.TRIP.HRS
Original Variables	LTURN.EVENT08 LTURN.EVENT09 LTURN.EVENT10 LTURN.EVENT11 LTURN.EVENT12 RTURN.EVENT08 RTURN.EVENT09 RTURN.EVENT10 RTURN.EVENT11 RTURN.EVENT12	ACCEL.15KM ACCEL.17KM ACCEL.20KM ACCEL.23KM BRAKE.20KM BRAKE.23KM	ACCEL.10KM ACCEL.13KM BRAKE.10KM BRAKE.13KM BRAKE.15KM BRAKE.17KM	CREDIT.SCORE VEH.USE_Pleasure DRIVER.AGE YRS.CLAIMS.FREE	PCT.TRIP.MON PCT.TRIP.TUE PCT.TRIP.WED PCT.TRIP.THU PCT.WKDAY.DRIV	PCT.TRIP.SAT PCT.TRIP.SUN PCT.WKEND.DRIV	EXPOSURE DISTANCE.DRIVEN	PCT.TRIP.2HRS PCT.TRIP.3HRS PCT.TRIP.4HRS

The results summarized in Table 5 are very encouraging regarding SAMME.C2. In terms of the MAvG performance metric, with a value of 0.76, it far outperformed all the other models. Further, with a Gini coefficient of 0.67, SAMME.C2 is considered the best performing classifier, as this was the highest among all the models considered. Although SAMME.C2 makes a number of sacrifices regarding the prediction accuracy of the majority class (here, the case of zero accidents), it produced a superior outcome for predicting the other classes considered minorities. As in the simulations, the worst classifiers were those that either did not resample (SAMME) or did resample but only used undersampling (RUSBoost). Recall also that in the simulation, SAMME.C2 only slightly outperformed the other two boosting algorithms that used the SMOTE resampling method. However, our results demonstrate when the empirical data are used, a cost-sensitive learning mechanism more significantly outperforms mechanisms that use resampling methods.

Increasing the tree depth can be treated as part of hyper-parameter tuning and, as pointed out in Ferrario and Hämmerli (2019), has the benefits of potentially capturing linearities and introducing interaction terms. For our empirical data, we additionally tested the effect of the weak learner's tree depth. We ran SAMME.C2 with depths of 1 through 4, and the resulting MAvGs were 0.76, 0.77, 0.72, and 0.64, respectively. Clearly, weak learners based on tree stumps and two depths perform similar to but slightly better than those with higher depths. Accordingly, increasing the depth can be recommended to reduce the effect of feature correlation. However, since we preprocessed the data with PCA to handle this issue, these differences in performance results between models of different depths were not unexpected. For this reason, we pursued the SAMME.C2 algorithm with tree stump weak learners.

To further assess the effectiveness of SAMME.C2, especially with respect to overfitting, we note that there was very little difference when using in-sample and stratified 10-fold cross-validation. Indeed, the MAvGs of in-sample and stratified 10-fold cross-validation were 0.76 and 0.73, respectively. We also examined a set of diagnostic learning curves to assess bias and variance, which led to patterns similar to those shown in Figure 4.

5.4. Comparing traditional and telematics features

Several researchers and practitioners continue to be interested in the usefulness of vehicle telematics information to build more customized pricing models. This subsection shows how the results of the SAMME.C2 algorithm can help enhance our understanding of driving behavior.

To evaluate the added value of telematics information, we compared the results of SAMME.C2 using only the 10 traditional variables and those using the 49 traditional and telematics variables. The results are summarized in Table 6.

Based on the confusion tables, it can easily be calculated that the MAvG was 0.76 for the SAMME.C2 using all 49 variables. In contrast, when only the

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

Table 5

Comparing the performance measures using MAvG based on the telematics dataset.

