Imbalanced Classification: From a robust optimization view

Dimitris Bertsimas dbertsim@mit.edu
Sloan School of Management and Operations Research Center
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

Yuchen Wang yuchenw@mit.edu
Operations Research Center
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

Editor:

Abstract

Classification on imbalanced datasets is usually a hard task in machine learning. There are already many methods to solve this problem, but they either delete some data or generate some data artificially. In this paper, we revisit the imbalanced classification problem from a robust optimization view without generating or deleting data. We compare the robust method, state-of-the-art oversampling and undersampling methods on 20 real-world datasets. We show the robust method has a performance edge over all other methods for both logistic regression and support vector machines (SVM). We also illustrate the balanced dataset generated by the robust method can be used to improve the performance of other machine learning methods like classification trees on imbalanced datasets.

Keywords: Robust Optimization, Imbalanced datasets, Classification

1. Introduction

For classification problems, there are some real-world datasets in which the number of samples of a given class is much smaller than the number of samples in other classes. This imbalance leads to the so-called "class imbalance" problem, one of the top 10 problems in data mining (Sun et al. (2009); Yang and Wu (2006)). For a binary classification problem, the class with more data is called the majority class, and the rest of the classes are called minority classes (Chawla et al. (2003)). The ratio between the majority class and minority classes is called the Imbalance Ratio (IR) (Chawla et al. (2002)). The imbalance problem occurs in a variety of areas from fraud detection to health care. For example, the ratio between children with clinically important traumatic brain injury (ciTBI) and children without is 1:112 (Bertsimas et al. (2019b)). The imbalance ratio can also be about 100 for fraud detection. (Provost and Fawcett (2001)). For these datasets, we need some methods to improve the accuracy of predicting minority class.

There are already several methods proposed to solve this problem. The first type of method is to increase the weights of the minority class in the loss function. For example, weighted logistic regression is to set the weight of minority class to be the imbalance ratio. By emphasizing minority class more in this way, the model will predict more accurately on minority class.

The second type of method called the oversampling method is to generate more minority class samples in order to convert an imbalanced dataset to a balanced dataset. Synthetic Minority Oversampling Technique (SMOTE) (Chawla et al. (2002)) is the most well-known algorithm of this type. They generate more minority class samples by introducing synthetic examples along with the line segments of the k minority class nearest neighbors. Batista et al. (2004) improves the quality of synthetic examples of SMOTE by removing misclassified instances. He et al. (2008) generates more synthetic data for minority class samples that are harder to learn and generate less synthetic for easier minority class samples. Douzas and Bacao (2019b) is the state-of-art oversampling method that finds a geometric region around each minority class samples where synthetic data are generated inside this region. However, the oversampling method increases the number of training samples that increases the training time of the model. It may also overfit on test set because of emphasizing too much on specific minority samples.

The third type of method called the undersampling method is to decrease the number of majority class to convert an imbalanced dataset to a balanced dataset. Majority samples are removed randomly or based on their distance to minority samples (Kubat et al. (1997)). Laurikkala (2001) proposes a neighborhood cleaning rule utilizing edited nearest-neighbor rule to identify noisy data in majority class. Instance hardness threshold (IHT) (Smith et al. (2014)) is to train a classier on the data first, and then the samples with lower probabilities are removed. However, the oversampling method may result in loss of information and discards potentially valuable data.

Recently, with the development of robust optimization, it was used more for classification problems. Pant et al. (2011) first utilized robust optimization to classification problem. They consider the bounded uncertainties of the data to improves the performance of SVM. Bertsimas et al. (2019a) considered the uncertainties in features, in labels on logistic regression, SVM and classification trees. However, these work only use robust optimization to deal with the uncertainties in the data and mainly focus on the general classification problem but not imbalanced datasets.

In this paper, we revisit the imbalance classification from a robust optimization view in order to solve the problem mentioned above. First, we illustrate how to use robust optimization to construct a balanced training set for both Logistic regression and SVM. Then, we show that the robust optimization approach improves the out-of-sample F1-score and AUC for Logistic regression and SVM. We also show the robust method is better than other oversampling and undersampling methods. In the end, we show the balanced training set generated by the robust method can also be utilized to improve the performance of Optimal Trees.

The paper is structured as follows. In Section 2, we describe how to use robust optimization on logistic regression and SVM. In section 3, we present computational results on 20 real-world datasets and compare it with other methods, including weighted logistic regression/SVM, IHT and GSMOTE. We conclude in Section 4.

2. Methods

In this section, we first describe how we apply robust optimization techniques to train logistic regression and SVM on a balanced dataset. Then we explain how to validate this

algorithm. In the end, we illustrate how to use the balanced datasets we generate to improve the performance of other classification methods like Optimal Trees.

We are given data $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, n$ with $\mathbf{x}_i \in \mathbb{R}^p$ and $y_i \in [0, 1]$. We normalize the data by applying the transformation

$$\hat{x}_{ij} := \frac{x_{ij} - m_j}{\sigma_j},\ i \in [n],\ j \in [p], \tag{1}$$

where m_j and σ_j are the mean and standard deviation of dimension j of the training samples. We define $[n] = 1, \ldots, n$.

Assume there are n_1 class 1 samples and n_0 class 0 samples. And the original dataset is consists of class 0 data $x_i^0, i \in [n_0]$ and class 1 data $x_j^1, j \in [n_1]$. For imbalanced datasets we consider, n_0 is much larger than n_1 and imbalance ratio is defined as $\frac{n_0}{n_1}$.

2.1 Logistic Regression

Logistic regression is one of the most widely used binary classification methods which can be solve efficiently for large-scale datasets (Friedman et al. (2010)). The original logistic regression problem is trying to find coefficients $\beta \in \mathbb{R}^p, \beta_0 \in \mathbb{R}$ by solving the following concave maximization problem:

$$\max_{\beta,\beta_0} \left[-\sum_{i=1}^{n_0} \log(1 + e^{(\beta x_i^0 + \beta_0)}) - \sum_{j=1}^{n_1} \log(1 + e^{-(\beta x_j^1 + \beta_0)}) \right].$$

Similar to the regularization term in the popular ridge regression Hoerl and Kennard (1970), a norm 2 regularization term can be added to the original objective function

$$\max_{\beta,\beta_0} \left[-\sum_{i=1}^{n_0} \log(1 + e^{(\beta x_i^0 + \beta_0)}) - \sum_{j=1}^{n_1} \log(1 + e^{-(\beta x_j^1 + \beta_0)}) \right] - \lambda ||\beta||_2^2.$$

