Bachelor's Thesis

Integrating Class-dependent Misclassification Cost into Feature Selection

Integration klassenspezifischer Missklassifikationskosten in Feature-Selection-Algorithmen

Daniel Oliver Theveßen
daniel.thevessen@student.hpi.de

Submitted on July 20, 2017

Knowledge Discovery and Data Mining Group
Hasso Plattner Institute, Germany

Supervisors
Arvind Kumar Shekar
Prof. Dr. Emmanuel Müller

Abstract

A frequent problem in classification is class imbalance, a state in which certain classes in a dataset are significantly more frequent than others. In these cases, labeling all samples as the most frequent class would produce a high accuracy, yet such a solution could not be described as a good classification. In other words, the cost of misclassifying a single sample is different for an infrequent class than for a frequent class. The field of cost-sensitive classification is well-researched. However, for high-dimensional data, feature selection as an additional preprocessing step is advisable in order to reduce noise, alleviate the curse of dimensionality, and improve classification accuracy in general. It is generally accepted that traditional, unadapted feature selection algorithms are not completely appropriate for imbalanced data. Several feature selection algorithms taking class imbalance into account have been proposed, focusing on binary classification problems. We present a novel feature selection algorithm that mitigates class imbalance in multiclass problems through the use of a weighting function. Further, this function allows it to be combined with known, externally motivated, misclassification cost depending on the individual dataset. We evaluate this approach using datasets from the UCI Machine Learning repository, and compare it to existing algorithms.

Contents

1 Introduction **1**
1.1 Contribution 2
1.2 Structure of the Thesis 2

2 Problem Definition **3**
2.1 Feature Selection 3
2.2 Problem setting 4

3 Related Work **6**

4 Weighted Relevance and Redundancy Scoring **9**
4.1 Relevance and Redundancy Scoring 9
4.2 General idea 10
4.2.1 Class-wise statistical dependence 10
4.2.2 Class Divergence 12
4.3 Pseudocode 14
4.4 Extension to Constraint Solving 15
4.5 Relationship to other divergences 16

5 Implementation Details **17**

6 Experiments **18**
6.1 Synthetic datasets 18
6.2 Real-world datasets 20

7 Conclusions **23**
7.1 Future Work 23

8 Appendix **24**
8.1 Relationship to Kullback-Leibler Divergence 24

References **27**

1 Introduction

A common problem in datasets of all kinds is the existence of class imbalance, a condition in which certain classes have significantly fewer samples in comparison to others. Minimizing general error rate in this case can lead to these classes being underestimated or even completely ignored by predictive models. As a result, the relative error rate, the ratio of incorrectly predicted values to real values on a per-class basis, may be significant for seldom occurring classes.

There are several frequently used methods to alleviate class imbalance [1]. The most common type of method focuses on artificially rebalancing the training set through either oversampling or undersampling [2][3][4], thus adjusting the general error rate to match the averaged relative error rate. Disadvantages of sampling methods have been known to be loss or distortion of information due to its artificial nature [5]. Other methods aim to apply a weighting function that benefits smaller classes, keeping all information while producing a similar result. One of the advantages of these weightings is the fact that they can be combined with known, externally motivated, misclassification cost. This is a common occurrence in fields such as face recognition or intrusion detection, in which misclassifying authorized access is not as costly as misclassifying an intruder [6][7].

The field of cost-sensitive classification is well-researched. However, for high-dimensional data, the reduction of dimensionality as an additional preprocessing step is advisable in order to reduce noise, alleviate the curse of dimensionality, and improve classification accuracy in general. The two principal types of dimensionality reduction are feature extraction, in which relevant information within each feature is extracted and condensed into a feature set, and feature selection, in which a subset of features as a whole is selected to be used for classification. Many real-world problems, in addition to classification accuracy, require a reduction in the amount of sensors that need to be measured. Since this is a requirement that can not be fulfilled with a set of feature combinations, we will be focusing on feature selection instead of extraction in this thesis.
Feature selection algorithms not considering class imbalance or cost may be unsuitable for cost-sensitive classifiers as the feature subset will have been selected without consideration for differing impact of false positives or false negatives. There is a general consensus that unadapted feature selection algorithms are not completely appropriate for imbalanced datasets [8]. This makes cost-sensitive feature selection necessary, which will be the topic of this thesis.

1.1 Contribution

We will be contributing a novel weighting method that extends an existing feature selection method in order to select the best features for an imbalanced dataset. Therefore, our primary use case will be the analysis of high-dimensional, imbalanced, datasets. Similar to comparable approaches for cost-sensitive classification, a weighting will be applied based on a cost function. This cost function will be generated at the beginning based on statistical imbalance in the supplied dataset, and may also be combined with a predetermined cost function passed to the algorithm. By applying a weighting based on information gathered as part of the feature selection algorithm's statistical analysis, there is no significant increase in runtime.