Accident Frequency	Recall statistics						
	SAMME	SAMME with SMOTE	SAMME with with SVMSMOTE	SAMME with KMeansSMOTE	RUSBoost	SMOTEBoost	SAMME.C2
Accident 0	1.00	0.79	0.88	0.80	0.70	0.99	0.67
Accident 1	0.01	0.45	0.50	0.52	0.43	0.10	0.67
Accident 2^+	0.00	0.39	0.09	0.26	0.09	0.22	0.96
MAvG	0.01	0.52	0.34	0.48	0.30	0.28	0.76
Gini coeff.	0.66	0.51	0.46	0.44	0.36	0.59	0.67

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

TABLE 6

CONFUSION TABLES BASED ON THE MODEL FIT OF SAMME.C2.

With traditional variables only

		Actual			
		Acc 0	Acc 1	Acc 2$^+$	Tot row
Predicted	Acc 0	11,674	175	1	11,850
	Acc 1	5319	274	3	5596
	Acc 2$^+$	3908	201	19	4128
	Tot col	20,901	650	23	21,574

With traditional and telematics variables

		Actual			
		Acc 0	Acc 1	Acc 2$^+$	Tot row
Predicted	Acc 0	14,077	94	0	14,171
	Acc 1	5930	438	1	6369
	Acc 2$^+$	894	118	22	1034
	Tot col	20,901	650	23	21,574

traditional variables are used, the MAvG was 0.58. This indicates that prediction models from SAMME.C2 that include telematics information outperform those considering only traditional feature variables. Interestingly, we also note that the Gini coefficients were 0.67 and 0.46, respectively, with a higher index for the model with both traditional and telematics variables. This is particularly relevant when a classification model is used to build accident frequency models because the index is related to the efficiency of the estimation of the expected number of accidents derived from the classifier. Prior studies have similarly concluded that claim frequency models that contain telematics information are far better than those containing only classical feature variables (see Boucher *et al.* (2017), Ayuso *et al.* (2019), and Guillen *et al.* (2020)). It should be noted that the diagonal values in the table are the numbers of correctly classified instances for each accident frequency according to the SAMME.C2 algorithm. With a straightforward comparison of these diagonal values, it can be deduced that the addition of the telematics variables improved the prediction.

Figure 8 depicts the relative importance of the feature variables, with the telematics variables listed first and followed by the traditional variables. With the assumption that a weak classifier can provide feature importance, the common procedure to calculate the index used in AdaBoost is to use the weighted average of the weak classifier's feature importance, using α_t as weights. In this paper, we used decision trees as weak classifiers, and the feature importance was evaluated according to the Gini impurities. Therefore, the feature importance of SAMME.C2 was computed as the weighted average of the decreases in these impurities. While the SAMME.C2 model was fitted based on the cluster of variables resulting from the PCA, we reverted these clusters back into the original variables when computing feature importance. Figure 8 provides the different degrees of contribution to the feature importance made by each class of accidents. In the figure, these contributions are distinguished according to color, as labeled.

There are a number of conclusions we can draw from this decomposition of the feature variable contributions. First, Figure 8 highlights

