UNIVERSITY OF CAPE TOWN
IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

Department of Computer Science

Computer Science Honours
Final Paper
2016

Title: An Analysis of Classification Techniques for the Prediction of Tuberculosis Defaulters and Community Health Worker Attrition

Author: Brian Mc George

Project abbreviation: ML4CHW

Supervisor: Dr Brian DeRenzi

Category	**Min**	**Max**	**Chosen**
Requirement Analysis and Design	0	20	
Theoretical Analysis	0	25	
Experiment Design and Execution	0	20	17
System Development and Implementation	0	15	10
Results, Findings and Conclusion	10	20	18
Aim Formulation and Background Work	10	15	15
Quality of Paper Writing and Presentation	10		10
Quality of Deliverables	10		10
Overall General Project Evaluation	0	10	
Total marks			80

An Analysis of Classification Techniques for the Prediction of Tuberculosis Defaulters and Community Health Worker Attrition

Brian Mc George
University of Cape Town
Cape Town, South Africa
mcgbri004@myuct.ac.za

ABSTRACT

As the use of electronic data capturing for community health projects becomes more widespread the availability of this type of data for research use is becoming more prevalent. In this paper we outline various issues that may arise when attempting to apply classification techniques to predict minority class events and apply several strategies to counteract them. We focus on the prediction of Tuberculosis defaulters and the attrition of community health workers. We compare 13 classification techniques, 13 data balancing techniques and 6 feature selection techniques to assess what combination produces the best classification results. We find that the use of a data balancing techniques greatly improves balanced accuracy on imbalanced datasets. We recommend the use of Logistic regression, Artificial neural networks, Random Forest and Bernoulli Naive Bayes for different use-cases. Adaptive synthetic sampling is recommended as an over-sampler and edited nearest neighbours as an under-sampler. Random forest is recommended for identifying the most relevant features as well as being able to remove noisy features that do not add to the classification.

1. INTRODUCTION

Classification techniques can be used to flag individuals who have a high probability of certain events occurring. In this paper we outline various issues that may arise when attempting to apply classification techniques to predict minority class events and apply several strategies to counteract them. We focus on the prediction of Tuberculosis (TB) defaulters and the attrition of community health workers (CHW).

A patient is considered to have defaulted if their treatment is interrupted for longer than a set duration, typically two months for TB treatment. In 2013 over 210 000 patients defaulted from TB treatment worldwide [65]. The rate of default in the Americas is the highest at 8% with Africa at 5% [65]. The consequences of defaulting TB treatment include: increased drug resistance, increased health system costs [37, 49], higher risk of mortality, continued risk of transmitting the disease to others [37] and increased rate of recurrent disease [32]. The spread of TB can be reduced if the individuals who have a high risk of defaulting can be predicted. This will also reduce health system costs.

Attrition is the loss of workers through resignation or abandonment of the work. If we can flag workers who are at likely to quit quickly enough then various intervention measures could be implemented in order to retain those workers. This could reduce costs because new workers would not have to be trained.

The aim of this paper is to inform the reader on issues that can occur when building classification models from imbalanced datasets. Bias can occur in the models since most classifiers expect equal weighting of samples for each class. This can lead to poor classification for the minority class, which is often the class of interest. The second aim of this paper is to propose recommendations on classification techniques, data balancing techniques and strategies to improve the overall classification for this problem domain. We examine a number of different data balancing techniques which either over-sample the minority class, under-sample the majority class or provide a combination of an over-sampling and under-sampling technique. We provide a large scale comparison of different classification techniques in order to determine which technique is best suited to these types of problems. As part of our evaluation of each classification technique, we examine how well each classifier works out-of-the-box compared when they are tuned by searching a grid of parameters. Furthermore, we apply a number of feature selection strategies to determine which strategy is best suited to remove redundant and noisy variables.

To facilitate the aforementioned experiments, we developed a testing system which allows new classification techniques and data sets to be supported quickly and easily. The testing system is designed to allow near-exact reproducibility of results.

In addition to our TB default dataset and attrition dataset, we include two real-world credit scoring datasets. The field of credit scoring in the financial space aims to determine if a financial institution should provide credit to an individual. This is a well researched binary classification problem. We included these as all the problems try to flag high-risk individuals and it will allow us to get a better understanding of how the results compare and generalise.

From our results we recommend the use of Logistic re-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2016 ACM. ISBN n/a.
DOI: n/a

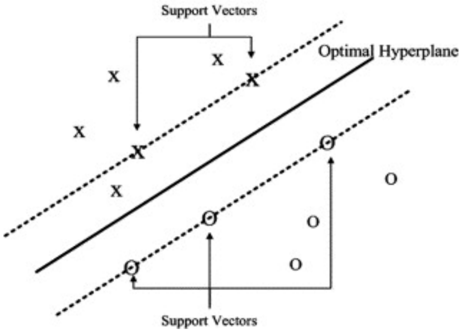

Figure 1: An overview of an SVM [43]

gression (LR), Artificial neural networks (ANN), Random Forest (RF) and Bernoulli Naive Bayes (NB) for different use-cases. Adaptive synthetic sampling is recommended as an over-sampler and edited nearest neighbours as an under-sampler. Random forest is recommended for identifying the most relevant features as well as being able to remove noisy features that do not add to the classification.

2. BACKGROUND

This section aims to provide an overview of all the techniques and metrics used in this paper.

2.1 Definition of a defaulter

The definition of a defaulter depends on its context. TB literature typically uses the World Health Organisation (WHO) definition that a defaulter is a person whose treatment has been disrupted for two or more consecutive months [10, 14, 32, 33, 49, 65].

2.2 Classification techniques

We selected a variety of classification techniques to benchmark. We include a selection of well known techniques, ensemble techniques and newer techniques that have shown promising results in other studies. Our choices are based partly on our review of classification techniques used in the credit scoring field in Section 3.2.

2.2.1 Support Vector Machines

Support Vector Machines (SVM) is a machine learning technique that can be used to produce regression or classification functions from a set of training data [45]. SVM works by mapping the input vectors into a high-dimensional feature space with the use of a kernel function [15]. The kernel function selected determines the if the mapping is linear or non-linear [45]. Linear, polynomial and radial basis function (RBF) are common kernel functions [27]. The polynomial and RBF kernel performs a non-linear mapping into the high-dimensional space [27]. This feature space is then searched to acquire an optimal hyperplane that separates the space with the maximum distance between the two classes [15]. Figure 1 shows an example of this. Hsu *et al.* [27] recommends the RBF kernel as a reasonable first choice but notes that it is not suitable when there are a large number of features. The linear kernel is recommend when there are a large number of features.

2.2.2 Artificial Neural Network

ANN is based on the functionality of the human brain [64]. Neurons in the brain are interconnected and process information in parallel [64]. A typical ANN is comprised of an input layer, k number of hidden layers and an output layer. The neurons in each layer are connected to the neurons in the next layer. A numeric weight is defined between each pair of connected neurons. An activation function defines if a neuron will fire [64]. The activation function bounds the value of a neuron to a specific range to limit the effect of divergent neurons [64]. By using an activation function, a non-linear combination of weights can be generated [64]. It has been proven that an ANN is able to approximate any continuous function if has at least one hidden node and an activation function that is both bounded to some range and non-constant [25].

ELM is an alternative approach to the conventional back-propagated ANN. Huang *el al.* [30] proved that the input and hidden layer weights can be randomly assigned if the activation function in the hidden layer is infinitely differentiable. By randomly assigning these weights, the weights for the output nodes can be determined analytically [30]. This allows ELMs to be trained orders of magnitude faster than a back-propagated ANN [29, 30]. ELMs have been shown to provide better results than SVMs and ANNs on a variety of classification and regression tasks [29, 30].

2.2.3 Logistic Regression

LR is a technique that models the chance of an outcome based on the input features [55]. Since chance is a ratio, the logarithm of the chance is modelled instead [55]: $\log(\frac{p}{1-p}) = \beta_0 + \beta_1 x_1 + ... + \beta_m x_m$. p represents the probability of an event (likelihood to default for example). β_0 represents the value of the criterion when the predictor is equal to 0. $\beta_1, ..., \beta_m$ are the regression coefficients associated with the $x_1, ..., x_m$ input features. The probability of an event can then be calculated as $p = \frac{1}{1+e^{-(\beta_0+\beta_1 x_1+\cdots+\beta_m x_m)}}$. A detailed overview of logistic regression can be found in [48] and [55].

2.2.4 k-Nearest Neighbours

The k-Nearest Neighbours (kNN) algorithm determines the output classification by examining the k nearest training examples in the feature space [35]. An input is classified by the majority vote of its neighbours.

2.2.5 Ensembles

An ensemble classifier is one which typically makes use of the aggregation of multiple classifiers. A single decision tree (DT) can be used for classification by branching on conjunctions of features and having the leaves represent the output class. A decision tree allows for easy interpretation of the generated model, however, typically provides relatively poor classification accuracy [59].

RF is a technique that fits a number of decision trees on random samples with replacement of the dataset. For each tree, n features are randomly selected and the tree is grown [9]. Samples that were not selected for training the tree is called out-of-bag data [9]. It is used to test the error rate of the forest [9, 59]. The output classification is decided by the majority vote of each tree [9].

Freund and Schapire's [19] AdaBoost uses a weighted sum of multiple classifiers in order to determine the output. A classifier is constructed in an iterative fashion. At each iteration a pool of classifiers are considered and one classifier is added to the committee of classifiers. Input which is still misclassified is assigned a higher weight at each iteration [7, 54]. The aim is that the process will select a classifier which helps the still misclassified inputs. The chosen classifier is assigned a weight which determines its power in the overall classification [7, 54]. The sign of the function: $C(x) = \sum_{i=1}^{m} \alpha_i k_i$ is used to determine the final classification with α_i denoting the weight of each classifier k_i [7].