1.2 Structure of the Thesis

This thesis is divided into seven parts. In the current section, we specify the area of problems to be addressed by our contributions, demonstrate their need in various applications, and lay out the structure of the thesis.

In section 2, we will specify the problem definition and explain the underlying assumptions. This includes setting the notation and mathematical definitions for all further work.

section 3 will present the current state of research in this particular area, contrasting the contribution to existing approaches and illustrating their differences. Furthermore, suitable candidates for comparison will be chosen and explained in detail.

The details of our new algorithm, as well as its formal basis, will be explained in section 4.

The implementation of this algorithm will be outlined in section 5, focusing particularly on implementation-specific optimizations and limitations.

In section 6, our contribution will be evaluated and compared to competing algorithms with experiments on both synthetic and published data. Subsequently, the results will be analyzed and summarized.

Finally, we will draw conclusions regarding the algorithm's merit in comparison to others in section 7. We will highlight the use cases where it is most applicable and most beneficial, and discuss its mathematical and computational limitations. In this context, we will lay out how weaknesses could be targeted and what other potential improvements could be added in future works.

2 Problem Definition

This section provides a mathematical specification of the stated problem. We present all mathematical definitions relevant to our approach and explain the underlying assumptions.

2.1 Feature Selection

Before we can introduce the specific problem we are tackling within the feature selection context, we must first outline some basic definitions relating to feature selection in general.

Definition 1 (General Classification Accuracy)**.** Given a Classifier K and a dataset D. For every object o in D there is a predicted class $C_p(o)$ and a real class $C_r(o)$. The Classification Accuracy is defined as

$$E_D(K) = \frac{|\{o \in D \mid C_p(o) = C_r(o)\}|}{|D|}.$$

Generally, one of the objectives of feature selection is to select a subset that will lead to maximal classification accuracy.

Definition 2 (Feature Selection)**.** Given an arbitrary Classifier K and a d-dimensional dataset $D = (X, Y)$ with a set of feature vectors $X = \{x_1, x_2, \ldots, x_d\}$ and a class vector Y. We want to select a subset $S_i \subseteq X$, given $D_{S_i} = (S_i, Y)$, with the property

$$\max_{\forall S_i \subseteq X} E_{D_{S_i}}(K).$$

This is a rough definition of a commonly used objective for feature selection. However, features do not have to be selected with a specific classifier in mind. Furthermore, it is important to note that feature selection, while often optimized for classification accuracy, serves multiple benefits, among these are

- Improving classification accuracy (see also curse of dimensionality [9])
- Reducing overfitting [10]
- Reducing the amount of measurements that need to be taken, e.g. fewer sensors on a device
- Reducing classification training time

Our contribution will be further improving the classification accuracy in the specific case of class imbalance, evaluated by a measure defined in the next subsection.

This thesis builds on an existing feature selection algorithm utilizing divergence. Therefore, we will quickly define the properties of a statistical divergence function for a better understanding of section 4.

Definition 3 (Divergence)**.** A statistical divergence function is a function $D(P \parallel Q)$ between two probability distributions P and Q that satisfies the properties [11]

1. $\forall P, Q : D(P \parallel Q) \geq 0$
2. $D(P \parallel Q) = 0 \iff P = Q.$

A particular instance of divergence function that will come up in this thesis is the Kullback-Leibler divergence [12], which is defined as follows

$$D_{KL}(P \parallel Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)}.$$

2.2 Problem setting

Feature selection generally aims to reduce the error rate of a given classifier. In our case, we will be dealing with imbalanced datasets. There is no exact threshold when a dataset can be considered imbalanced, however, to make the problem definition more precise, we will be defining the subset of datasets at which we are aiming.

Definition 4 (Class imbalance)**.** Given a d-dimensional dataset D with a set of feature vectors $X = \{x_1, x_2, \ldots, x_d\}$ and a class vector Y. There exists a frequency distribution f_Y which contains the frequency of all distinct class values in Y. We define D to be **imbalanced** if

$$gini(f_Y) \geq 0.3.$$

The Gini coefficient is a general measure of inequality in a frequency distribution [13]. It is also defined for probability distributions. Therefore, we can give as examples a two-class problem with an 90:10 distribution or a three-class problem with a 50:45:5 distribution, both reaching this threshold. Any dataset with the same or greater inequality will be considered imbalanced in this thesis.

On these datasets, we will select features in order to train a classifier. To quantify the measure we are aiming to optimize, we are using a different accuracy measure defined for each class individually.