Normally, we will first randomly assigning data to the training and validation sets. Then we decide the hyper-parameter by using validation sets performance. In the end, we use the selected hyper-parameter to retrain the regularized logistic regression on the combination of training and validation sets. However, instead of randomly assigning data to training sets and use all the data, we introduce binary variables $u_i, i \in [n_0]$ and $v_j, j \in [n_1]$ to decide whether we use data x_i^0 and x_j^1 in the training set. In order to make sure we use a balanced training set, we add constraints

$$\sum_{j=1}^{n_1} v_j = tn_1, \quad \sum_{i=1}^{n_0} u_i = tkn_1,$$

where k is a hyper-parameter to decide the portion between class 0 and class 1, t is the portion of training set in whole dataset. So the first constraint means the number of class 1 data we use in training set is t of the whole dataset. The second constraint means the number of class 0 samples we use in training set is k times the number of class 1 samples we use. Instead of randomly assigning data to training sets, we combine this selection process

into with the optimization directly which gives us the following optimization problem to solve:

$$\max_{\beta,\beta_0} \min_{u\in\mathcal{U},v\in\mathcal{V}} \left[-\sum_{i=1}^{n_0} u_i \log(1+e^{(\beta x_i^0+\beta_0)}) - \sum_{j=1}^{n_1} v_j \log(1+e^{-(\beta x_j^1+\beta_0)}) \right] - \lambda||\beta||_2^2$$

with

$$\mathcal{U} = \left\{ u : \sum_{i=1}^{n_0} u_i = tkn_1, u_i \in \{0,1\} \right\}, \mathcal{V} = \left\{ \sum_{j=1}^{n_1} v_j = tn_1, v_j \in \{0,1\} \right\}.$$

As the inner minimization problem is linear in u and v, this problem is equivalent to optimizing over the convex hull of $\mathcal{U}$ and $\mathcal{V}$

$$\text{conv}(\mathcal{U}) = \left\{ u : \sum_{i=1}^{n_0} u_i = tkn_1, 0 \le u_i \le 1 \right\}, \text{conv}(\mathcal{V}) = \left\{ v : \sum_{i=1}^{n_1} v_j = tn_1, 0 \le v_i \le 1 \right\}.$$

So the original problem is equivalent to

$$\max_{\beta,\beta_0} \min_{u\in\text{conv}(\mathcal{U}),v\in\text{conv}(\mathcal{V})} \left[-\sum_{i=1}^{n_0} u_i \log(1+e^{(\beta x_i^0+\beta_0)}) - \sum_{j=1}^{n_1} v_j \log(1+e^{-(\beta x_j^1+\beta_0)}) \right] - \lambda||\beta||_2^2$$

with

$$\text{conv}(\mathcal{U}) = \left\{ u : \sum_{i=1}^{n_0} u_i = tkn_1, 0 \le u_i \le 1 \right\}, \text{conv}(\mathcal{V}) = \left\{ v : \sum_{i=1}^{n_1} v_j = tn_1, 0 \le v_i \le 1 \right\}.$$

This problem belongs to the robust optimization problem. It achieves the following two things compared to the nominal problem at same time:

1. It is trained on a balanced dataset with ratio k between class 0 and class 1 samples.

2. It finds the "hardest" training set satisfying the first condition.

In order to solve this problem, we calculate the robust counterpart of the inner problem

$$\begin{aligned} &\min_{u,v} \left[-\sum_{i=1}^{n_0} u_i \log(1+e^{(\beta x_i^0+\beta_0)}) - \sum_{j=1}^{n_1} v_j \log(1+e^{-(\beta x_j^1+\beta_0)}) \right] \\ &\text{s.t.} \quad \sum_{i=1}^{n_0} u_i = ktn_1, \quad \sum_{j=1}^{n_1} v_j = tn_1, \\ &\qquad 0 \le u_i, v_j \le 1, \quad i \in [n_0], \quad j \in [n_1]. \end{aligned} \tag{2}$$

by introducing the dual variables θ_0 for the first constraint, θ_1 for the second constraint and dual variable p_i, q_i for the third set of constraints to get

$$\begin{aligned}\max_{\theta_0,\theta_1,p,q} \quad & ktn_1\theta_0 + tn_1\theta_1 + \sum_{i=1}^{n_0} p_i + \sum_{j=1}^{n_1} q_j \\ \text{s.t.} \quad & \theta_0 + p_i \le -\log(1 + e^{(\beta x_i^0 + \beta_0)}), \quad i \in [n_0], \\ & \theta_1 + q_j \le -\log(1 + e^{-(\beta x_j^1 + \beta_0)}), \quad j \in [n_1], \\ & p_i, q_j \le 0, \quad i \in [n_0], \quad j \in [n_1].\end{aligned}$$

Substituting this inner maximization problem back into the outer maximization we get the following optimization problem to solve:

$$\begin{aligned}\max_{\beta,\beta_0,\theta_0,\theta_1,p,q} \quad & ktn_1\theta_0 + tn_1\theta_1 + \sum_{i=1}^{n_0} p_i + \sum_{j=1}^{n_1} q_j - \lambda||\beta||_2^2 \\ \text{s.t.} \quad & \theta_0 + p_i \le -\log(1 + e^{(\beta x_i^0 + \beta_0)}), \quad i \in [n_0], \\ & \theta_1 + q_j \le -\log(1 + e^{-(\beta x_j^1 + \beta_0)}), \quad j \in [n_1], \\ & p_i, q_j \le 0, \quad i \in [n_0], \quad j \in [n_1]. \end{aligned} \tag{3}$$

Because this problem has a twice continuously differentiable concave objective function and constraints so we can solve it with an interior point method (Bertsekas (1999)).

In some extreme cases, the optimal solution of problem (3) will be $\beta = 0$ which is meaningless. This is because the model finds this very conservative solution to make sure it achieves good result on the hardest set. So we add the following constraints to the original problem and solve it again:

$$\begin{aligned}\max_{\beta,\beta_0,\theta_0,\theta_1,p,q} \quad & ktn_1\theta_0 + tn_1\theta_1 + \sum_{i=1}^{n_0} p_i + \sum_{j=1}^{n_1} q_j - \lambda||\beta||_2^2 \\ \text{s.t.} \quad & \theta_0 + p_i \le -\log(1 + e^{(\beta x_i^0 + \beta_0)}), \quad i \in [n_0], \\ & \theta_1 + q_j \le -\log(1 + e^{-(\beta x_j^1 + \beta_0)}), \quad j \in [n_1], \\ & \beta_k - \hat{\beta}_k \le \delta, \quad \beta_0 - \hat{\beta}_0 \le \delta, \quad k \in [p], \\ & \hat{\beta}_k - \beta_k \le \delta, \quad \hat{\beta}_0 - \beta_0 \le \delta, \quad k \in [p], \\ & p_i, q_j \le 0, \quad i \in [n_0], \quad j \in [n_1] \end{aligned} \tag{4}$$

where $\hat{w}, \hat{b}$ is the solution of weighted logistic regression and δ is a small number to guarantee the solution found by the robust optimization is not 0.