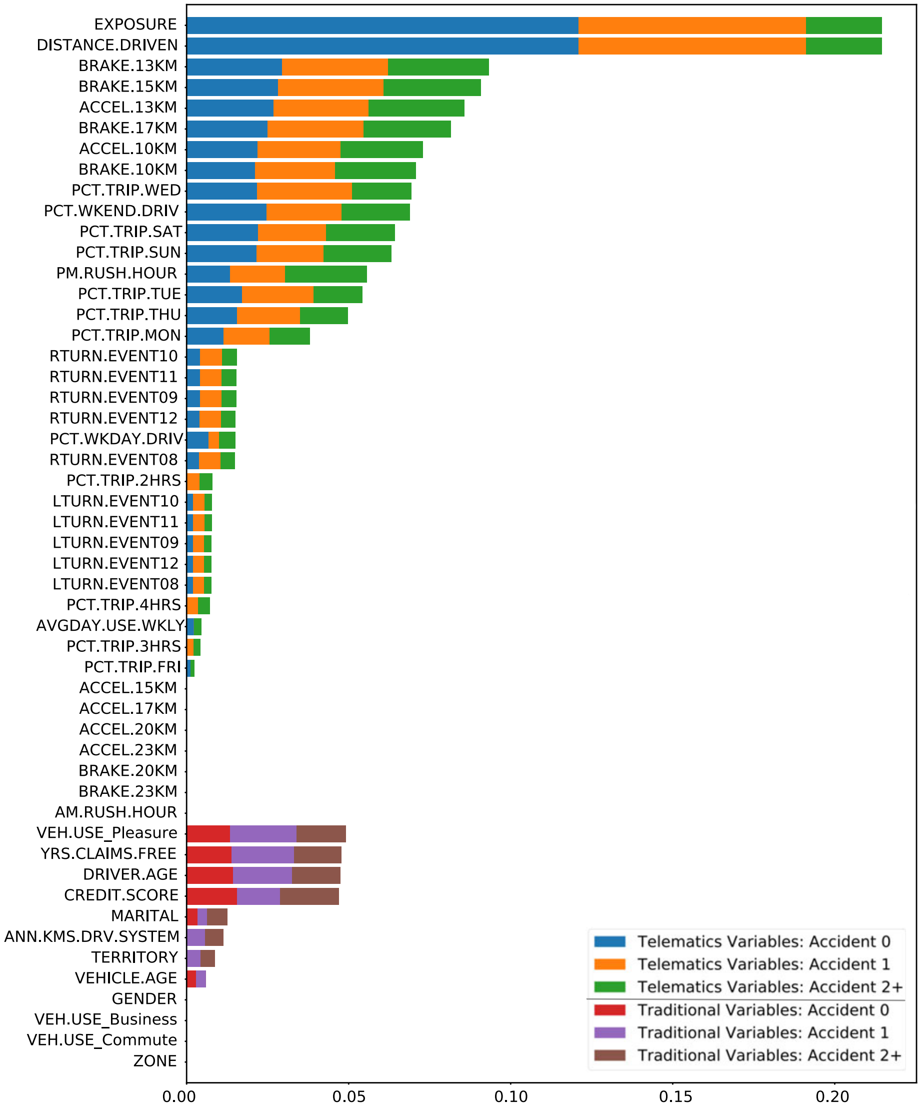


FIGURE 8: Feature importance based on the `SAMME.C2` predictions – using training data.

that there are a number of telematics variables that are better features of accident frequency than traditional variables. For example, the top five telematics features (EXPOSURE, DISTANCE.DRIVEN, BRAKE.13KM, BRAKE.15KM, and ACCEL.13KM) are more important than the top traditional features (VEH.USE_Pleasure and YRS.CLAIMS.FREE). This seems to indicate that traditional features that are usually considered significant could be replaced by information drawn from telematics. Second, the top two feature variables, EXPOSURE and DISTANCE.DRIVEN, far outweigh the others.

Because the degree of contribution to each frequency class is different, this does not necessarily imply that the more frequently one drives, the more likely it is that one will be in accidents more often.

Third, as described in Section 5.1, we can classify the telematics variables into those considered as driving maneuvers and those related to how much time a driver spends on the road. The fact that EXPOSURE and DISTANCE.DRIVEN dominated in terms of feature importance indicates that driving maneuvers are less important feature variables than time and distance traveled. However, some of the top six telematics variables (EXPOSURE, DISTANCE.DRIVEN, BRAKE.13KM, BRAKE. 15KM, ACCEL.13KM, and BRAKE.17KM) were related to driving maneuvers. Suppositionally, the former result can be explained by the effect of a change in driver behavior based on knowledge that constant monitoring is occurring when a telematics device is installed. This change would affect driving maneuvers more than it would with the time spent and distance traveled on the road. Finally, we found that gender had little effect in terms of predicting accident frequency. This is in line with previous studies, such as Ayuso *et al.* (2016), which suggest gender discrimination may not be necessary for telematics information. It may be that differences in driving behavior according to gender are reflected well by telematics information. However, we also note that this might only be what our data indicate and that there may be no significant difference based on gender irrespective of the model used.