2.2.6 *Naive Bayes*

NB makes an assumption that each input feature is independent of each other [42, 53]. This allows multiple features be uncoupled from one another which simplifies the algorithm. Naive Bayes uses conditional probability to classify new inputs [42]. Using Bayes theorem, the following equation is derived: $p(C_i|\mathbf{x}) = \frac{p(C_i) \times p(x_1|C_i) \times \ldots \times p(x_n|C_i)}{p(\mathbf{x})}$ with input $\mathbf{x} = (x_1, ..., x_n)$ and class label C_i [42, 53]. Since $p(x)$ is identical for each class, it is typically ignored [53]. To classify an input, the probabilities for each class are calculated using the aforementioned equation and the output is the class with the highest probability [42].

The event model of Naive Bayes classification describes the assumed distribution of features. The Gaussian event model assumes that $p(x_i|C_i)$ follows a Gaussian distribution [34]. This allows support of continuous x_i values [34]. The multivariate Bernoulli event model assumes that features are independent boolean values [47].

2.2.7 *Clustering-launched classification*

CLC is a binary classifier. CLC first clusters the data into groups using a diverse hierarchical k-means algorithm [45]. The clusters are divided into positive subsets and negative subsets [45]. Support vectors are then used to separate the positive and negative subsets for each cluster [45]. Redundant or repeated support vectors are removed thereafter [45].

2.3 Data Balancing algorithms

This section examines the different data balancing approaches and their respective algorithms. Our problem space often has an imbalanced number of examples for each class. Most classifiers expect an equal number of samples per class else will bias heavily towards the majority class. we include an analysis of several data balancing algorithms to ensure our classifiers are able to detect both the positive and negative class as best as possible. We chose our balancing techniques to facilitate an evaluation of both over-sampling and under-sampling techniques. Included in the evaluation is techniques that use simple approaches and as others that use sophisticated algorithms.

2.3.1 *Over-sampling*

Random over-sampling (ROS) is a technique that randomly replicates examples in the minority class [5]. However, this technique can increase the likelihood of over-fitting since exact copies are made from the minority class [5].

The Synthetic minority over-sampling technique (SMOTE) [11] forms new minority samples by interpolating along the line segment on some or all of the k nearest minority class neighbours of a minority example. This approach attempts to alleviate the over-fitting that can occur from using random over-sampling.

Adaptive synthetic sampling (ADASYN) [24] is a variation of SMOTE which uses a weighed distribution for different minority class examples according to their difficulty in learning. More synthetic data is generated for minority class examples that are more difficult to learn compared to those that are easier to learn.

2.3.2 *Under-sampling*

Random under-sampling (RUS) is a technique that randomly eliminates examples in the majority class [5]. However, this technique can remove potentially useful information from the training set [5].

Condensed Nearest Neighbour (CNN) rule [21] finds a consistent subset of examples. A subset D of E is consistent if a 1-nearest neighbour classifier trained with D correctly classifies E [5]. The process draws one random majority class example and all minority class examples and puts them in D [5]. Every misclassified example is then added from E to D [5]. This process attempts to remove examples that are far away from the decision border and therefore seen as less relevant for learning [5].

The Tomek link (TL) algorithm [60] examines two examples $\mathbf{x}_i$ and $\mathbf{x}_j$ belonging to different classes. A TL occurs if there is not an example $\mathbf{x}_k$ such that $d(\mathbf{x}_i, \mathbf{x}_k) < d(\mathbf{x}_i, \mathbf{x}_j)$ or $d(\mathbf{x}_j, \mathbf{x}_k) < d(\mathbf{x}_j, \mathbf{x}_i)$. If two examples form a TL then either one is noise or it is a borderline case [5]. This information can then be used to under-sample the majority class [5].

One-sided selection (OSS) [36] applies TL to remove borderline and noisy majority class examples then applies CNN to remove majority examples far from the decision border.

Neighbourhood cleaning rule (NCL) [38] uses the edited nearest neighbour rule (ENN) to remove majority class examples. ENN removes examples whose label differs from the class of at least two of its nearest 3 neighbours. NCL examines each example $\mathbf{x}_i$ and its 3 nearest neighbours. If $\mathbf{x}_i$ belongs to the majority class and the neighbours contradict this class then $\mathbf{x}_i$ is removed [5]. If $\mathbf{x}_i$ belongs to the minority class and the neighbours contradict this class then the neighbours from the majority class are removed [5].

Instance threshold hardening (ITH) [57] uses the probability estimates from a classifier (such as SVM with a linear kernel) when k-fold cross validation is applied. It selects the m examples from the majority class that have the highest probability estimates for that class when tested in k-fold cross validation. m is the number of samples in the minority class.

NearMiss-1 (NM-1) [66] picks majority class examples which have the smallest average distance to the three nearest minority class examples.

Cluster centroids (CC) applies the k-means algorithm with m clusters to the majority class and uses the coordinates of the cluster centroids as the majority samples. As in ITH, m is the number of samples in the minority class.

2.3.3 *Combination*

SMOTE + TL [4] first applies SMOTE then applies TL. Applying SMOTE can cause minority samples to extend too deep into the majority class space or the opposite where majority class samples extend too deep into the minority class space [4]. TL is used as a data cleaning method to remove examples from both classes to produce well-defined class clusters [4].

SMOTE + ENN works similarly to SMOTE + TL but

ENN is more aggressive at removing samples than TL [5].

2.4 Evaluation metrics

The number of true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN) are used to define several metrics which are used to compare the results in this paper. The true positive rate (TPR) or sensitivity defines the proportion of actual positives which are predicted as positive. The TPR is calculated as $\frac{TP}{TP+FN}$. The true negative rate (TNR) or specificity defines the proportion of actual negatives which are predicted as negative. The TNR is calculated as $\frac{TN}{TN+FP}$.

Accuracy is typically a poor measure of quality for imbalanced datasets as classifiers tend to bias towards the majority class [5, 12]. Several balanced performance metrics are used instead. The balanced accuracy (BACC) provides an equal weighting in TPR and TNR. It is calculated as $\frac{TPR+TNR}{2}$. Matthews correlation coefficient (MCC) also provides a balanced measure of classification quality and is scored between -1 and 1. The MCC is calculated as: $\frac{(TP\times TN)-(FP\times FN)}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$. The Brier score (BS) measure the accuracy of probabilistic predictions in a range of 0 to 1 where a lower score is better [58]. The BS is calculated as: $\frac{1}{N}\sum_{t=1}^{N}(f_t - o_t)^2$ where N is the number of predictions made, f_t is the probability of the positive class being true and o_t is the actual outcome. The receiver operating curve (ROC) compares TPR for a given FPR. Area under ROC (AUC) is an aggregate measure which averages performance of the classifier over all threshold values. However, recent results suggest that AUC does not treat the relative cost of misclassification the same for each classifier [23]. H-measure is an alternative to AUC that remedies this issue through use of a β-distribution which specifies the relative severity of misclassification in a consistent manner for each classifier [23]. H-measure ranges from 0 for a random classifier to 1 for a perfect classifier [23].

3. RELATED WORK

There have been relatively few studies that apply classification techniques to directly predict defaulters of treatment and CHW attrition. The work that has been done for TB default focuses on determining the individual features associated with treatment default. To get an understanding of the work we want to do, we evaluated the credit scoring field as it is a similar binary classification problem and has been well researched.

3.1 Determining predictors of TB default

There have been many studies which focus on determining the factors associated with TB default. We evaluated a selection of these publications [10, 32, 33, 37, 49, 56]. The majority of techniques use a form of logistic regression to determine the association.

The datasets used by the publications contain different features. Age and gender are common throughout the datasets. History of past default is available for all datasets except for Shargie *et al.* [56]. Lackey *et al.* [37] only picked individuals who did not have a history of past default. Jittimanee [33] was the only publication with the feature that did not find it to be significant to the 95% confidence level. However, it did have an odds ratio of 2.19 and a p-value of 0.12. It can therefore be deduced that a history of past default has a strong correlation to default. Two out of three publications with the alcohol abuse feature available, found it to be significant. Three of the four publications with side effects as a feature, found it was significant. Shargie *et al.* [56] and Jittimanee *et al.* [33] measured distance and time to treatment site respectively. It can be reasoned that the aforementioned features will be significant in other datasets since they were found to be significant in the majority of the publications. Other significant features such as illegal drug use, use of herbal medication, daily jobs, history of lung cancer and history of liver disease only appeared once in the datasets. It cannot be discerned if the significance is generalisable or specific to the dataset. The identification of the same features as significant is fairly consistent for the publications that have those features in their dataset.

3.2 Credit scoring

The credit scoring field aims to predict high risk individuals who should not be provided credit. This field has been well researched in the last 20 years. We reviewed a selection of these papers to gain insight on classifier performance as well examine their experimental methodology. Lessmanna *et al.* [41] conducted a large scale review of credit scoring papers in the last 10 years. Lessmanna *et al.* [41] also conducted an independent evaluation of traditional and novel classification techniques on 5 different credit scoring techniques. Lessmanna *et al.* [41] found that ensemble techniques performed better than individual classifiers with random forest achieving the best results averaged over all the datasets. ANN, logistic regression and SVM with RBF kernel provided the best results for individual classifiers with ANN beating logistic regression and SVM [41]. This result contradicts several earlier studies which found SVMs to outperform ANNs [15, 28, 31, 43].