2.2 Support Vector Machines

Support vector machines are another widely used binary classification methods proposed by Cortes and Vapnik (1995). We consider the hinge loss classifier with regularization term

to be the nominal method for SVM with linear kernel:

$$\min_{w,b}\left[\sum_{i=1}^{n_0}\max\left\{1+w'x_i-b,0\right\}+\sum_{j=1}^{n_1}C\max\left\{1-w'x_j+b,0\right\}\right]+\lambda||w||_2^2$$

where C is weight of class 1 samples.

After adding constraints to select the proper samples for training like logistic regression, the problem we are going to solve is

$$\min_{w,b}\max_{u\in\mathcal{U},v\in\mathcal{V}}\left[\sum_{i=1}^{n_0}u_i\max\left\{1+w'x_i-b,0\right\}+\sum_{j=1}^{n_1}Cv_j\max\left\{1-w'x_j+b,0\right\}\right]+\lambda||w||_2^2$$

with

$$\mathcal{U}=\left\{u:\sum_{i=1}^{n_0}u_i=tkn_1,u_i\in\{0,1\}\right\},\mathcal{V}=\left\{v:\sum_{j=1}^{n_1}v_j=tn_1,v_j\in\{0,1\}\right\}.$$

By replacing set $\mathcal{U}$ and $\mathcal{V}$ with their convex hull, the optimization problem is equivalent to:

$$\begin{aligned}\min_{w,b}\max_{u,v}\ &\left[\sum_{i=1}^{n_0}u_i\max\left\{1+w'x_i-b,0\right\}+\sum_{j=1}^{n_1}Cv_j\max\left\{1-w'x_j+b,0\right\}\right]+\lambda||\beta||_2^2\\ \text{s.t.}\ &\sum_{i=1}^{n_0}u_i=tkn_1,\quad\sum_{j=1}^{n_1}v_j=tn_1,\quad 0\le u_i,v_j\le 1.\end{aligned}$$

In order to solve this problem, we calculate the robust counterpart of the inner problem

$$\begin{aligned}\max_{u,v}\ &\left[\sum_{i=1}^{n_0}u_i\max\left\{1+w'x_i-b,0\right\}+\sum_{j=1}^{n_1}Cv_j\max\left\{1-w'x_j+b,0\right\}\right]\\ \text{s.t.}\ &\sum_{i=1}^{n_0}u_i=tkn_1,\quad\sum_{j=1}^{n_1}v_j=tn_1,\quad 0\le u_i,v_j\le 1\end{aligned}$$

by introducing the dual variables θ_0 for the first constraint, θ_1 for the second constraint and dual variable p_i,q_i for the third set of constraints to get

$$\begin{aligned}\min_{w,b}\ &\sum_{i=1}^{n_0}p_i+\sum_{j=1}^{n_1}q_j+tkn_1\theta_0+tn_1\theta_1\\ \text{s.t.}\ &p_i+\theta_0\ge 1+w'x_i-b,\quad i\in[n_0],\\ &p_i+\theta_0\ge 0,\quad i\in[n_0],\\ &q_j+\theta_1\ge C(1-w'x_j+b),\quad j\in[n_1],\\ &q_j+\theta_1\ge 0,\quad j\in[n_1],\\ &p_i,q_j\ge 0,\quad i\in[n_0],\quad j\in[n_1].\end{aligned}$$

Substituting this inner minimization problem back into the outer minimization we get the following optimization problem to solve:

$$\begin{aligned}
\min_{w,b} \quad & \sum_{i=1}^{n_0} p_i + \sum_{j=1}^{n_1} q_j + tkn_1\theta_0 + tn_1\theta_1 + \lambda||w||_2^2 \\
\text{s.t.} \quad & p_i + \theta_0 \geq 1 + w'x_i - b, \quad i \in [n_0], \\
& p_i + \theta_0 \geq 0, \quad i \in [n_0], \\
& q_j + \theta_1 \geq C(1 - w'x_j + b), \quad j =\in [n_1], \\
& q_j + \theta_1 \geq 0, \quad j \in [n_1], \\
& p_i, q_j \geq 0, \quad i \in [n_0], \quad j \in [n_1]
\end{aligned} \tag{5}$$

which is a linear optimization problem.

Similarly to logistic regression, if the optimal solution of problem (5) is $w = 0$. We add the following constraints to the original problem and solve it again:

$$\begin{aligned}
\min_{w,b} \quad & \sum_{i=1}^{n_0} p_i + \sum_{j=1}^{n_1} q_j + tkn_1\theta_0 + tn_1\theta_1 + \lambda||w||_2^2 \\
\text{s.t.} \quad & p_i + \theta_0 \geq 1 + w'x_i - b, \quad i \in [n_0], \\
& p_i + \theta_0 \geq 0, \quad i \in [n_0], \\
& q_j + \theta_1 \geq C(1 - w'x_j + b), \quad j =\in [n_1], \\
& q_j + \theta_1 \geq 0, \quad j \in [n_1], \\
& w_k - \hat{w}_k \leq \delta, \quad b - \hat{b} \leq \delta, \quad k \in [p], \\
& \hat{w}_k - w_k \leq \delta, \quad \hat{b} - b \leq \delta, \quad k \in [p], \\
& p_i, q_j \geq 0, \quad i \in [n_0], \quad j \in [n_1]
\end{aligned} \tag{6}$$

2.3 Validation

Because we use a special way to select samples for training, the validation process will also be different to traditional one in order to maximize the out-of-sample performance. The whole process for training robust logistic regression is shown as follows.

Algorithm 1 Robust Logistic Regression for Imbalanced Datasets

1: Assume the whole dataset is d. We randomly partition d into two parts: the training d_{train} (75%) and test set d_{test} (25%).
2: Select potential parameter $k \in \{k_1, k_2, k_3, \ldots, k_i\}$ to tune over.
3: **for** $k = k_1, \ldots, k_i$ **do**
4: Solve problem 3 with parameter k_i and $t = 0.75$.
5: If the solution of the optimization problem we just solved is trivial (β=0), we solve problem 4 with parameter k_i instead.
6: **end for**
7: Identify the solution k^* with largest F1-score on the whole imbalanced training set d_{train}.
8: Solve the problem 3 with selected parameter k^* and $t = 1$.
9: If the solution of the optimization problem we just solved is trivial (β=0), we solve problem 4 with parameter k^* instead.
10: Return the parameter β, β_0.