Before we conclude this section, we want to express precaution regarding the use of our algorithm for pricing purposes. Throughout the paper, we have highlighted that `SAMME.C2` contains cost values that stress weights tilted toward learning preference from data in minority classes. Intuitively, this could lead to imbalances and possibly biased results when the algorithm is used to estimate accident frequency for premium determination. The results reported in Table 7 show rough estimates of the predicted number of accidents for the various algorithms compared in this paper (separately for the simulated and empirical data). Learning algorithms that place emphasis on minority classes clearly lead to higher estimated mean accident frequencies and such is true for `SAMME.C2` and the other similar models reported in the table. It should be noted that algorithms that do not place similar emphasis on minority classes, such as `SAMME` and `SMOTEBoost`, may lead to less-biased prediction results. The immediate concern with the direct use of classification techniques for insurance pricing is the possibility of overpricing certain groups of drivers and underpricing others. See Wüthrich and Buser (2020) for similar discussions.

However, the utility of `SAMME.C2` to understand and rank important features of driving behavior cannot be underestimated, especially when telematics information is included. Such feature selection can be used in *a posteriori* classification in insurance ratemaking, underwriting by introducing telematics-related queries during the selection process, and other post-pricing risk management techniques, such as insurance reserving. Insurance reserving is an important actuarial function used to determine the amount of funds necessary

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

TABLE 7

COMPARING THE ROUGHLY ESTIMATED NUMBER OF ACCIDENTS FOR THE VARIOUS LEARNING ALGORITHMS.

	Predicted number of accidents						
Dataset	SAMME	SAMME with SMOTE	SAMME with SVMSMOTE	SAMME with KMeansSMOTE	RUSBoost	SMOTEBoost	SAMME.C2
Simulated	0.039	0.101	0.087	0.155	0.057	0.175	0.155
Empirical	0.001	0.243	0.203	0.114	0.014	0.283	0.337

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

to cover future losses in as realistic a manner as possible. It is the responsibility of a company's chief actuary to ensure that estimated reserves, regardless of the purpose, are calculated based on the state of risk of the insurance portfolio at the valuation date. An investigation of how classification models can serve these purposes will be an interesting project for future work.

6. Concluding remarks

Over the past 5 to 6 years, there has been an influx of research related to vehicle telematics in actuarial science. This article expands on this literature, focusing on challenges not previously addressed and using an innovative classification model to handle imbalanced data. In particular, our work focuses on an understanding of how the addition of telematics information helps improve the recognition of driving behavior to improve understanding of accident frequency. We treated the modeling of accident frequency as a multi-class classification problem with highly imbalanced classes. As with our empirical data, we found that for motor insurance claim portfolios, a large number of drivers with zero accidents, a few with exactly one accident, and far fewer with two or more accidents are typically observed. We reviewed existing multi-class classification models that address the handling of the presence of minority classes and found that a combination of resampling procedures and boosting algorithms was suitable for our intentions. In particular, we propose an algorithm that is a combination of boosting and cost-sensitive learning, which we call SAMME.C2, to address the imbalances within multi-class classification problems. The injection of cost-sensitive factors has the effect of placing a heavy penalty on the misclassification of minority classes, while a heavy reward is simultaneously placed on the correct classifications of these classes. Further, we demonstrate how PCA methods can be employed to explore the nature of the strong dependencies common among telematics feature variables. The results of our model fitting to our telematics dataset are very promising. Overall, we conclusively demonstrate the superior performance of SAMME.C2 compared with other known boosting algorithms that are also combined with resampling.