Several other papers have also found ensamble techniques to be better than a single classifier [26, 50, 62, 63]. Tsai *et al.* [61] was the only paper that we reviewed that found that a stand-alone ANN performed better than an ensemble of classifiers.

Of the credit scoring papers we reviewed [1, 3, 6, 15, 17, 26, 31, 28, 39, 43, 45, 46, 50, 61, 62, 63] all provide measures of the correctness of the classifier such as accuracy, true positive rate and false positive rate. However, only a few [17, 31, 46, 63] conduct statistical tests to determine if there is a significant difference between results. Lessmanna *et al.* also recommends using the Brier test to measure the accuracy of the probability estimates produced by the classifiers. None of the evaluated papers in Lessmanna *et al.*'s review or our own review used such a test.

4. EXPERIMENTAL DESIGN AND EVALUATION

A systematic experimental design was developed to ensure repeatable results.

4.1 Experimental Approach

We divide our investigation into 5 complementary experiments. The first experiment focus on determining the importance of data balancing and parameter tuning for each classifier. The second experiment transitions into benchmarking each classifier and producing recommendations on which are suitable choices. The third experiment focuses on determining optimal data balancing schemes. The forth experiment uses feature selection to remove redundant features and to determine the most important features. The firth experiment determines how the time to default affects patient classification.

All our experiments use stratified $n\times 5$-fold validation to divide a single dataset into multiple training and testing datasets. Stratified k-fold divides the dataset into k segments and ensures that each segment contains the same ratio of positive and negative examples as the dataset as a whole. k training and testing sets are created by using one segment as the testing dataset and every other segments as the training data. This is repeated such that every segment is used as testing data. We then repeat the stratified 5-fold validation n times to obtain robustness in results. The results for each fold and n runs are then averaged before being presented. 5-fold was used instead of 10-fold to ensure that there was a reasonable sample of defaulters per fold.

All data is first pre-processed before it is used for each experiment. The median, mean or most frequent value are common approaches to fill in missing values for samples. Another approach is to remove samples that contain missing values. We opted to remove samples with missing data to prevent it causing bias in our classification. Most classification techniques are not equipped to handle categorical data. To address this issue, we use "one-hot encoding" to encode a feature with n categories into n separate binary features corresponding to each category. The feature that corresponds with the categorical value is set to 1 while the other $n-1$ features are set to 0. We standardise all numeric features to ensure that it has zero mean and unit variance since most classifiers expect features to have a normal distribution. For each fold, the mean and standard deviation are derived from the training set and then used to standardise both the training and testing set.

4.1.1 Parameter tuning

As part of the evaluation of the different classification parameter tuning is for our datasets and its effect on each classifier. We also want to see if just the addition of a data balancing algorithm can improve the classification. We first test each classifier at its default parameters. In the case where a classifier does not have default parameters available, we provide our own reasonable defaults and label those classifiers accordingly. To see the effect of a data balancing algorithm on the classifiers with default parameters, we will use the data balancer that results in the highest BARR. Finally, for each classifier we search a grid of reasonable parameters and select the parameters that yield the highest BARR. The grid includes the parameters of the classifier as well as the different data balancing techniques.

Over-fitting the classifiers is a concern when applying the grid search. To prevent the classifiers from over-fitting the parameter grid is kept fairly coarse and three runs of stratified 5-fold cross-validation is executed and the results averaged. To provide fair comparison between parameter sets and to facilitate repeatability, the data balancer, stratified k-fold algorithm and classifier itself use fixed initialisation values. These initialisation values differ across each of the three runs. This ensures that each parameter set utilises the same training and testing folds per run but that these folds differ across runs. The initialisation values are saved in the experiment results to facilitate reproducibility. We chose BARR to compare the results between classifiers and scenarios.

4.1.2 Comparison of classification techniques

This experiment focuses on comparing each optimised classifier in detail. The same testing procedure is used as outlined in Section 4.1.1 except in this experiment we record several additional metrics to compare the classification techniques in detail: TPR, TNR, BARR, MCC, BS and time to fit each training fold. To determine how the results generalise against the other classification datasets, a final scatter plot is presented which compares difference of TPR and FPR from the median of each for every classifier on every dataset.

The recorded metrics will allow us to determine which classifier is best suited for our treatment default datasets but also allow us to see if the results generalise to the credit default datasets and to reason over why we see those results.

In order to deduce if there is any significant different between results, we apply statistical tests. Since classifier results are typically not normally distributed and since we want to determine if there is significance over multiple datasets. We follow Demšar's [16] recommendation and use the Friedman test with the Nemenyi post-hoc test to compare multiple classifiers over multiple datasets. The Friedman test is a non-parametric equivalent of the repeated-measures ANOVA and has a null-hypothesis that each classifier is equivalent to one another and therefore their average ranks are equal [16]. If the null-hypothesis is rejected then a post-hoc test can be conducted to determine if the performance between two individual classifiers are equal. The Nemenyi test has been criticised as being overly conservative [20]. However, we prefer this approach as it will be more resilient to minor differences caused by noise and stochastic variance. We use 15 runs of 5-fold validation when testing each classifier on each dataset. The large number of runs makes it easier for the Nemenyi test to identify differences between the classifiers.

4.1.3 Comparison of data balancing algorithms

We would like to investigate each data balancing technique against each classifier. We use our parameter grid results from Section 4.1.1 to obtain the optimal parameters for each classifier on every data balancing algorithm. As before, the highest BARR is used to determine these optimal parameters. To ensure repeatable results, 10 runs of stratified 5-fold cross validation is executed and the results averaged. A set of unique randomly selected initialisation keys used for each run and recorded with the results. These are used so that k-fold cross validation, data balancing and classification algorithms produce deterministic output.

We present our results as a scatter plot which compares difference of true positive and false positive from the median of each for each classifier on the TB and attrition datasets. We use an additional scatter plot to show difference of true positive and false positive from the median for each data balancer on each dataset. Each data balancer result is created by averaging the results of each classifier when using the particular data balancing technique.

4.1.4 Comparison of feature selection algorithms

Feature selection can be used to help remove noisy features that do not contribute to the overall classification [22]. It can also speed up training times for large datasets [22]. Aside from the aforementioned benefits, we want to use feature selection to get a better understanding of how the features are being utilised by each classifier. Typically only classifiers that create a model with some linear combination of features can be easily interpreted.

A number of feature selection strategies were selected for this experiment: Analysis of variance (ANOVA) F-value with χ^2 tests, logistic regression, linear SVM, Bernoulli naive Bayes, decision tree, random forest. The feature selection is calculated on the training examples and not the entire dataset to prevent bias in our results [8].

The ANOVA F-test is used on numeric features and χ^2 test on categorical features. The ANOVA F-test tests the null hypothesis that 2 or more groups have the same population mean. The χ^2 test the null hypothesis that features that are independent of its class and therefore not relevant to the classification. We chose a p-value of 0.1 to reject each of the null hypotheses.

We measure our feature selection techniques in two different ways. In the first approach we only retain the 15 most important features. We want to identify the technique that can retain the most important features for our different classifiers and will therefore result in the lowest reduction in BARR. We test this as the papers identified in Section 3.1 typically make use of LR to identify the predictors of default. Our experiment will determine if this is the best approach.

In the second approach we apply recursive feature elimination (RFE) in order to identify the optimal number of parameters for the particular technique. RFE removes the n least important feature at each iteration until only k features remain. Good feature selection techniques should be able to reduce the overall features while retaining or even improving the classification. For each feature selection technique the set of features that result in the highest BARR are retained. For some training folds, this could result in no features being removed.

To test the feature selection strategies we record the BARR of each classifier with each feature selection strategy. For the RFE approach the minimum features, maximum features and average features selected is also recorded since each fold of the k-fold cross-validation could have a different number of features selected.

4.1.5 An analysis of time to default in the Lima Peru dataset

We want to determine if the time it takes for an individual to default affects their classification "profile". We use several default ranges: 0-30 days, 0-60 days, 0-100 days, 0-200 days, 50-100 days, 100-200 days, 200-1000 days, 300-1000 days. The non-defaulters are randomly divided into to two sets. The first set is joined with the defaulters in the range and used as the training set while the other set is joined with the defaulters outside the range and used as the testing set. This process is repeated over 100 runs and the results are then averaged. A set of unique random initialisation keys are generated for each run. This is used to ensure that the split of non-defaulters is the same for each default range and that each data balancer and classifier gets the same initialisation values. If the time to default does not play a large role in the classification then we except to see similar results for each default range. If it does play a large role then we expect to see poor classification results and variability in results between default ranges.

4.2 Datasets

The experiments were run on a set of real-world datasets: one TB default dataset from Lima Peru [37], one CHW attrition dataset from India and two credit scoring datasets. The TB dataset was obtained from the Dryad digital repository. The CHW attrition dataset was obtained from Dimagi. The two credit scoring datasets (later referred to as the German and Australian dataset) were obtained from the UCI machine learning repository [44]. Table 1 provides an overview of the characteristics of the datasets. The Peru TB datasets is highly imbalanced which makes it ideal to test different balancing algorithms. The German dataset and India attrition dataset are slightly imbalanced while the Australian dataset is balanced. The Peru data set came with values pre-discretized and therefore only contains categorical features. The India attrition dataset contains 90 days of CHW evaluation measures as well as the project and sector which the CHW is part of. The classification label is whether the CHW is still present at the end of the second quarter.

4.3 Limitations

While our approach has been designed to limit bias and over-fitting as far as possible, the scale of our comparison and size of our datasets could introduce over-fitting nonetheless. For this reason, our observations and conclusions incorporate the use of trends, prior knowledge and comparison to our other datasets.