For SVM, we also use the same validation process. The only difference is to change the criterion $\beta = 0$ to $w = 0$.

In the end of this section, we want to point out after we find the parameter β, β_0 from the above algorithm. We can also find the corresponding balanced set we used by solving optimization problem 2. This balanced dataset can be also utilized to improve the performance of other classification methods like Optimal Trees. In section 3, we will show how the balanced training sets improve the performance of Optimal Trees model.

3. Computation experiments

In this section, we report the performance of robust methods on a variety of imbalanced real-world datasets. We also compare the robust methods with oversampling and undersampling methods in these datasets across two metrics: Area Under Curve (AUC) and F1-score. AUC is defined as the area under the Receiver Operating Characteristic (ROC) curve which usually used to measure the ability of a binary classifier system as its prediction threshold changes (Bradley (1997)). F1-score is another measure of binary classifier's accuracy which is defined as the harmonic mean of precision and recall (Chinchor (1992)).

3.1 Experiment Setup

We select 20 classification datasets from the UCI Machine Learning Repository (Dua and Graff (2017)). The datasets were selected with different sizes and imbalanced ratio. The imbalance ratio (IR) chancing from 3.7 to 33.7, and the number of observations changes from 79 to 245,057.

In order to create imbalanced datasets, we consider the one-versus-rest problem of predicting the occurrence of the minority class in the data set. The minority class is listed behind the dataset name. For example, for datasets with name flag-0, we want to predict class 0 in this dataset. For non-robust methods, we partition each dataset into three parts: the training (50%), validation (25%), and testing set (25%). First, we train the model using the training set with different hyper-parameters. Then, we select the best hyper-parameters

using validation set F1-score. After that, we retrain the model using the selected hyper-parameters on the combined training and validation sets. In the end, we report the AUC and F1-score on testing set. For robust methods, we combine the training and validation set as new training set and use the validation process described in last section to find the final model. In the end, we report the AUC and F1-score on testing set. All methods are tested on the same random splits, and the experiments are repeated five times for each dataset with different random splits. We report the average test set AUC and F1-score across all five splits.

In these real-world datasets, we consider three classification methods: logistic regression, SVM and Optimal Trees. For logistic regression and SVM, we use the implementation from Pedregosa et al. (2011). For optimal trees, we use the implementation from Interpretable AI's Julia Interface (Interpretable AI (2019)). We implemented all robust methods using the JuMP software package in JULIA (Lubin and Dunning (2015)) and the solver IPOPT (Wächter and Biegler (2006)). For oversampling method GSMOTE, we use the implementation from Douzas and Bacao (2019a). For undersampling method IHT, we use the implementation from Lemaître et al. (2017).

For logistic regression and SVM, we set the regularized parameter λ to be 1 and the weight to be balanced. For Optimal Trees, we tune the depth from 1 to 8. For GSMOTE, we tune the the number of nearest neighbors $k \in \{3, 5\}$, the truncation factor from $\{-1.0, -0.5, 0, 0.25, 0.5, 0.75, 1\}$ and the deformation factor from $\{0.0, 0.2, 0.4, 0.6, 0.8, 1\}$. For Robust methods, we tune the ratio k from $\left\{1, \frac{1}{4}IR, \frac{1}{2}IR, \frac{3}{4}IR\right\}$.

3.2 Comparisons

First, we compare the F1-score of all methods. Table 1, 2, 3 show the out-of-sample F1-score for logistic regression, SVM and Optimal Trees correspondingly. Rank 1 indicates the method had the best performance on a dataset, and Rank 4 indicates the method performed worst. The column with head weighted represents the performance of weighted logistic regression, SVM, or Optimal Trees. The column with head undersample represents the performance of IHT, and the column with head oversample represents the performance of GSMOTE. The column with head robust represents the method proposed in this paper. For logistic regression, the robust method increases out-of-sample F1-score by a 0.059 (10.3%) on average compared to weighted logistic regression and a 0.06 (9.2%) on average compared to oversampling method GSMOTE. For SVM, the robust method increases out-of-sample F1-score by a 0.035 (6.1%) on average compared to weighted SVM and a 0.037 (6.4%) on average compared to oversampling method GSMOTE. For Optimal Trees, the robust method increases the out-of-sample F1-score by a 0.013 (2.1%) on average compared to weighted Optimal Trees and a 0.028 (4.7%) on average compared to oversampling method GSMOTE. Figure 1 shows the F1-score rank for different methods which illustrates the robust method is the strongest method in terms of average rank, followed by GSMOTE and the weighted method.

Next, we compare the AUC of all methods. Table 4, 5, 6 show the out-of-sample AUC for logistic regression, SVM and Optimal Trees correspondingly. For logistic regression, the robust method increases out-of-sample AUC by a 0.011 (1.3%) on average compared to weighted logistic regression and a 0.008 (0.9%) on average compared to oversampling

