An Analysis of Classification Techniques for the Prediction of Treatment Defaulters

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

ABSTRACT

1. INTRODUCTION

In 2013 over 210 000 patients defaulted from Tuberculosis (TB) treatment worldwide [45]. The rate of default in the Americas is the highest at 8% with Africa at 5% [45]. The consequences of defaulting TB treatment include: increased drug resistance, increased health system costs [27, 36], higher risk of mortality, continued risk of transmitting the disease to others [27] and increased rate of recurrent disease [22]. 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.

In this paper we explore several issues that may arise when attempting to apply classification techniques to predict minority class events such as treatment default. We explore a number of techniques and strategies to improve the overall classification result. Most notably we look at how well these classifiers work out-of-the-box compared when they are tuned by searching a grid of parameters. To address class imbalance, we examine different data balancing techniques which either over-sample the minority class or under-sample the majority class. Furthermore, to see how each classifier responds to having fewer features available, we apply a number of feature selection strategies on each classification technique.

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.

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. The selected classification techniques are evaluated against real-world treatment default datasets and financial datasets. The paper will evaluate and compare how the techniques differ across the datasets. If the relative results are similar then future credit scoring research could be applicable to treatment default prediction too.

© 2016 ACM. ISBN n/a.
DOI: n/a

2. RELATED WORK

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 [8, 12, 22, 23, 36, 45].

2.2 Determining predictors of TB default

There have been many studies which focus on determining the factors associated with TB default but few have used machine learning techniques to predict treatment defaulters. Table **??** contains an overview of a selection of publications on determining the factors associated with TB default. The majority of techniques use a form of logistic regression ti 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.* [41]. Lackey *et al.* [27] only picked individuals who did not have a history of past default. Jittimanee [23] [28] 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.* [41] and Jittimanee *et al.* [23] measured distance and time to treatment site respectively. It can be reasoned that the aforementioned feature's significance will generalise to 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. BACKGROUND

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

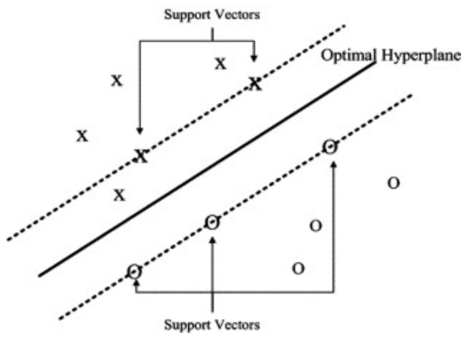

Figure 1: An overview of an SVM [31]

3.1 Classifiers

This section examines the different classification techniques and outlines how each technique produces its output.

3.1.1 Support Vector Machines

Support vector machine (SVM) is a machine learning technique that can be used to produce regression or classification functions from a set of training data [33]. SVM works by mapping the input vectors into a high-dimensional feature space with the use of a kernel function [13]. The kernel function selected determines the if the mapping is linear or non-linear [33]. Linear, polynomial and radial basis function (RBF) are common kernel functions [19]. The polynomial and RBF kernel performs a non-linear mapping into the high-dimensional space [19]. This feature space is then searched to acquire an optimal hyperplane that separates the space with the maximum distance between the two classes [13]. Figure 1 shows an example of this. Hsu *et al.* [19] 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.

3.1.2 Artificial Neural Network

An artificial neural network (ANN) is based on the functionality of the human brain [44]. Neurons in the brain are interconnected and process information in parallel [44]. 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 [44]. The activation function bounds the value of a neuron to a specific range to limit the effect of divergent neurons [44]. By using an activation function, a non-linear combination of weights can be generated [44]. 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 [18].

Extreme learning machines (ELM) is an alternative approach to the conventional back-propagated ANN. Huang *el al.* [21] 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 [21]. This allows ELMs to be trained orders of magnitude faster than a back-propagated ANN [20, 21]. ELMs have been shown to provide better results than SVMs and ANNs on a variety of classification and regression tasks [20, 21].