5. SOFTWARE DEVELOPMENT

This section outlines the software development methodology and details of the implementation.

5.1 Development principles and methodology

Throughout this project we strived to ensure that our software is highly configurable and modular such that core components can be re-used for each experiment. Our goal was to create a system that could support multiple datasets and classifiers in a generic manner. We wanted the system to handle the entire experimental process including preprocessing and result visualisation.

We followed an iterative development methodology for our software. We used feedback from our weekly meeting to improve our experimental methodology and visualisation of results.

5.2 Implementation details

We developed the system in Python 2.7 since it has widespread library support and facilitates quick development. We support datasets formatted as a comma separated values (CSV) file and allow new binary classification datasets to be supported with the addition of just a few lines in the configuration file. All pre-processing operations such as removal of samples with missing values, creation of dummy variables and scaling of data are executed within the code base.

To support a wide variety of techniques, we used several pre-existing libraries for our classification and data balancing techniques. We used scikit-learn [51] for most of the classification techniques with the exception of ELM and CLC. Akusok *et al.*'s [2] Python ELM toolbox was used for the implementation of the ELM. The CLC implementation was obtained from the original authors [13] as a compiled C++

Table 1: Data set summary

Data set	Entries	Number of numerical features	Number of categorical features	Number of binary features	Data balance ratio (Negative:Positive)
Lima Peru TB [37]	1186	0	6	8	8.65:1
India Attrition	4801	90	2	0	1.9:1
German Credit Scoring	1000	7	11	2	2.33:1
Australian Credit Scoring	690	6	5	3	1.25:1

executable. By creating a wrapper, we can support classifiers written in different languages or which have a different interface. We used wrappers to provide a scikit-learn interface for the ELM and CLC classifier. In addition, the CLC classifier expects a tab separated values (TSV) file as input and produces the classification model and predictions as a TSV file. The wrapper is used to accommodate this.

Different data balancing techniques can be used per classifier. For the data balancing algorithms, we used Lemaître *et al.*'s [40] imbalanced-learn package to provide a scikit-learn compatible implementation.

Every dataset can use its own configuration file which contains the parameters to use for each classifier as well as which data balancing algorithm to use alongside each classifier. Classifiers and datasets can be easily enabled or disabled as desired. We already provide a reasonable search space in our parameter tuning algorithm. However, it is simple to modify or add a new parameter tuning configuration file if a more fine-grained search is required.

After each experiment is completed, the statistics of it will be computed and saved as a CSV file. The relevant graphs will also be generated and saved. These have been used in Section 6.

The system has been designed that it can execute the aforementioned experiments for any binary classification problem not just the ones outlined in this paper. With some modifications, it could also be extended to support multi-class classification problems too. The code has been written in a generic manner that reduces the need to replicate code. This allows new experiments to be created with minimal overhead. Each experiment is also multi-threaded to reduce the execution time.

Our statistical tests were conducted in R because of the lack of availability of a suitable package in Python. We used the PMCMR [52] package for all statistical testing.

6. RESULTS AND DISCUSSION

This section presents and discusses the results of the experiments outlined in Section 4.1.

6.1 Parameter Tuning

The results of the parameter tuning experiment are summarised in Figure 2. For all the datasets, there is a net gain in balanced accuracy from using a data balancer. However, as expected, the Australian credit dataset which is already balanced, only sees very minor improvements. Gaussian naive Bayes is the only classifier which sees a large improvement for the Australia dataset. This is likely because the IHT balancer which was used with it was able to remove samples which are noisy for building the classifier model.

We see the largest improvement in BARR on the Lima TB dataset, incorporating a data balancing technique greatly improves the BACC for all but one classifier. The exception is Bernoulli naive Bayes where there is no difference in BACC for each dataset. We could not find any literature to explain why no difference was seen. In addition, non of the parameters that were searched, affected the classification greatly.

Note that ELM did not have default parameters so we set the default ELM to have one hidden layer with 20 neurons using the sigmoid activation function. We chose this as it provides a reasonable first attempt for the ELM architecture and would likely not over-fit any of our datasets. Per chance, this architecture provided the best classification results for the Lima TB dataset when searching the parameter space. There is therefore no difference between the ELM results using default with balancer or tuned on the Lima TB dataset.

The CLC classifier was excluded as the classifier would crash when applied to the Lima TB dataset without any data balancing technique.

Across all of our datasets, we see marginal improvements on average from tuning our classifiers. Typically the improvement is less than 0.05 from using the default parameters with a data balancer. On the Lima TB dataset the ANN sees an improvement of 0.1 with the tuned parameters over just using the default with a data balancer. On the other datasets we see large improvements for the ELM when tuned. Given that the architecture of a multi-layered perception such as an ANN or ELM While the SVM with RBF and linear kernel see similar results between default with balancer and tuned, the SVM with polynomial kernel sees a large improvement on the credit scoring datasets when tuned.

A classifier that has minor differences between default with balancer and tuned BARR across each dataset is preferred. It shows that the classifier will always produce the best results it can without the user having to manually identify the optimal parameters for each dataset or subset of a dataset that is used to answer a particular research question. Logistic regression, AdaBoost, Random forest, Bernoulli Naive Bayes and SVM with RBF and linear kernel showed similar results between default with balancer and tuned across all our datasets. If these classifiers show promising classification results compared to the other classifiers then they will be recommended. We investigate that in Section 6.2.

6.2 Classifier results

Table 2 and 3 contain the results for each classifier on the Lima TB and India attrition datasets respectively. SVM with linear and polynomial kernel saw very long training times when the ability to produce probabilistic output was enabled for the India attrition dataset. We disabled probabilistic output so results for BS and h-measure are unavailable for those classifiers. The CLC classifier does not support probabilistic output and therefore does not contain results for those metrics either. Figure 3 compares the dif-

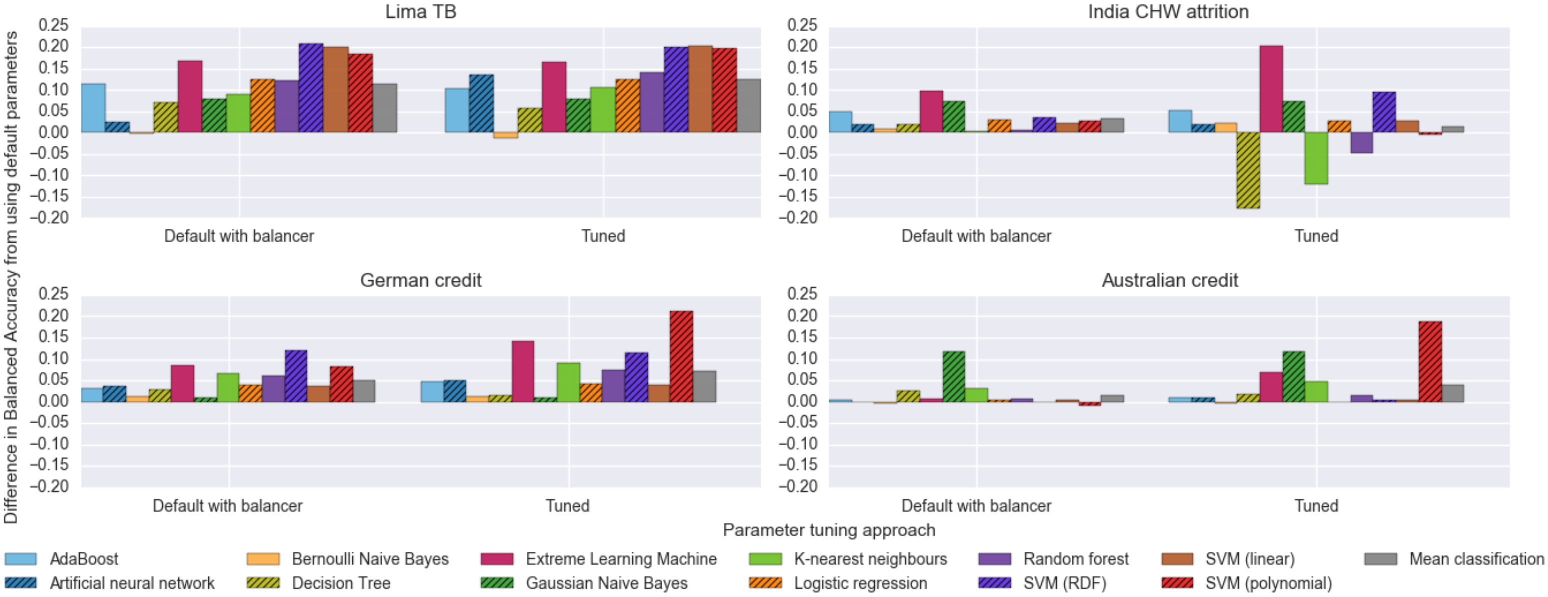

Figure 2: Difference in balanced accuracy from using default parameters with different parameter tuning approaches

ference in TPR and FPR from the median TPR and FPR calculated for every classifier on each dataset. Classifiers that consistently produce better than average results should be in the upper right quadrant. Classifiers in the upper left and lower right can produce better than average results if the improvement in either TPR or TNR exceeds that of the reduction in TNR or TPR.

For all our datasets ANN's and LR produce high quality classification. LR has two benefits over an ANN, it produces a white-box model that can be interpreted to understand the classification process [18] and its training time is orders of magnitude faster than an ANN.