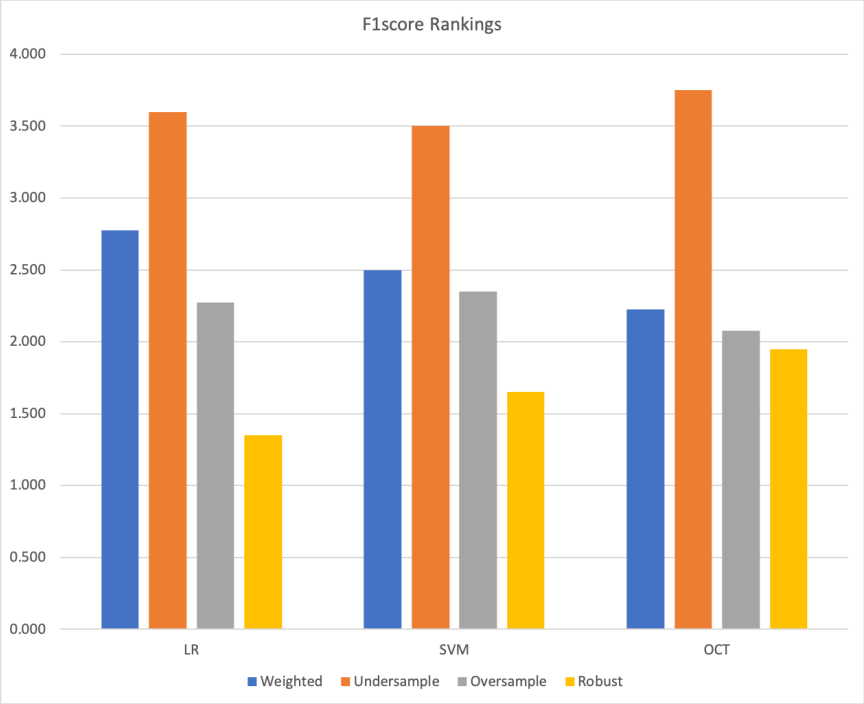

Figure 1: F1-score rank for different methods

UCI Dataset Name	n	p	IR	Weighted	Undersample	Oversample	Robust
hepatitis	79	20	5.1	0.638 (2)	0.470 (4)	0.593 (3)	0.671 (1)
fertility	99	13	7.2	0.182 (4)	0.210 (2)	0.254 (1)	0.19 (3)
flags-0	193	60	3.8	0.665 (3)	0.518 (4)	0.674 (2)	0.703 (1)
image-segmentation-path	209	20	6	0.935 (3)	0.716 (4)	0.953 (2)	1 (1)
dermatology-5	357	35	6.4	1 (1.5)	0.552 (4)	1 (1.5)	0.992 (3)
libras-move	360	91	14	0.827 (3)	0.214 (4)	0.849 (1)	0.829 (2)
thoracic-surgery	469	25	5.7	0.25 (3)	0.269 (1)	0.24 (4)	0.258 (2)
climate-model-simulation-crashes	539	19	11	0.551 (2)	0.308 (4)	0.536 (3)	0.727 (1)
connectionist-bench-10	989	11	10	0.194 (4)	0.213 (1)	0.195 (3)	0.212 (2)
yeast-me-2	1484	9	28.1	0.209 (3)	0.083 (4)	0.218 (2)	0.311 (1)
car-eval-34	1728	22	11.9	0.842 (3)	0.243 (4)	0.883 (2)	0.923 (1)
ozone-level	2536	73	33.7	0.28 (3)	0.096 (4)	0.292 (2)	0.366 (1)
sick-euthyroid	3163	43	9.8	0.638 (3)	0.224 (4)	0.639 (2)	0.725 (1)
statlog-project-landsat-satellite-5	4434	37	8.4	0.384 (3)	0.287 (4)	0.385 (2)	0.531 (1)
wine-quality	4898	12	25.8	0.189 (3)	0.092 (4)	0.191 (2)	0.224 (1)
optical-digits	5620	65	9.1	0.778 (2)	0.343 (4)	0.776 (3)	0.83 (1)
letter-img	20000	17	26.2	0.561 (3)	0.1 (4)	0.563 (2)	0.793 (1)
chess-king-rook-vs-king-14	28055	35	5.2	0.509 (2)	0.37 (4)	0.509 (3)	0.510 (1)
shuttle	58000	10	3.7	0.935 (2)	0.632 (4)	0.934 (3)	0.944 (1)
skin-segmentation	245056	4	3.8	0.849 (3)	0.808 (4)	0.849 (2)	0.859 (1)
Average F1score				0.571	0.337	0.577	0.63
Average rank				2.775	3.6	2.275	1.35

Table 1: F1-score and rank for different type of logistic regression on each of the 20 datasets.

UCI Dataset Name	n	p	IR	Weighted	Undersample	Oversample	Robust
hepatitis	79	20	5.1	0.645 (2)	0.421 (4)	0.621 (3)	0.671 (1)
fertility	99	13	7.2	0.24 (3)	0.309 (1)	0.247 (2)	0.117 (4)
flags-0	193	60	3.8	0.664 (3)	0.487 (4)	0.673 (2)	0.722 (1)
image-segmentation-path	209	20	6	0.923 (3)	0.513 (4)	0.94 (1)	0.935 (2)
dermatology-5	357	35	6.4	0.979 (3)	0.529 (4)	0.99 (2)	1 (1)
libras-move	360	91	14	0.873 (1)	0.342 (4)	0.854 (2)	0.638 (3)
thoracic-surgery	469	25	5.7	0.249 (3)	0.25 (2)	0.232 (4)	0.338 (1)
climate-model-simulation-crashes	539	19	11	0.548 (2)	0.32 (4)	0.52 (3)	0.696 (1)
connectionist-bench-10	989	11	10	0.197 (2)	0.203 (1)	0.186 (3)	0.159 (4)
yeast-me-2	1484	9	28.1	0.213 (3)	0.143 (4)	0.22 (2)	0.304 (1)
car-eval-34	1728	22	11.9	0.901 (2)	0.288 (4)	0.906 (1)	0.899 (3)
ozone-level	2536	73	33.7	0.276 (2)	0.133 (4)	0.274 (3)	0.387 (1)
sick-euthyroid	3163	43	9.8	0.63 (3)	0.263 (4)	0.632 (2)	0.751 (1)
statlog-project-landsat-satellite-5	4434	37	8.4	0.386 (3)	0.442 (2)	0.377 (4)	0.464 (1)
wine-quality	4898	12	25.8	0.195 (3)	0.096 (4)	0.197 (2)	0.22 (1)
optical-digits	5620	65	9.1	0.779 (2)	0.396 (4)	0.769 (3)	0.833 (1)
letter-img	20000	17	26.2	0.558 (3)	0.249 (4)	0.56 (2)	0.794 (1)
chess-king-rook-vs-king-14	28055	35	5.2	0.504 (3)	0.442 (4)	0.506 (2)	0.517 (1)
shuttle	58000	10	3.7	0.924 (3)	0.843 (4)	0.925 (2)	0.956 (1)
skin-segmentation	245056	4	3.8	0.865 (1)	0.863 (4)	0.865 (2)	0.865 (3)
Average F1score				0.577	0.377	0.575	0.612
Average rank				2.5	3.5	2.35	1.65

Table 2: F1-score and rank for different type of SVM on each of the 20 datasets.