3.1.3 Logistic Regression

Logistic regression is a technique that models the chance of an outcome based on the input features [40]. Since chance is a ratio, the logarithm of the chance is modelled instead [40]: $\log(\frac{p}{1-p}) = \beta_0 + \beta_1 x_1 + ... + \beta_m x_m$. 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 $p = \frac{1}{1+e^{-(\beta_0+\beta_1 x_1+\dots+\beta_m x_m)}}$. A detailed overview of logistic regression can be found in [35] and [40].

3.1.4 k-Nearest Neighbours

The k-Nearest Neighbours (kNN) algorithm determines the output classification by examining the k nearest training examples in the feature space [25]. An input is classified by the majority vote of its neighbours.

3.1.5 Ensembles

An ensemble classifier is one which typically makes use of the aggregation of multiple classifiers. A single decision tree 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 [43].

Random forest 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 [7]. Samples that were not selected for training the tree is called out-of-bag data [7]. It is used to test the error rate of the forest [7, 43]. The output classification is decided by the majority vote of each tree [7].

Freund and Schapire's [14] 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 [5, 39]. 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 [5, 39]. 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 [5].

3.1.6 Naive Bayes

Naive Bayes makes an assumption that each input feature is independent of each other [30, 38]. This allows multiple features be uncoupled from one another which simplifies the algorithm. Naive Bayes uses conditional probability to classify new inputs [30]. 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 [30, 38]. Since $p(x)$ is identical for each class, it is typically ignored [38]. 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 [30].

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 [24]. This allows support of continuous x_i values [24]. The multivariate Bernoulli event model assumes that features are independent boolean values [34].

3.1.7 Clustering-launched classification

Clustering-launched classification (CLC) is a binary classifier. CLC first clusters the data into groups using a diverse hierarchical k-means algorithm [33]. The clusters are divided into positive subsets and negative subsets [33]. Support vectors are then used to separate the positive and negative subsets for each cluster [33]. Redundant or repeated support vectors are removed thereafter [33].

3.2 Data Balancers

This section examines the different data balancing approaches and their respective algorithms.

3.2.1 Over-sampling

Random over-sampling (ROS) is a technique that randomly replicates examples in the minority class [4]. However, this technique can increase the likelihood of over-fitting since exact copies are made from the minority class [4].

The Synthetic minority over-sampling technique (SMOTE) [9] 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) [17] 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.

3.2.2 Under-sampling

Random under-sampling (RUS) is a technique that randomly eliminates examples in the majority class [4]. However, this technique can remove potentially useful information from the training set [4].

Condensed Nearest Neighbour (CNN) rule [15] 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 [4]. The process draws one random majority class example and all minority class examples and puts them in D [4]. Every misclassified example is then added from E to D [4]. This process attempts to remove examples that are far away from the decision border and therefore seen as less relevant for learning [4].

The Tomek link (TL) algorithm [1] 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 [4]. This information can then be used to under-sample the majority class [4].

One-sided selection (OSS) [26] 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) [28] 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 [4]. If $\mathbf{x}_i$ belongs to the minority class and the neighbours contradict this class then the neighbours from the majority class are removed [4].

Instance threshold hardening (ITH) [42] 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) [46] 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.

3.2.3 Combination

SMOTE + TL [3] 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 [3]. TL is used as a data cleaning method to remove examples from both classes to produce well-defined class clusters [3].

SMOTE + ENN works similarly to SMOTE + TL but ENN is more aggressive at removing samples than TL [4].

3.3 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 [4, 10]. 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: $\frac{(TP \times TN)-(FP \times FN)}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$. Informedness is similar to balanced accuracy but ranges between -1 and 1 and is defined as $TPR + TNR - 1$.

4. METHODS

4.1 Approach

We aimed to evaluate a set of statistical and machine learning classification techniques on treatment default data sets and compare it to that applied to other related classification datasets. A variety of classification techniques were chosen to test against: ANN, SVM using a linear, RBF and polynomial kernel, logistic regression, decision tree, AdaBoost, random forest, kNN, Gaussian naive Bayes, Bernoulli naive Bayes, ELM and CLC. These were chosen to include a selection of well known techniques, ensemble techniques and newer techniques that have shown promising results in other studies.