The ANN produces the best BARR for the India attrition dataset. As the ANN can produce a non-linear model it should be able to produce results better than those that are limited to a linear model. However, our statistical tests indicate that that is no statistical difference between the BARR of the ANN and LR ($p = 0.79$) across the datasets. It is possible that our datasets are too small and contain too few features to truly utilise the ANN to its full potential.

Bernoulli NB produced impressive results for the India attrition dataset showing no significant difference in BARR from the ANN ($p = 0.37$). Bernoulli NB has the added benefit that it scales well with data size and number of features. The performance of Bernoulli NB is unexpected as it is designed to operate on binary features. For each dataset we already convert categorical features to multiple binary features as part of the one-hot encoding. However, Bernoulli NB has to apply binarizing on numerical features to transform them into binary features. This process could result in the loss of important information. Since our numerical features are scaled to that of a normal distribution, a binarization threshold of 0 is optimal where input greater than 0 becomes 1 and input lower becomes 0. On our attrition dataset which has 3840 entries per fold and 270 features, once pre-processing is completed, is able to fit a fold in 0.02 seconds. This is the fastest of all the classifiers for the particular dataset. In addition it showed the most stable results in our parameter tuning experiment indicating that minimal tuning is required. Bernoulli NB is therefore well suited for rapid experimentation and prototyping of ideas because of its speedy training and solid classification performance.

The ELM showed BARR on par with SVM with RBF kernel ($p = 0.99$) and ANN ($p = 0.99$) on the two credit scoring datasets matching that of Huang *et al.* [29, 30]. However, on our TB and attrition dataset the ELM was significantly worse than that of the ANN ($p < 0.01$) but obtained similar results to that of the SVM with RBF kernel ($p = 0.99$). The ELM does deliver on its claim that it can learn significantly faster than an ANN. It is 13 times faster on the attrition dataset, 64 times faster on the TB dataset, 189 times faster on the German credit dataset and 802 times faster on the Australian credit scoring dataset.

We were unable to replicate Luo *et al.*'s [45] results for the CLC on the credit scoring datasets. We were only able to achieve a BARR of 0.6158 and 0.6833 on the German and Australian dataset respectively where Luo *et al.* achieved an accuracy of 0.8480 and 0.8652. Since the Australian dataset is balanced, BARR is the same as accuracy. On the Lima TB dataset and India attrition dataset we were unable to achieve results better than random with the CLC classifier. Luo *et al.*'s accuracy of 0.8480 on the German credit scoring dataset is far higher than that seen in the other credit scoring literature utilizing the dataset. The literature typically obtains an accuracy of 0.77 for the dataset [28, 50, 61, 63]. We tried a variety of different formatting for our datasets to ensure that we had it formatted correctly for the CLC classifier. We tested different values for the classification labels, different placement of the label either as the first column or last column in the dataset and the dataset formatted as either CSV or TSV. Our testing indicated that the classification label should be the first column and that it should be -1 for negative and 1 for positive and formatted as a TSV file. Based on our observations, we would not recommend

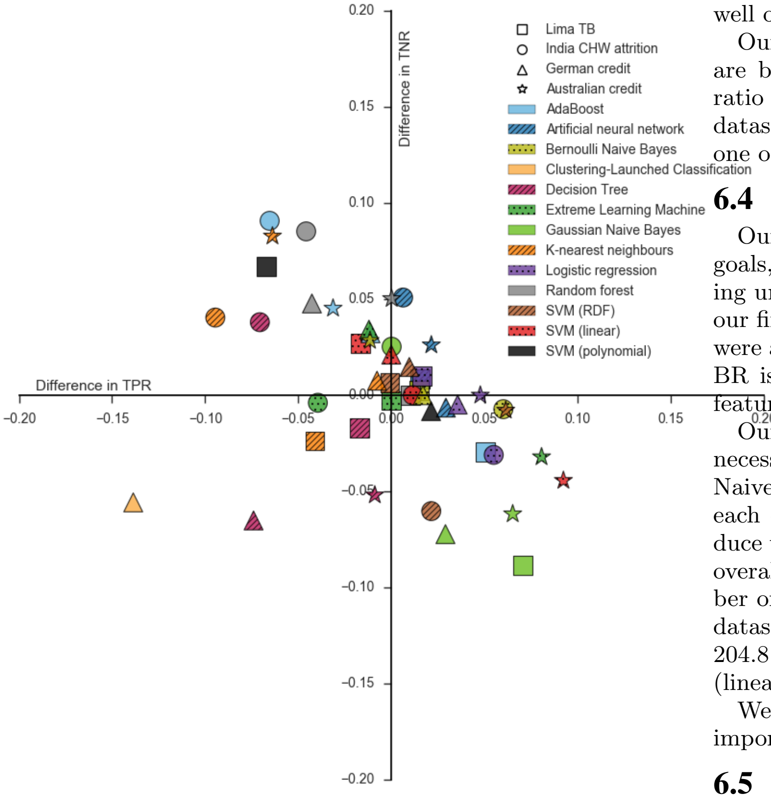

Figure 3: Difference from median TPR and TNR on each dataset for every classifier

the use of CLC for future classification tasks.

Table 4 contains the p-values from the Nemenyi multiple comparison test with the ANN. The ANN ranks highest across all the datasets for BARR, MCC and BS. We reject the null-hypothesis at a p-value < 0.05. The only classifiers which are not significantly different to the ANN is RF and LR.

6.3 Data Balancing

Our results from our data balancing experiments are summarised in Figure 4 and Figure 5. Figure 4 examines each classifier against each data balancer for the Lima TB and India attrition datasets. Some techniques such as SVM, LR, decision tree and RF have the ability to conduct internal class balancing. Unfortunately, this is not available for all classifiers and is implementation specific. The tuned SVM and LR make use of the internal class balancing and showed little variation between different balancing algorithms. They have been excluded from our graphs. Figure 4 suggests that there is no clear winner among the data balancers. In Figure 4 we average each balancer over the classifiers. ADASYN improves TNR for a minor reduction in TPR for the Lima TB dataset while EEN is able to improve the TPR for a minor reduction in TNR for the two credit scoring datasets.

Unlike our classifiers, there is no clear-cut recommendations on which data balancers to use. For our credit scoring datasets, under-sampling techniques such as EEN and NCR produced the best results. For our Lima TB and India attrition dataset, oversampling techniques such as ADASYN produced the best results. SMOTETomek also performed well on the India attrition dataset.

Our observations suggest that over-sampling techniques are best suited to datasets that have a higher imbalance ratio where under-sampling techniques are better suited to datasets that have a lower imbalance ratio. We suggest using one of ADASYN, SMOTETomek, EEN or NCR.

6.4 Feature selection

Our feature selection experiment is divided into two subgoals, identifying the most important predictors and removing un-important features. Figure 6 contains the results for our first sub-goal. Across our datasets RF and decision tree were able to identify the most important features. Bernoulli BR is not recommended for selecting the most important features.

Our second goal was to use feature selection to remove unnecessary features while maintaining overall BARR. Bernoulli Naive Bayes only removed a small subset of features from each dataset. SVM (linear) and RF were both able to reduce the number of features substantially while maintaining overall BARR. However, SVM (linear) reduced the number of features less that that of RF. On the India Attrition dataset the number of features were reduced from 272 to 204.8 and 191 on average for each training fold using SVM (linear) and RF respectively.

We recommend the use of RF for both identifying the most important predictors and removing un-important features.

6.5 Analysis of time to default on Lima TB dataset

The only default range that showed a large variation in BARR from that of the others was training using the 300+ default range. Using this range resulted in a low TPR when applied to the testing set which contains the other half of non-defaulters and defaulters outside this default range.

Our analysis suggests that a defaulter's "profile" is similar regardless of the time they actually default.

7. CONCLUSIONS AND FUTURE WORK

While our comparison of classifiers, data balancers and features selection strategies is extensive we are limited by the datasets that were available to us for this study. The number of samples in our datasets are small compared to what is available in a production environment which limits our ability to see how our results scale to much larger datasets.

Future work includes the use of datasets which are orders of magnitude larger and the incorporation of temporal data into TB default prediction. Temporal data could include information from each check-up improve the prediction. Our TB default data only contained information available at registration.

The use of a data balancing techniques greatly improved BARR for our imbalanced datasets. Parameter tuning only resulted in marginal improvements for most classifiers. The ANN on the TB dataset and ELM on the other datasets saw the greatest improvement in BARR from parameter tuning.