When SAMME.C2 was applied to our telematics dataset, we are able to draw conclusive evidence on how telematics information affects accident frequencies. First, we found that telematics are significantly better features of accident frequency than typical classical variables used for risk classification (e.g., DRIVER.AGE and GENDER). This indicates that telematics variables can provide a better understanding of driver behavior. Second, we were able to group telematics data according to variables related to the percentage of time drivers spend on the road (e.g., EXPOSURE and DISTANCE.DRIVEN) and those related to driving maneuvers (e.g. BRAKE.13KM, BRAKE.15KM, and ACCEL.13KM). Broadly speaking, we found that the cluster of variables that describes the percentage of time drivers are on the road are more important feature variables than the cluster of those related to driving maneuvers. Two

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

observations about driving maneuvers can help us to explain this. First, the presence of a telematics device in the vehicle is, in an indirect sense, an encouragement of good driving behavior. Second, it is likely that drivers have more control over these driving maneuvers, whereas time spent on the road may be more a result of necessity (e.g., driving to school or work, driving family members to events, or driving for a family vacation). However, we did find evidence of moderate correlations between some of these driving maneuvers and accident frequency. In particular, the more frequently a driver was found to violate these driving maneuvers, the more likely they were to be in an accident. Modeling claim frequency to understand driving behavior is an important part of evaluating and classifying risks for pricing and reserving. However, many insurance companies still do not have the ability to gather telematics-related information. It is the hope of this paper that important telematics information can be collected in the form of proxy variables queried in an underwriting questionnaire. In the future, we aim to investigate the effects of telematics variables on claim severity and the damages associated with an accident. It will also be interesting to investigate the incorporation of classification techniques into insurance ratemaking.

Acknowledgments

The authors thank The Co-operators General Insurance Company for providing the empirical data. B. So acknowledges and thanks the support from the Society of Actuaries (SOA) through its James C. Hickman Scholar program. E.A. Valdez thanks the SOA for funding of the project on “Applying Data Mining Techniques in Actuarial Science” through its Centers of Actuarial Excellence (CAE) grant program. The authors thank the editor and anonymous referees for their helpful comments and suggestions that immensely improved the final version of this work.

References

Ayuso, M., Guillen, M. and Nielsen, J.P. (2019) Improving automobile insurance ratemaking using telematics: incorporating mileage and driver behaviour data. *Transportation* **46**, 735–752.

Ayuso, M., Guillen, M. and Pérez-Marín, A.M. (2016) Telematics and gender discrimination: some usage-based evidence on whether men’s risk of accidents differs from women’s. *Risks* **4**, 1–10.

Bhowan, U., Zhang, M. and Johnston, M. (2010) Genetic programming for classification with unbalanced data. *Proceedings 13th European Conference on Genetic Programming, EuroGP 2010*, pp. 1–13. Springer-Verlag Berlin.

Boucher, J.-P., Côté, S. and Guillen, M. (2017) Exposure as duration and distance in telematics motor insurance using generalized additive models. *Risks* **5**, 1–23.

Chawla, N.V., Bowyer, K.W., Hall, L.O. and Kegelmeyer, W.P. (2002) SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research* **16**, 321–357.

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

Chawla, N.V., Lazarevic, A., Hall, L.O. and Bowyer, K.W. (2003) SMOTEBoost: Improving prediction of the minority class in boosting. *PKDD 2003: Proceedings of the Seventh European Conference on Principles and Practice of Knowledge Discovery*, pp. 107–119. Springer-Verlag: Berlin-Heidelberg.

Constantinescu, C.C., Stancu, I. and Panait, I. (2018) Impact study of telematics auto insurance. *Review of Financial Studies* **3**(4), 17–35.

Douzas, G., Bacao, F. and Last, F. (2018) Improving imblanced learning through a heuristic oversampling method based on k-means and smote. *Information Sciences* **465**, 1–20.

Fernández, A., García, S., Galar, M., Prati, R.C., Krawczyk, B. and Herrera, F. (2018). *Learning from Imbalanced Data Sets*. Switzerland: Springer.

Ferrario, A. and Hämmerli, R. (2019) On Boosting: Theory and Applications. Available at SSRN: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3402687

Ferreira, A.J. and Figueiredo, M.A. (2012) Boosting algorithms: A review of methods, theory, and applications. In *Ensemble Machine Learning: Methods and Applications* (eds. C. Zhang and Y. Ma), chap. 2, pp. 35–85. Springer Science.