Many datasets have substantially fewer defaulters than those cured. However, most classification algorithms expect a 50:50 split between classes. We therefore include an analysis of several data balancing algorithms: CC, ENN, IHS, NM-1, NCR, OSS, RUS, TL, ADASYN, ROS, SMOTE, ensemble of SMOTE + EEN and ensemble of SMOTE + TL. These balancing techniques were chosen 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.

All the experiments use stratified 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. Results for each fold 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 examples. Another approach is to remove examples that contain missing values. We opted to remove examples 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

We want to determine how important 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, informedness and time to fit each training fold. In addition, a receiver operating characteristic (ROC) curve is also plotted as an additional means of comparing the classification techniques and to indicate what TPR can be achieved for an acceptable false positive rate (FPR). 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.

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.

A bar chart is plotted with each classifier against the BARR for the treatment default datasets. As in section 4.1.2, a scatter plot is presented which compares difference of true positive and false positive from the median of each for each classifier on each data balancing algorithm on each dataset.

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 [16]. It can also speed up training times for large datasets [16]. 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 [6].

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 other feature selection techniques in two different ways. In the first approach we use the median feature weight as the threshold and only keep features that are rated more important. 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. 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 least important feature at each iteration until only one feature remains. 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 Evaluation

The experiments were run on a set of real-world datasets: one TB default dataset from Lima Peru [27] and two credit scoring datasets. The TB dataset was obtained from the Dryad digital repository. The two credit scoring datasets (later referred to as the German and Australian dataset) were obtained from the UCI machine learning repository [32]. Table 1 provides an overview of the characteristics of the dataset. The Peru TB dataset is highly imbalanced which makes it an ideal dataset to test different balancing algorithms. The German dataset is slightly imbalanced while the Australian dataset is balanced. The Peru data set came with values pre-discretized and therefore only contains categorical features.

5. IMPLEMENTATION

We developed a system in Python 2.7 that makes it simple to execute the experiments outlined in section 4.1 against multiple datasets and classifiers. In addition the system has been designed to allow for easy support of additional datasets and classifiers. 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 ten lines in the configuration file. All pre-processing operations such as removal of examples 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 [37] 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 [11] as a compiled C++ 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 [29] 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 have already provided 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. To improve overall

6. RESULTS

1. Tables with true positive, true negative, false positive and false negative rate for each classifier on each

Table 1: Data set summary

Data set	Entries	Number of numerical features	Number of categorical features	Data balance ratio (Cured:Default)
Lima Peru TB [27]	1186	0	15	8.65:1
German Credit Scoring	1000	7	13	2.33:1
Australian Credit Scoring	690	6	8	1.25:1

dataset

2. ROC curves which can be used to determine true positive for an acceptable amount of false positives for each classification technique on each dataset
3. Graphs which outline difference in accuracy of each data balancing technique for each dataset
4. Discussion on best classification technique
5. Discussion on data balancing results
6. Outline similarities and differences in results between the two TB datasets and between TB and financial datasets and determine if they are similar enough that results for the German and Australian credit scoring datasets could be used for the two TB datasets.

7. CONCLUSIONS AND FUTURE WORK

1. Likely something along the lines of utilising more temporal based data so that classification is not just done at registration but also at each check-up for example. Future work may also be the testing of more classification techniques as well as datasets from other parts of the world.

8. REFERENCES

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

[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] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

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

[17] 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.

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

[19] 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.*

[20] 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.

[21] 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.

[22] 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.

[23] 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.

[24] 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.

[25] 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.

[26] 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.

[27] 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.

[28] 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.

[29] 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.

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

[31] 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.

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

[33] 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.

[34] 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.

[35] 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.

[36] 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.

[37] 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.

[38] 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.

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

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

[41] 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.

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

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

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

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

[46] 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.