UCI Dataset Name	n	p	IR	Weighted	Undersample	Oversample	Robust
hepatitis	79	20	5.1	0.463 (3)	0.459 (4)	0.578 (1)	0.526 (2)
fertility	99	13	7.2	0.348 (2)	0.165 (3)	0.067 (4)	0.389 (1)
flags_0	193	60	3.8	0.707 (2)	0.558 (4)	0.567 (3)	0.719 (1)
image-segmentation_path	209	20	6	0.867 (3)	0.519 (4)	0.935 (2)	0.95 (1)
dermatology-5	357	35	6.4	0.977 (2)	0.97 (4)	0.983 (1)	0.976 (3)
libras_move	360	91	14	0.355 (3)	0.183 (4)	0.586 (1)	0.454 (2)
thoracic-surgery	469	25	5.7	0.202 (3)	0.256 (1)	0.089 (4)	0.222 (2)
climate-model-simulation-crashes	539	19	11	0.474 (1)	0.264 (4)	0.448 (2)	0.421 (3)
connectionist-bench_10	989	11	10	0.622 (2)	0.273 (4)	0.6 (3)	0.639 (1)
yeast_me2	1484	9	28.1	0.188 (3)	0.147 (4)	0.28 (1)	0.21 (2)
car_eval_34	1728	22	11.9	0.861 (3)	0.262 (4)	0.907 (1)	0.875 (2)
ozone_level	2536	73	33.7	0.199 (2)	0.102 (4)	0.151 (3)	0.218 (1)
sick_euthyroid	3163	43	9.8	0.774 (2)	0.263 (4)	0.865 (1)	0.758 (3)
statlog-project-landsat-satellite-5	4434	37	8.4	0.786 (1)	0.277 (4)	0.747 (3)	0.771 (2)
wine_quality	4898	12	25.8	0.193 (3)	0.092 (4)	0.207 (1)	0.204 (2)
optical_digits	5620	65	9.1	0.818 (2)	0.371 (4)	0.818 (1)	0.778 (3)
letter_img	20000	17	26.2	0.84 (3)	0.137 (4)	0.859 (1)	0.849 (2)
chess-king-rook-vs-king-14	28055	35	5.2	0.487 (1)	0.361 (3)	0.175 (4)	0.468 (2)
shuttle	58000	10	3.7	1 (1.5)	0.99 (4)	1 (3)	1 (1.5)
skin-segmentation	245056	4	3.8	0.998 (3)	0.515 (4)	0.998 (2)	0.998 (1)
Average F1score				0.608	0.358	0.593	0.621
Average rank				2.275	3.75	2.1	1.875

Table 3: F1-score and rank for different type of Optimal Trees on each of the 20 datasets.

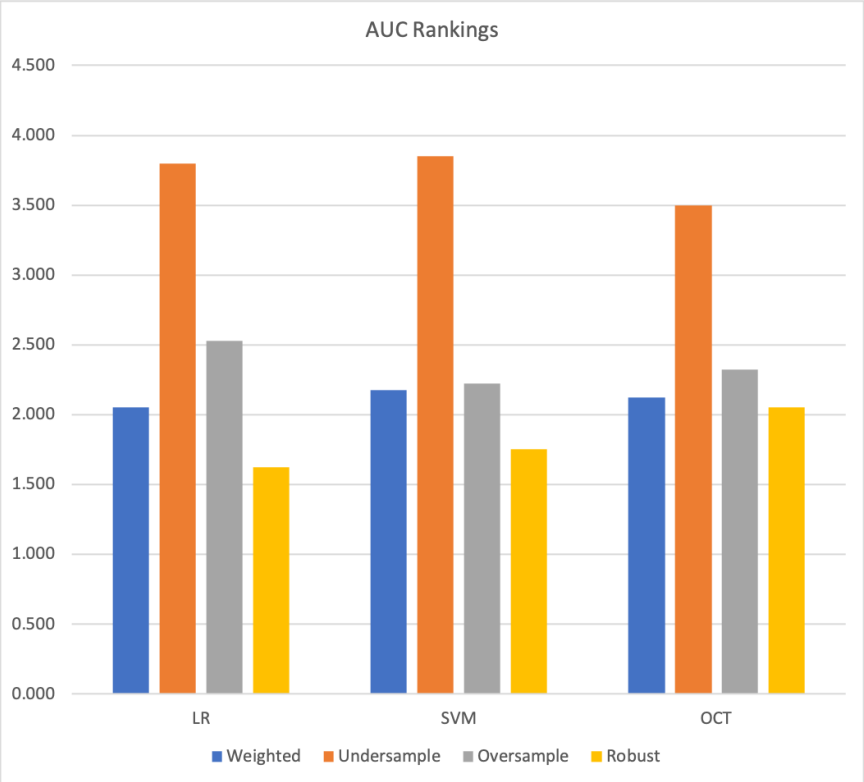

Figure 2: AUC rank for different methods

method GSMOTE. For SVM, the robust method increases out-of-sample AUC by a 0.011 (1.3%) on average compared to weighted SVM and a 0.009 (1.0%) on average compared to oversampling method GSMOTE. For Optimal Trees, the robust method increases the out-of-sample AUC by a 0.005 (0.6%) on average compared to weighted Optimal Trees and a 0.014 (1.7%) on average compared to oversampling method GSMOTE. Figure 2 shows the AUC rank for different methods which illustrates the robust method is the strongest method in terms of average rank, followed by GSMOTE and the weighted method. The exact counts of wins, ties and losses for robust methods compared to other methods are shown in Table 8 and 9.

3.3 Computationally Tractability and Speed

The robust method does not change the nature of the optimization problem complexity. By adding robustness, Logistic Regression changes from unconstrained convex optimization to constrained convex optimization. SVM remains linear optimization.

In order to better understand the scalability of the robust method, we also compared the total time of all methods for logistic regression and SVM in selected UCI datasets. All problems are solved using the MIT Engaging Cluster (Intel Xeon 2.1 GHz) with 1 CPU core and 16GB of memory. The results are shown in Table 7. For logistic regression, the robust method slowed down computation by 1 to 2 orders of magnitude, which caused by solving a