LR, ANN, RF and Bernoulli NB are recommended for several reasons. Bernoulli NB showed above median performance and it can trained incredibly fast. This allows it to scale well for large datasets and facilitates rapid testing and improvement of one's experimental design. ANN produced excellent results in all our datasets and achieved the

Table 2: Lima TB metrics for each classifier

Classifier	BARR[*]	MCC	BS	H-measure	TPR	TNR	Time to fit[†] (s)
ANN	**0.7099 (0.0414)**	**0.3032 (0.0414)**	0.1751 (0.0414)	0.000 (0.0414)	0.6408 (0.0414)	0.7791 (0.0414)	0.2787 (0.0414)
LR	0.7040 (0.0422)	0.3019	**0.1741**		0.6398	0.7788	0.0054
SVM (linear)	0.7002 (0.0419)	0.2959	n/a		0.6064	**0.7939**	0.2066
AdaBoost	0.7002 (0.0432)	0.2764	0.2408		**0.6672**	0.7331	0.0718
RF	0.6985 (0.0436)	0.2829	0.1944		0.6366	0.7604	0.0404
SVM (RDF)	0.6975 (0.0371)	0.2847	0.1965		0.6276	0.7675	0.0109
ELM	0.6951 (0.0469)	0.2806	0.1849		0.6193	0.7709	0.0043
Bernoulli NB	0.6948 (0.0474)	0.2786	0.1892		0.6244	0.7651	0.0006
SVM (poly)	0.6874 (0.0436)	0.2926	n/a		0.5496	0.8252	0.0017
Gaussian NB	0.6847 (0.0588)	0.2660	0.2823		0.6877	0.6818	0.0011
DT	0.6580 (0.0524)	0.2374	0.2167		0.5525	0.7635	0.0005
*k*NN	0.6579 (0.0439)	0.2237	0.2276		0.5662	0.7497	**0.0002**
CLC	0.5344 (0.0642)	0.0462	n/a		0.5525	0.5163	0.0688

[*] Standard deviation in brackets
[†] Time to fit individual fold

Table 3: India attrition metrics for each classifier

Classifier	BARR[*]	MCC	BS	H-measure	TPR	TNR	Time to fit[†] (s)
ANN	**0.8184 (0.0115)**	0.6126	0.1363		0.8213	0.8155	3.7183
Bernoulli NB	0.8172 (0.0130)	0.5986	0.1526		0.8787	0.7558	**0.0194**
RF	0.8125 (0.0152)	**0.6129**	**0.1299**		0.7762	0.8488	1.8219
Gaussian NB	0.8071 (0.0170)	0.5878	0.1969		0.8192	0.7950	0.0276
AdaBoost	0.8051 (0.0142)	0.6024	0.1626		0.7562	**0.8540**	6.4620
LR	0.8025 (0.0134)	0.5693	0.1471		0.8729	0.7320	0.4384
SVM (linear)	0.7965 (0.0108)	0.5623	n/a		0.8296	0.7634	7.9538
DT	0.7728 (0.0166)	0.5274	0.2007		0.7489	0.7967	0.2903
SVM (RBF)	0.7714 (0.0134)	0.5106	0.1662		0.8476	0.6952	5.8640
ELM	0.7694 (0.0150)	0.5134	0.1577		0.7777	0.7611	0.2878
*k*NN	0.7670 (0.0144)	0.5185	0.1590		0.7318	0.8021	0.0642
SVM (poly)	0.5245 (0.0069)	0.1093	n/a		**0.9870**	0.0620	1.4683

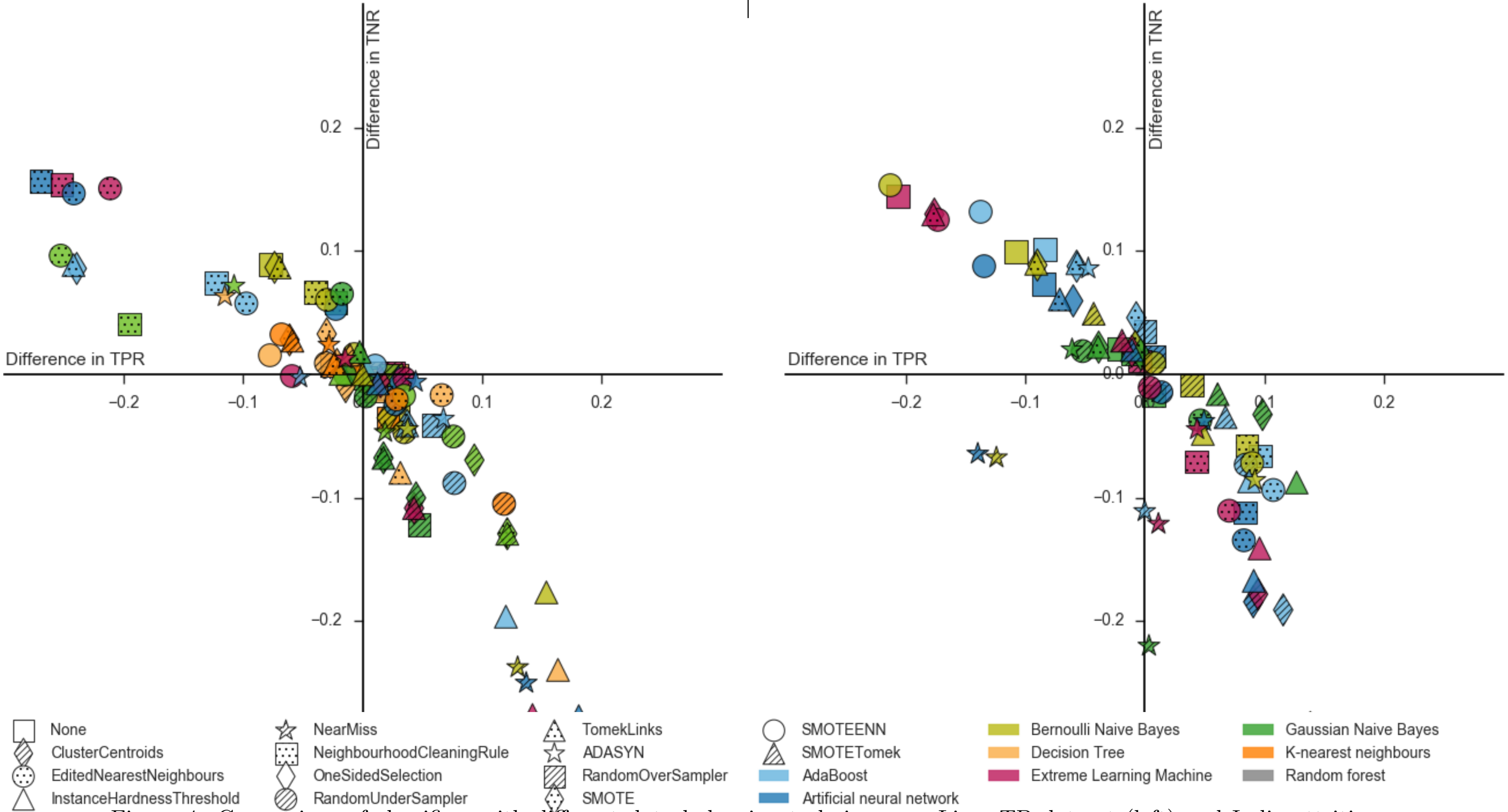

Figure 4: Comparison of classifiers with different data balancing techniques on Lima TB dataset (left) and India attrition dataset (right). SVM and Logistic regression excluded because of minimal variation.

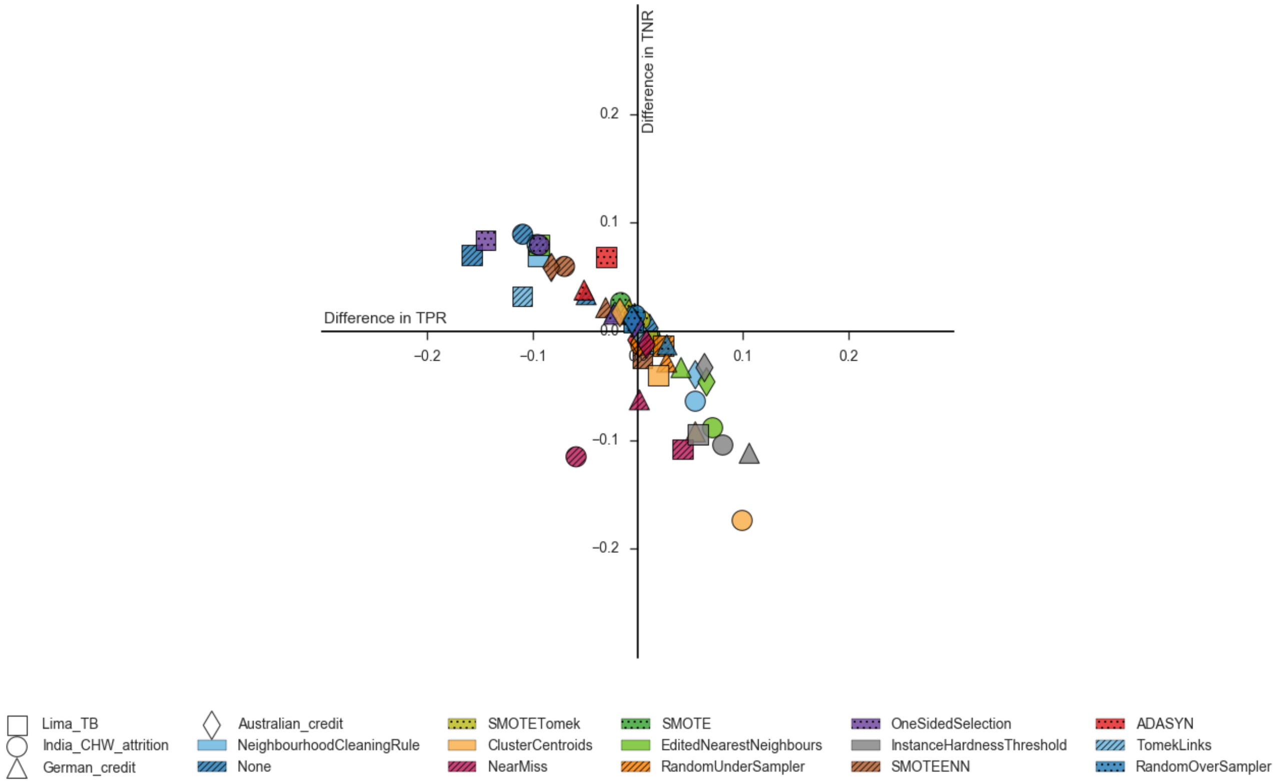

Figure 5: Difference from median TPR and TNR on each dataset for every balancing technique