Fowlkes, E.B. and Mallows, C. (1983) A method for comparing two hierarchical clusterings. *Journal of the American Statistical Association* **78**(383), 553–569.

Freund, Y. and Schapire, R.E. (1997) A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences* **55**(1), 119–139.

Friedman, J., Hastie, T. and Tibshirani, R. (2000) Additive logistic regression: A statistical view of boosting. *The Annals of Statistics* **28**(2), 337–407.

Galar, M., Fernández, A., Barrenechea, E., Bustince, H. and Herrer, F. (2012) A review on emsembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Review* **42**(4), 463–484.

Gao, G., Meng, S. and Wüthrich, M.V. (2019) Claims frequency modeling using telematics card driving data. *Scandinavian Actuarial Journal* **2**, 143–162.

Gao, G., Wang, H. and Wüthrich. M.V. (2021) Boosting poisson regression models with telematics car driving data. *Machine Learning*.

Guillen, M., Nielsen, J.P., Pérez-Marín, A.M. and Elpidorou, V. (2020) Can automobile insurance telematics predict the risk of near-miss events? *North American Actuarial Journal* **24**(1), 141–152.

Hand, D.J. and Till, R.J. (2001) A simple generalisation of the area under the ROC curve for multiple class classification problems. *Machine Learning* **45**(2), 171–186.

Hastie, T., Tibshirani, R. and Friedman, J. (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer.

Holland, J.H. (1975) *Adaptation in Natural and Artifical Systems*. Ann Arbor: Univesity of Michigan Press.

Mühlenbein, H. (1997) Genetic algorithms. In *Local Search in Combinatorial Optimization* (eds. E.H. Aarts and J.K. Lenstra), pp. 137–172. Princeton University Press.

Orphanoudakis, S.C., Chronaki, C.E., Tsiknakis, M. and Kostomanolakis, S.G. (1998) Telematics in healthcare. In *Medical Image Databses* (ed. S.T. Wong), chap. 10, pp. 251–281. New York: Springer.

Pazzani, M., Merz, C., Murphy, P., Ali, K., Hume, T. and Brunk, C. (1994) Reducing misclassification costs. *ICML 1994: Proceedings of the Eleventh International Conference on Machine Learning*, pp. 217–225. San Francisco, CA: Morgan Kaufman Publishers Inc..

Pednault, E.P., Rosen, B.K. and Apte, C. (2000) Handling imbalanced data sets in insurance risk modeling. Technical report, Association for the Advancement of Artificial Intelligence (AAAI).

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. and Duchesnay, E. (2011) Scikit-learn: machine learning in Python. *Journal of Machine Learning Research* **12**, 2825–2830.

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

Pérez-Marín, A. M., Guillen, M., Alcañiz, M. and Bermúdez, L. (2019) Quantile regression with telematics information to assess the risk of driving above the posted speed limit. *Risks* **7**, 1–11.

Pesantez-Narvaez, J., Guillen, M. and Alcañiz, M. (2019) Predicting motor insurance claims using telematics data – XGBoost versus logistic regression. *Risks* **7**, 1–16.

Schapire, R.E. and Singer, Y. (1999) Using boosting algorithms using confidence-rated predictions. *Machine Learning* **37**, 297–336.

Seiffert, C., Khoshgoftaar, T.M., Van Hulse, J. and Napolitano, A. (2010) RUSBoost: A hybrid approach to alleviating class imbalance. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* **40**(1), 185–197.

Shon, H.S., Batbaatar, E., Kim, K.O., Cha, E.J. and Kim, K.-A. (2020) Classification of kidney cancer data using cost-sensitive hybrid deep learning approach. *Symmetry* **12**, 154.

Sun, Y., Kamel, M.S., Wong, A.K. and Wang, Y. (2007) Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition* **40**(12), 3358–3378.