UCI Dataset Name	n	p	IR	Weighted	Undersample	Oversample	Robust
hepatitis	79	20	5.1	0.935 (3)	0.905 (4)	0.957 (1)	0.943 (2)
fertility	99	13	7.2	0.428 (3)	0.407 (4)	0.485 (2)	0.516 (1)
flags-0	193	60	3.8	0.911 (2)	0.887 (4)	0.902 (3)	0.919 (1)
image-segmentation-path	209	20	6	1 (2)	0.988 (4)	1 (2)	1 (2)
dermatology-5	357	35	6.4	1 (2.5)	1 (2.5)	1 (2.5)	1 (2.5)
libras-move	360	91	14	0.965 (1)	0.94 (4)	0.957 (3)	0.964 (2)
thoracic-surgery	469	25	5.7	0.615 (3)	0.625 (2)	0.601 (4)	0.627 (1)
climate-model-simulation-crashes	539	19	11	0.941 (2)	0.915 (4)	0.938 (3)	0.97 (1)
connectionist-bench-10	989	11	10	0.612 (2)	0.559 (4)	0.608 (3)	0.645 (1)
yeast-me-2	1484	9	28.1	0.838 (3)	0.822 (4)	0.842 (2)	0.867 (1)
car-eval-34	1728	22	11.9	0.998 (2)	0.998 (3.5)	0.998 (3.5)	0.999 (1)
ozone-level	2536	73	33.7	0.876 (3)	0.866 (4)	0.887 (2)	0.891 (1)
sick-euthyroid	3163	43	9.8	0.948 (2)	0.917 (4)	0.947 (3)	0.949 (1)
statlog-project-landsat-satellite-5	4434	37	8.4	0.853 (1)	0.822 (4)	0.852 (2)	0.851 (3)
wine-quality	4898	12	25.8	0.785 (1)	0.765 (4)	0.782 (2)	0.782 (3)
optical-digits	5620	65	9.1	0.982 (2)	0.977 (4)	0.981 (3)	0.982 (1)
letter-img	20000	17	26.2	0.989 (1)	0.987 (4)	0.989 (2)	0.989 (3)
chess-king-rook-vs-king-14	28055	35	5.2	0.83 (2)	0.808 (4)	0.83 (3)	0.831 (1)
shuttle	58000	10	3.7	0.992 (1)	0.983 (4)	0.992 (2)	0.991 (3)
skin-segmentation	245056	4	3.8	0.948 (2.5)	0.947 (4)	0.948 (2.5)	0.948 (1)
Average AUC				0.872	0.856	0.875	0.883
Average rank				2.050	3.800	2.525	1.625

Table 4: AUC and rank for different type of logistic regression on each of the 20 datasets.

UCI Dataset Name	n	p	IR	Weighted	Undersample	Oversample	Robust
hepatitis	79	20	5.1	0.913 (3)	0.835 (4)	0.941 (1)	0.937 (2)
fertility	99	13	7.2	0.466 (3)	0.609 (1)	0.497 (2)	0.463 (4)
flags-0	193	60	3.8	0.878 (3)	0.828 (4)	0.894 (2)	0.912 (1)
image-segmentation-path	209	20	6	1 (2)	0.958 (4)	1 (2)	1 (2)
dermatology-5	357	35	6.4	1 (2)	0.966 (4)	1 (2)	1 (2)
libras-move	360	91	14	0.947 (1)	0.894 (4)	0.946 (2)	0.938 (3)
thoracic-surgery	469	25	5.7	0.616 (2.5)	0.6 (4)	0.606 (2.5)	0.692 (1)
climate-model-simulation-crashes	539	19	11	0.938 (3)	0.89 (4)	0.943 (2)	0.970 (1)
connectionist-bench-10	989	11	10	0.615 (2)	0.568 (4)	0.603 (3)	0.644 (1)
yeast-me-2	1484	9	28.1	0.841 (3)	0.756 (4)	0.843 (2)	0.869 (1)
car-eval-34	1728	22	11.9	0.998 (3)	0.995 (4)	0.998 (2)	0.998 (1)
ozone-level	2536	73	33.7	0.87 (3)	0.861 (4)	0.871 (2)	0.891 (1)
sick-euthyroid	3163	43	9.8	0.95 (2)	0.914 (4)	0.949 (3)	0.951 (1)
statlog-project-landsat-satellite-5	4434	37	8.4	0.853 (2)	0.835 (4)	0.85 (3)	0.86 (1)
wine-quality	4898	12	25.8	0.788 (1)	0.707 (4)	0.78 (3)	0.784 (2)
optical-digits	5620	65	9.1	0.981 (1.5)	0.967 (4)	0.981 (1.5)	0.98 (3)
letter-img	20000	17	26.2	0.989 (1)	0.984 (4)	0.989 (2)	0.987 (3)
chess-king-rook-vs-king-14	28055	35	5.2	0.831 (2)	0.813 (4)	0.83 (3)	0.831 (1)
shuttle	58000	10	3.7	0.991 (1.5)	0.988 (4)	0.991 (1.5)	0.991 (3)
skin-segmentation	245056	4	3.8	0.947 (2.5)	0.947 (4)	0.947 (2.5)	0.947 (1)
Average AUC				0.871	0.846	0.873	0.882
Average rank				2.175	3.85	2.225	1.75

Table 5: AUC score and rank for different type of SVM on each of the 20 datasets.

UCI Dataset Name	n	p	IR	Weighted	Undersample	Oversample	Robust
hepatitis	79	20	5.1	0.668 (4)	0.674 (3)	0.765 (1)	0.707 (2)
fertility	99	13	7.2	0.652 (2)	0.418 (4)	0.474 (3)	0.678 (1)
flags_0	193	60	3.8	0.883 (1)	0.798 (3)	0.728 (4)	0.841 (2)
image-segmentation_path	209	20	6	0.944 (3)	0.823 (4)	0.989 (1)	0.978 (2)
dermatology-5	357	35	6.4	0.997 (1)	0.995 (3)	0.997 (2)	0.99 (4)
libras_move	360	91	14	0.658 (4)	0.664 (3)	0.903 (1)	0.791 (2)
thoracic-surgery	469	25	5.7	0.494 (3)	0.571 (1)	0.472 (4)	0.514 (2)
climate-model-simulation-crashes	539	19	11	0.849 (1)	0.761 (4)	0.791 (3)	0.794 (2)
connectionist-bench_10	989	11	10	0.893 (3)	0.695 (4)	0.923 (1)	0.901 (2)
yeast_me2	1484	9	28.1	0.834 (1)	0.819 (2)	0.744 (4)	0.774 (3)
car_eval_34	1728	22	11.9	0.991 (2)	0.793 (4)	0.994 (1)	0.985 (3)
ozone_level	2536	73	33.7	0.755 (1)	0.684 (3)	0.652 (4)	0.733 (2)
sick_euthyroid	3163	43	9.8	0.961 (1)	0.675 (4)	0.953 (3)	0.954 (2)
statlog-project-landsat-satellite-5	4434	37	8.4	0.937 (2)	0.675 (4)	0.946 (1)	0.923 (3)
wine_quality	4898	12	25.8	0.713 (3)	0.656 (4)	0.758 (2)	0.762 (1)
optical_digits	5620	65	9.1	0.926 (2)	0.792 (4)	0.916 (3)	0.927 (1)
letter_img	20000	17	26.2	0.978 (3)	0.76 (4)	0.981 (2)	0.983 (1)
chess-king-rook-vs-king-14	28055	35	5.2	0.831 (1)	0.648 (4)	0.8 (3)	0.818 (2)
shuttle	58000	10	3.7	1 (1)	0.997 (4)	1 (3)	1 (2)
skin-segmentation	245056	4	3.8	0.999 (3)	0.753 (4)	0.999 (1.5)	0.999 (1.5)
Average AUC				0.848	0.732	0.839	0.853
Average rank				2.1	3.5	2.375	2.025

Table 6: AUC score and rank for different type of Optimal Trees on each of the 20 datasets.