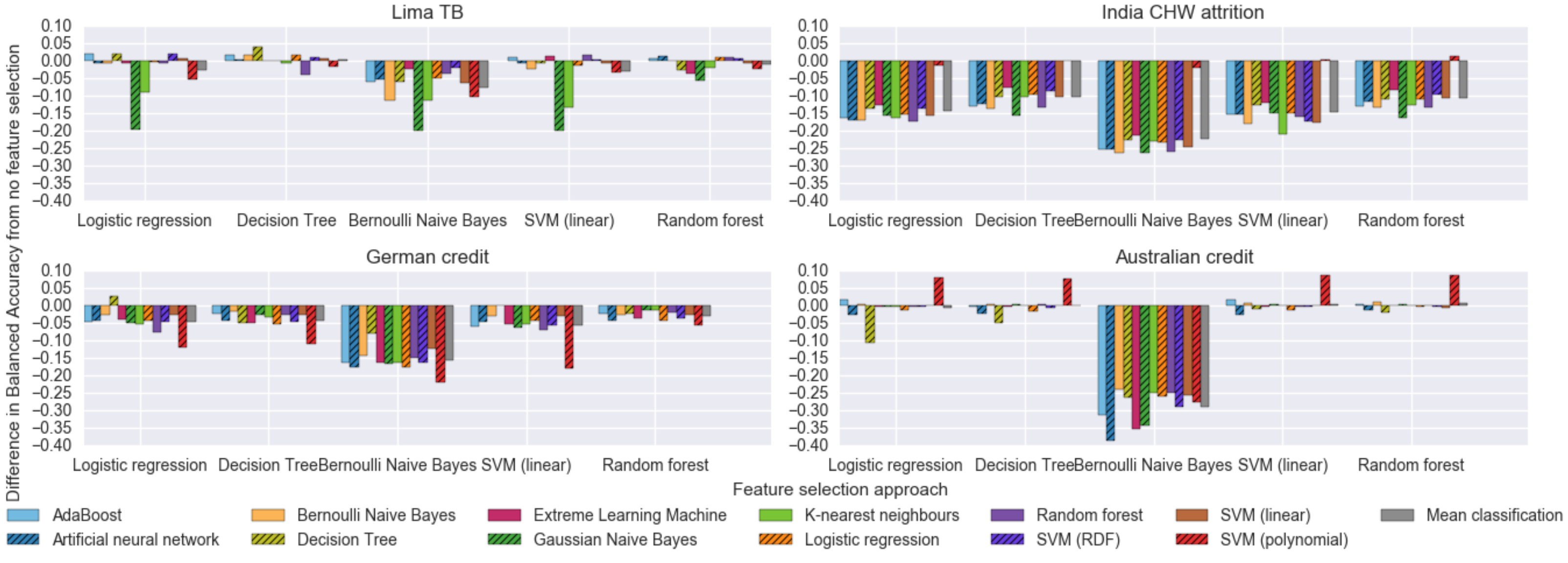

Figure 6: Difference in balanced accuracy from using no feature selection with different feature selection approach forced to select 15 features for each classifier

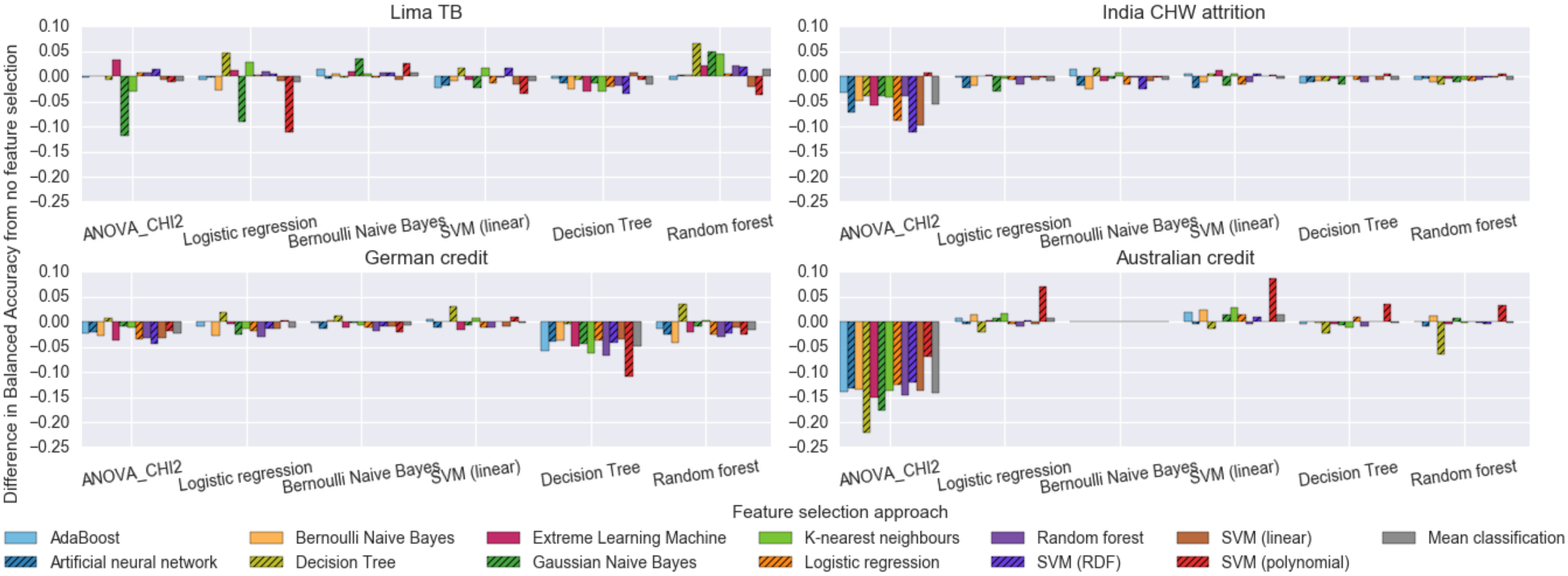

Figure 7: Difference in balanced accuracy from using no feature selection with different feature selection approachs allowed to use any number of features

Table 4: Pairwise comparison using Nemenyi multiple comparison against ANN for BARR, MCC and BS across all datasets

Classifier	BARR (p-value)	MCC (p-value)	BS (p-value)
Bernoulli NB	0.0027	0.0002	<0.0001
RF	0.89797	1.0	1.0
Gaussian NB	<0.0001	<0.0001	<0.0001
AdaBoost	<0.0001	0.0103	<0.0001
LR	0.7912	0.27606	0.0677
SVM (linear)	0.0010	0.0281	n/a
DT	<0.0001	<0.0001	<0.0001
SVM (RBF)	<0.0001	<0.0001	<0.0001
ELM	<0.0001	<0.0001	0.0003
kNN	<0.0001	<0.0001	<0.0001
SVM (poly)	<0.0001	<0.0001	n/a
CLC	<0.0001	<0.0001	n/a

highest BARR and MCC in the India attrition dataset. LR produced the good BARR and MCC in the Lima TB and German credit dataset. It also showed competitive performance in the other two datasets. LR produces a white-box model which allows for easier interpretation of the classification process.

There was no stand-out data balancing algorithm. However, ADASYN was best on the Lima TB dataset and ENN was best on the two credit scoring datasets.

Most of papers in Section 3.1 used LR to determine the predictors associated with TB default. Our results suggest that RF is better at identifying important predictors. Random forest also showed the best performance when RFE was applied. It was able to both reduce the number of features while retaining similar BARR and in some cases was able to improve the BARR too.

8. REFERENCES

[1] H. A. Abdou, M. D. D. Tsafack, C. G. Ntim, and R. D. Baker. Predicting creditworthiness in retail banking with limited scoring data. *Knowledge-Based Systems*, 2016.

[2] A. Akusok, K. M. BjÃűrk, Y. Miche, and A. Lendasse. High-performance extreme learning machines: A complete toolbox for big data applications. *IEEE Access*, 3:1011–1025, 2015.

[3] E. Angelini, G. di Tollo, and A. Roli. A neural network approach for credit risk evaluation. *The Quarterly Review of Economics and Finance*, 48(4):733–755, 2008.

[4] G. E. Batista, A. L. Bazzan, and M. C. Monard. Balancing training data for automated annotation of keywords: a case study. In *WOB*, pages 10–18, 2003.

[5] G. E. A. P. A. Batista, R. C. Prati, and M. C. Monard. A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explor. Newsl.*, 6(1):20–29, June 2004.

[6] H. A. Bekhet and S. F. K. Eletter. Credit risk assessment model for jordanian commercial banks: Neural scoring approach. *Review of Development Finance*, 4(1):20–28, 2014.

[7] J. Bergstra, N. Casagrande, D. Erhan, D. Eck, and B. Kégl. Aggregate features and adaboost for music classification. *Machine Learning*, 65(2):473–484, 2006.

[8] M. Bermingham, R. Pong-Wong, A. Spiliopoulou, C. Hayward, I. Rudan, H. Campbell, A. Wright, J. Wilson, F. Agakov, P. Navarro, and C. Haley. Application of high-dimensional feature selection: evaluation for genomic prediction in man. *Scientific reports*, 5:10312, 2015.

[9] A.-L. Boulesteix, S. Janitza, J. Kruppa, and I. R. KÃűnig. Overview of random forest methodology and practical guidance with emphasis on computational biology and bioinformatics. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(6):493–507, 2012.

[10] M. Chan-Yeung, K. Noertjojo, C. Leung, S. Chan, and C. Tam. Prevalence and predictors of default from tuberculosis treatment in hong kong. *Hong Kong Medical Journal*, 9(4):263–270, 2003.