Tang, Y., Zhang, Y.-Q., Chawla, N.V. and Krasser, S. (2009) SVMs modeling for highly imbalanced classification. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, **39**(1), 281–288.

Verbelen, R., Antonio, K. and Claeskens, G. (2018) Unravelling the predictive power of telematics data in car insurance pricing. *Journal of the Royal Statistical Society: Series C (Applied Statistics)* **67**(5), 1275–1304.

Wüthrich, M.V. and Buser, C. (2020) Data analytics for non-life insurance pricing. Available at SSRN: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2870308

Yang, Q. and Wu, X. (2006) 10 challenging problems in data mining research. *International Journal of Information Technology & Decision Making* **5**(4), 597–604.

Zhang, S. (2020) Cost-sensitive KNN classification. *Neurocomputing* **391**, 234–242.

Zhu, J., Zou, H., Rossett, S. and Hastie, T. (2009) Multi-class AdaBoost. *Statistics and Its Interface*, **2**, 349–360.

Banghee So
Department of Mathematics
Towson University, 7800 York Rd
Towson, MD, 21252, USA
E-Mail: bso@towson.edu

Jean-Philippe Boucher
Département de Mathématiques
Université du Québec à Montréal, 201
Avenue du Président-Kennedy
Montréal, Québec, H2X 3Y7, Canada
E-Mail: boucher.jean-philippe@uqam.ca

Emiliano A. Valdez (Corresponding author)
Department of Mathematics
University of Connecticut
341 Mansfield Road, Storrs, CT, 06269-1009, USA
E-Mail: emiliano.valdez@uconn.edu

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

Appendix A. Detailed steps of the `SAMME` and `Ada.C2` algorithms

Algorithm 2: `SAMME`: multi-class AdaBoost

```
Input: Training dataset $\boldsymbol{x}_i \in X, y_i \in Y = \{1, 2, \ldots, K\}, T$
Output: Final classifier $H(\boldsymbol{x}_i)$
Set initial distribution of dataset equally distributed:
$D_1(i) = \frac{1}{N}, \quad i = 1, 2, \ldots, N$ ;
for $t = 1, \ldots, T$ do
    Train weak classifier using the distribution $D_t$ ;
    Get weak classifier $h_t : X \to k \in \{1, 2, \ldots, K\}$ ;
    Compute $\epsilon_t = \frac{\sum_{i=1}^{N} D_t(i) I(y_i \neq h_t(\boldsymbol{x}_i))}{\sum_{i=1}^{N} D_t(i)}$ ;
    Choose $\alpha_t = \log\left(\frac{1-\epsilon_t}{\epsilon_t}\right) + \log(K-1)$ ;
    Update $D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t I(y_i = h_t(\boldsymbol{x}_i)))}{\sum_{j=1}^{N} D_t(j) \exp(-\alpha_t I(y_j = h_t(\boldsymbol{x}_j)))}$ ;
end
Return the final classifier: $H(\boldsymbol{x}_i) = \underset{k}{\operatorname{argmax}} \sum_{t=1}^{T} \alpha_t I(h_t(\boldsymbol{x}_i) = k)$ ;
```

Algorithm 3: `Ada.C2`: cost-sensitive binary AdaBoost

```
Input: Training dataset $\boldsymbol{x}_i \in X, y_i \in Y = \{0, 1\}, C(y_i), T$
Output: Final classifier $H(\boldsymbol{x}_i)$
Set initial distribution of dataset equally distributed:
$D_1(i) = \frac{1}{N}, \quad i = 1, 2, \ldots, N$ ;
for $t = 1, \ldots, T$ do
    Train weak classifier using the distribution $D_t$ ;
    Get weak classifier $h_t : X \to k \in \{0, 1\}$ ;
    Compute $\epsilon_t = \frac{\sum_{i=1}^{N} C(y_i) D_t(i) I(y_i \neq h_t(\boldsymbol{x}_i))}{\sum_{i=1}^{N} C(y_i) D_t(i)}$ ;
    Choose $\alpha_t = \frac{1}{2} \log\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$ ;
    Update $D_{t+1}(i) = \frac{C(y_i)\, D_t(i) \exp(-\alpha_t I(y_i = h_t(\boldsymbol{x}_i)))}{\sum_{j=1}^{N} C(y_j)\, D_t(j) \exp(-\alpha_t I(y_j = h_t(\boldsymbol{x}_j)))}$ ;
end
Return the final classifier: $H(\boldsymbol{x}_i) = \underset{k}{\operatorname{argmax}} \sum_{t=1}^{T} \alpha_t I(h_t(\boldsymbol{x}_i) = k)$ ;
```