		UCI dataset (number of points; dimension; imbalance ratio)					
		fertility	connectionist-bench-10	yeast-me-2	sick-euthyroid	letter-img	shuttle
Model	Method	(99;13;7)	(989;11;10)	(1484;9;28)	(3163;43;10)	(20000;17;26)	(58000;10;4)
Logistic regression	Weighted	0.01	0.01	0.01	0.03	0.07	0.24
	Undersample	0.03	0.04	0.05	0.16	0.3	1.05
	Oversample	1.24	5.03	8.03	22.69	111.77	391.22
	Robust	9.69	18.36	17.73	259.47	374.54	1035.09
Support vector machines	Weighted	0	0	0.01	0.03	0.05	0.22
	Undersample	0.01	0.12	0.16	2.06	11.67	51.83
	Oversample	0.65	4.77	7.64	21.96	106.36	370.51
	Robust	0.27	4.38	2.85	20.25	180.11	453.51

Table 7: Solving time for selected UCI datasets in seconds

constrained convex optimization problem. For SVM, the robust method takes longer than the weighted and undersampling method but close to the oversampling method. The results here show the robust method can be solved in an acceptable time for all datasets.

4. Conclusions

In this paper, we proposed a robust optimization framework of training logistic regression and SVM on imbalanced datasets by balancing majority class and minority class data. Then we show the balanced dataset we generated can also be utilized to improve the performance of other methods like Optimal Trees. We applied this algorithm to 20 imbalanced datasets collected from the UCI machine learning repository. The proposed algorithm ranks first on average compared to the state-of-the-art oversampling and undersampling method. In the

Model	Methods	Wins	Losses	Ties
LR	Robust vs Weighted	19	1	0
	Robust vs Undersample	17	3	0
	Robust vs Oversample	17	3	0
SVM	Robust vs Weighted	15	5	0
	Robust vs Undersample	18	2	0
	Robust vs Oversample	14	6	0
Optimal Trees	Robust vs Weighted	13	6	1
	Robust vs Undersample	19	1	0
	Robust vs Oversample	10	10	0

Table 8: Pairwise comparisons of robust methods with other methods on F1-score

Model	Methods	Wins	Losses	Ties
LR	Robust vs Weighted	13	5	2
	Robust vs Undersample	19	0	1
	Robust vs Oversample	13	5	2
SVM	Robust vs Weighted	12	6	2
	Robust vs Undersample	19	1	0
	Robust vs Oversample	12	6	2
Optimal Trees	Robust vs Weighted	10	10	0
	Robust vs Undersample	17	3	0
	Robust vs Oversample	12	7	1

Table 9: Pairwise comparisons of robust methods with other methods on AUC

end, we illustrate the proposed algorithm can be solved in an acceptable time for sizeable datasets.

References

Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, 6(1):20–29, 2004.

Dimitri P Bertsekas. Nonlinear programming. athena scientific. *Belmont, MA*, 1999.

Dimitris Bertsimas, Jack Dunn, Colin Pawlowski, and Ying Daisy Zhuo. Robust classification. *INFORMS Journal on Optimization*, 1(1):2–34, 2019a.

Dimitris Bertsimas, Jack Dunn, Dale W Steele, Thomas A Trikalinos, and Yuchen Wang. Comparison of machine learning optimal classification trees with the pediatric emergency care applied research network head trauma decision rules. *JAMA pediatrics*, 2019b.

Andrew P Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern recognition*, 30(7):1145–1159, 1997.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16: 321–357, 2002.

Nitesh V Chawla, Aleksandar Lazarevic, Lawrence O Hall, and Kevin W Bowyer. Smoteboost: Improving prediction of the minority class in boosting. In *European conference on principles of data mining and knowledge discovery*, pages 107–119. Springer, 2003.

Nancy Chinchor. The statistical significance of the muc-4 results. In *Proceedings of the 4th conference on Message understanding*, pages 30–50. Association for Computational Linguistics, 1992.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3): 273–297, 1995.

Georgios Douzas and Fernando Bacao. Geometric SMOTE a geometrically enhanced drop-in replacement for SMOTE. *Information Sciences*, 501:118–135, October 2019a. doi: 10.1016/j.ins.2019.06.007. URL `https://doi.org/10.1016/j.ins.2019.06.007`.

Georgios Douzas and Fernando Bacao. Geometric smote a geometrically enhanced drop-in replacement for smote. *Information Sciences*, 501:118–135, 2019b.

Dheeru Dua and Casey Graff. Uci machine learning repository (2017). *URL http://archive. ics. uci. edu/ml*, 37, 2017.

Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.

Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, pages 1322–1328. IEEE, 2008.

Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

LLC Interpretable AI. Interpretable AI documentation, 2019. URL `https://www.interpretable.ai`.

Miroslav Kubat, Stan Matwin, et al. Addressing the curse of imbalanced training sets: one-sided selection. In *Icml*, volume 97, pages 179–186. Nashville, USA, 1997.

Jorma Laurikkala. Improving identification of difficult small classes by balancing class distribution. In *Conference on Artificial Intelligence in Medicine in Europe*, pages 63–66. Springer, 2001.

Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17):1–5, 2017. URL `http://jmlr.org/papers/v18/16-365.html`.

Miles Lubin and Iain Dunning. Computing in operations research using julia. *INFORMS Journal on Computing*, 27(2):238–248, 2015.

Raghav Pant, Theodore B Trafalis, and Kash Barker. Support vector machine classification of uncertain and imbalanced data using robust optimization. In *Proceedings of the 15th WSEAS international conference on computers*, pages 369–374. World Scientific and Engineering Academy and Society (WSEAS) Stevens Point ..., 2011.

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

Foster Provost and Tom Fawcett. Robust classification for imprecise environments. *Machine learning*, 42(3):203–231, 2001.

Michael R Smith, Tony Martinez, and Christophe Giraud-Carrier. An instance level analysis of data complexity. *Machine learning*, 95(2):225–256, 2014.

Yanmin Sun, Andrew KC Wong, and Mohamed S Kamel. Classification of imbalanced data: A review. *International Journal of Pattern Recognition and Artificial Intelligence*, 23(04): 687–719, 2009.

Andreas Wächter and Lorenz T Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1):25–57, 2006.

Qiang Yang and Xindong Wu. 10 challenging problems in data mining research. *International Journal of Information Technology & Decision Making*, 5(04):597–604, 2006.