[11] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *J. Artif. Int. Res.*, 16(1):321–357, June 2002.

[12] N. V. Chawla, N. Japkowicz, and A. Kotcz. Editorial: Special issue on learning from imbalanced data sets. *SIGKDD Explor. Newsl.*, 6(1):1–6, June 2004.

[13] T.-S. Chen, C.-C. Lin, Y.-H. Chiu, H.-L. Lin, and R.-C. Chen. *A New Binary Classifier: Clustering-Launched Classification*, pages 278–283. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[14] I. Cherkaoui, R. Sabouni, I. Ghali, D. Kizub, A. C. Billioux, K. Bennani, J. E. Bourkadi, A. Benmamoun, O. Lahlou, R. E. Aouad, and K. E. Dooley. Treatment default amongst patients with tuberculosis in urban morocco: Predicting and explaining default and post-default sputum smear and drug susceptibility results. *PLoS ONE*, 9(4):1–9, April 2014.

[15] P. Danenas and G. Garsva. Selection of support vector machines based classifiers for credit risk domain. *Expert Systems with Applications*, 42(6):3194–3204, 2015.

[16] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, Dec. 2006.

[17] V. S. Desai, J. N. Crook, and G. A. O. Jr. A comparison of neural networks and linear scoring models in the credit union environment. *European Journal of Operational Research*, 95(1):24–37, 1996.

[18] S. Dreiseitl and L. Ohno-Machado. Logistic regression and artificial neural network classification models: a methodology review. *Journal of Biomedical Informatics*, 35(5âĂŞ6):352 – 359, 2002.

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

[20] S. Garcıa and F. Herrera. An extension on âĂIJstatistical comparisons of classifiers over multiple data setsâĂİ for all pairwise comparisons. *Journal of Machine Learning Research*, 9:2677–2694, 2008.

[21] K. Gowda and G. Krishna. The condensed nearest neighbor rule using the concept of mutual nearest neighborhood (corresp.). *IEEE Transactions on Information Theory*, 25(4):488–490, Jul 1979.

[22] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, Mar. 2003.

[23] D. J. Hand. Measuring classifier performance: A coherent alternative to the area under the roc curve. *Mach. Learn.*, 77(1):103–123, Oct. 2009.

[24] H. He, Y. Bai, E. A. Garcia, and S. 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, June 2008.

[25] K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991.

[26] N.-C. Hsieh and L.-P. Hung. A data driven ensemble classifier for credit scoring analysis. *Expert Systems with Applications*, 37(1):534–545, 2010.

[27] C.-W. Hsu, C.-C. Chang, C.-J. Lin, et al. A practical guide to support vector classification. *Department of Computer Science, National Taiwan University.*

[28] C.-L. Huang, M.-C. Chen, and C.-J. Wang. Credit scoring with a data mining approach based on support vector machines. *Expert Systems with Applications*, 33(4):847–856, 2007.

[29] G. B. Huang, H. Zhou, X. Ding, and R. Zhang. Extreme learning machine for regression and multiclass classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 42(2):513–529, April 2012.

[30] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew. Extreme learning machine: Theory and applications. *Neurocomputing*, 70(1âĂŞ3):489 – 501, 2006. Neural NetworksSelected Papers from the 7th Brazilian Symposium on Neural Networks (SBRN '04)7th Brazilian Symposium on Neural Networks.

[31] Z. Huang, H. Chen, C.-J. Hsu, W.-H. Chen, and S. Wu. Credit rating analysis with support vector machines and neural networks: a market comparative study. *Decision Support Systems*, 37(4):543–558, 2004. Data mining for financial decision making.

[32] U. M. Jha, S. Satyanarayana, P. K. Dewan, S. Chadha, F. Wares, S. Sahu, D. Gupta, and L. S. Chauhan. Risk factors for treatment default among re-treatment tuberculosis patients in india, 2006. *PLoS ONE*, 5(1):1–7, January 2010.

[33] S. X. Jittimanee, E. A. Madigan, S. Jittimanee, and C. Nontasood. Treatment default among urban tuberculosis patients, thailand. *International Journal of Nursing Practice*, 13(6):354–362, 2007.

[34] G. H. John and P. Langley. Estimating continuous distributions in bayesian classifiers. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, UAI'95, pages 338–345, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[35] J. M. Keller, M. R. Gray, and J. A. Givens. A fuzzy k-nearest neighbor algorithm. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-15(4):580–585, July 1985.

[36] M. Kubat and S. Matwin. Addressing the curse of imbalanced training sets: One-sided selection. In *In Proceedings of the Fourteenth International Conference on Machine Learning*, pages 179–186. Morgan Kaufmann, 1997.

[37] B. Lackey, C. Seas, P. Van der Stuyft, and L. Otero. Patient characteristics associated with tuberculosis treatment default: A cohort study in a high-incidence area of lima, peru. *PLoS ONE*, 10(6):1–11, 2015.

[38] J. Laurikkala. Improving identification of difficult small classes by balancing class distribution. In *Proceedings of the 8th Conference on AI in Medicine in Europe: Artificial Intelligence Medicine*, AIME '01, pages 63–66, London, UK, UK, 2001. Springer-Verlag.

[39] T.-S. Lee, C.-C. Chiu, C.-J. Lu, and I.-F. Chen.

Credit scoring using the hybrid neural discriminant technique. *Expert Systems with Applications*, 23(3):245–254, 2002.

[40] G. Lemaître, F. Nogueira, and C. K. Aridas. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *CoRR*, abs/1609.06570, 2016.

[41] S. Lessmanna, H. Seowb, B. Baesenscd, and L. C. Thomasd. Benchmarking state-of-the-art classification algorithms for credit scoring: A ten-year update. In *Credit Scoring and Credit Control XIII*, Edinburgh, UK, 2013.

[42] D. D. Lewis. *Naive (Bayes) at forty: The independence assumption in information retrieval*, pages 4–15. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

[43] S.-T. Li, W. Shiue, and M.-H. Huang. The evaluation of consumer loans using support vector machines. *Expert Systems with Applications*, 30(4):772–782, 2006.

[44] M. Lichman. UCI machine learning repository, 2013.

[45] S.-T. Luo, B.-W. Cheng, and C.-H. Hsieh. Prediction model building with clustering-launched classification and support vector machines in credit scoring. *Expert Systems with Applications*, 36(4):7562–7566, 2009.

[46] R. Malhotra and D. Malhotra. Evaluating consumer loans using neural networks. *Omega*, 31(2):83–96, 2003.

[47] A. McCallum, K. Nigam, et al. A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 752, pages 41–48, 1998.

[48] C. Mood. Logistic regression: Why we cannot do what we think we can do, and what we can do about it. *European Sociological Review*, 26(1):67–82, 2010.

[49] B. Muture, M. N. Keraka, P. K. Kimuu, E. W. Kabiru, V. O. Ombeka, and F. Oguya. Factors associated with default from treatment among tuberculosis patients in nairobi province, kenya: A case control study. *BMC Public Health*, 11(1):696–105, September 2011.

[50] L. Nanni and A. Lumini. An experimental comparison of ensemble of classifiers for bankruptcy prediction and credit scoring. *Expert Systems with Applications*, 36(2, Part 2):3028–3033, 2009.

[51] 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.

[52] T. Pohlert. *The Pairwise Multiple Comparison of Mean Ranks Package (PMCMR)*, 2014. R package.

[53] I. Rish. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, pages 41–46. IBM New York, 2001.

[54] R. Rojas. Adaboost and the super bowl of classifiers a tutorial introduction to adaptive boosting, Computer Science Department, Freie Universität, Berlin. 2009.

[55] S. Sandro. Understanding logistic regression analysis. *biochem*, 24(1):12–18, 2014.

[56] E. B. Shargie and B. Lindtjorn. Determinants of treatment adherence among smear-positive pulmonary tuberculosis patients in southern ethiopia. *PLoS Med*, 4(2):1–8, February 2007.

[57] M. R. Smith, T. Martinez, and C. Giraud-Carrier. An instance level analysis of data complexity. *Mach. Learn.*, 95(2):225–256, May 2014.

[58] E. W. Steyerberg, A. J. Vickers, N. R. Cook, T. Gerds, M. Gonen, N. Obuchowski, M. J. Pencina, and M. W. Kattan. Assessing the performance of prediction models: a framework for some traditional and novel measures. *Epidemiology (Cambridge, Mass.)*, 21(1):128–138, 2010.

[59] V. Svetnik, A. Liaw, C. Tong, J. C. Culberson, R. P. Sheridan, and B. P. Feuston.

[60] I. Tomek. Two modifications of cnn. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6(11):769–772, Nov 1976.

[61] C.-F. Tsai and J.-W. Wu. Using neural network ensembles for bankruptcy prediction and credit scoring. *Expert Systems with Applications*, 34(4):2639–2649, 2008.

[62] B. Twala. Multiple classifier application to credit risk assessment. *Expert Systems with Applications*, 37(4):3326–3336, 2010.

[63] G. Wang, J. Hao, J. Ma, and H. Jiang. A comparative assessment of ensemble learning for credit scoring. *Expert Systems with Applications*, 38(1):223–230, 2011.

[64] S.-C. Wang. *Interdisciplinary Computing in Java Programming*, chapter Artificial Neural Network, pages 81–100. Springer US, Boston, MA, 2003.

[65] World Health Organisation. Global tuberculosis report 2015.

[66] I. Zhang and I. Mani. knn approach to unbalanced data distributions: a case study involving information extraction. In *Proceedings of workshop on learning from imbalanced datasets*, 2003.