https://doi.org/10.1017/asb.2021.22 Published online by Cambridge University Press

Appendix B. Traditional and telematics variables in the dataset

TABLE 8

VARIABLE NAMES AND DESCRIPTIONS.

Type	Variable	Description
Traditional	DRIVER.AGE	Age of driver
	GENDER	Gender of the driver (M/F)
	VEHICLE.AGE	Vehicle age
	MARITAL	Marital status
	VEH.USE	Use of vehicle: Pleasure, Commute, Farmer, Business
	CREDIT.SCORE	Credit score of driver
	ZONE	Zone where driver lives: rural, urban
	ANN.KMS.DRV.SYSTEM	Kilometer driven declared by driver
	YRS.CLAIMS.FREE	Number of years claims free
	TERRITORY	Territory where vehicle is rated
Telematics	EXPOSURE	Exposure time in percentage of 365 days
	DISTANCE.DRIVEN	Total distance driven
	PCT.TRIP.xxx	Percent of driving day xxx of week: MON/TUE/.../SUN
	PCT.TRIP.xxx	Percent vehicle driven in xxx hrs: 2HRS/3HRS/4HRS
	PCT.xxx.DRIV	Percent vehicle driven in xxx of week: WKDAY/WKEND
	xx.RUSH.HOUR	Percent of driving in xx rush hours: AM/PM
	AVGDAY.USE.WKLY	Average number of days used per week
	ACCEL.xxKM	Number of sudden acceleration 10/13/15.../23 km/h/s per 1000km
	BRAKE.xxKM	Number of sudden brakes 10/13/15.../23 km/h/s per 1000km
	LTURN.EVENTxx	Number of left turn per 1000km with intensity 08/09/10/11/12
	RTURN.EVENTxx	Number of right turn per 1000km with intensity 08/09/10/11/12
Response	ACC_FREQ	Frequency of accidents during observation: 0/1/2+

Appendix C. Confusion tables based on the telematics dataset: SAMME, SAMME with SMOTE, RUSBoost, SMOTEBoost

TABLE 9

SAMME.

		Actual			
		Acc 0	Acc 1	Acc 2^{+}	Tot row
Predicted	Acc 0	20,882	643	20	21,545
	Acc 1	19	7	3	29
	Acc 2^{+}	0	0	0	0
	Tot col	20,901	650	23	21,574

TABLE 10

SAMME WITH SMOTE.

		Actual			
		Acc 0	Acc 1	Acc 2^{+}	Tot row
Predicted	Acc 0	16,491	226	5	16,722
	Acc 1	3336	295	9	3640
	Acc 2^{+}	1074	129	9	1212
	Tot col	20,901	650	23	21,574

TABLE 11

RUSBOOST.

		Actual			
		Acc 0	Acc 1	Acc 2^{+}	Tot row
Predicted	Acc 0	14,558	333	6	14,897
	Acc 1	5983	282	15	6280
	Acc 2^{+}	360	35	2	397
	Tot col	20,901	650	23	21,574

TABLE 12

SMOTEBOOST.

		Actual			
		Acc 0	Acc 1	Acc 2^{+}	Tot row
Predicted	Acc 0	20,669	579	16	21,264
	Acc 1	209	67	2	278
	Acc 2^{+}	23	4	5	32
	Tot col	20,901	650	23	21,574