Definition 5 (Recall)**.** Given a Classifier K and a dataset D. For every object o in D there is a predicted class $C_p(o)$ and a real class $C_r(o)$. For every class c in the set of possible classes C, there is a set of real positives $P_c = \{o \in O \mid C_r(o) = c\}$ and a set of predicted true positives $TP_c \subseteq CP_c$ with $TP_c = \{o \in O \mid C_r(o) = c \wedge C_p(o) = c\}$. Recall is then defined as [14]

$$recall(K, c) = \frac{|TP_c|}{|CP_c|}.$$

Definition 6 (Precision)**.** Given a Classifier K and a dataset D. For every object $o \in D$ there is a predicted class $C_p(o)$ and a real class $C_r(o)$. For every class $c \in C$, there is a set of predicted positives $PP_c = \{o \in O \mid C_p(o) = c\}$ and a set of predicted true positives $TP_c \subseteq PP_c$ with $TP_c = \{o \in O \mid C_r(o) = c \wedge C_p(o) = c\}$. Precision is then defined as [14]

$$precision(K, c) = \frac{|TP_c|}{|PP_c|}.$$

Definition 7 (F_1 score)**.** Given the recall and precision values for every class $c \in C$, the F_1 measure is the harmonic mean of these two values. It is therefore defined as [14]

$$F_1(K, c) = 2 \cdot \frac{1}{\frac{1}{recall(K,c)} + \frac{1}{precision(K,c)}} = 2 \cdot \frac{precision(K, c) \cdot recall(K, c)}{precision(K, c) + recall(K, c)}.$$

This score gives a fairly holistic overview of the accuracy for each class. To optimize a specific value, we will average this score over all classes. Since we want to consider the accuracy for each class of the same importance, regardless of its frequency, we will be looking at the so-called macro-averaged F_1 score. This is defined as the simple mean, i.e.

$$M_C(K) = \frac{1}{|C|} \sum_{c \in C} F_1(K, c).$$

This is a measure that does not bias toward the most frequent class, and is therefore well-suited for evaluation on imbalanced datasets [15].
With this function we can now define an objective for our feature selection algorithm. Out of all feature subsets $F_i \subseteq X$, we aim to find the one where

$$\max_{\forall F_i \subseteq X} M_C(K).$$

In other words, our problem can be formally defined as finding a feature selection algorithm that will maintain a high accuracy in every class despite significant imbalance in the dataset.

3 Related Work

We present the existing literature on dimensionality reduction methods targeting class imbalance. We will collect and compare different feature selection algorithms to our new method. Despite their slightly different outcome, we introduce some feature extraction algorithms and analyze how their approaches are applicable to feature selection.

Class imbalance in classification first emerged as an area of research more than a decade ago [16], and has been an important consideration for data analysis ever since. The idea of utilizing dimensionality reduction to mitigate class imbalance arose soon after [8], prompting a substantive amount of research into this topic. Many publications have evaluated the general ability of feature selection algorithms to mitigate class imbalance, although the more specific case of adjusting these algorithms to further enhance this ability has not been paid attention to quite as much.

One of the earliest feature selection approaches to imbalanced datasets has been described by Zheng et al. [17], introducing a framework that splits a multilabel problem [18], as is common in a text categorization context, into multiple binary classification (and feature selection) problems and then wraps various metrics. Based on the observation that there are features indicating membership of a class (positive features) as well as features indicating non-membership (negative), one-sided metrics are used to generate two distinct feature sets and later joined. The authors theorize that negative features are scored lower than positive features in imbalanced datasets, thus leading to their underestimation by two-sided metrics. Picking the k best negative features independent of positive ones aims to correct this behavior.
This approach is well-defined for multilabel problems, as in this context it is common to use different feature subsets for each label. If we were to extend this definition to multiclass problems, we would have to define an algorithm that combines the feature subsets of each binary classification problem in a satisfactory manner. However, the approach in its original form is already well-defined for binary classification, so it would be applicable for datasets of this type.

A well-known method for feature selection on imbalanced datasets is Feature Assessment by Sliding Thresholds (FAST) first described by Chen et al. [19]. This algorithm works closely with a given classifier and varies the decision thresholds of each feature, calculating the true positive rate and true negative rate along the way. This can be used to generate a ROC curve, and analyze the area under the curve to evaluate the feature. The resulting score between 0.5 and 1 can be interpreted as the predictive power of the feature, regardless of the final decision threshold, and is used to create a feature ranking. The area under the ROC curve

(also known as AUC) is commonly used to evaluate classification algorithms [20], as it provides a good performance measure unaffected by imbalance.
Because it is explicitly defined for a single decision threshold, FAST works only on binary classification problems in its described form. It would have to be extended to slide multiple thresholds, which would significantly increase its runtime by requiring an exponential amount of classifier iterations. Another approach would be to combine the individual feature subsets, but such a method has not been described yet.
Furthermore, FAST is closely dependent on the used classification method. This leaves it somewhat susceptible to classifier-specific biases and errors, and feature subsets selected by it might not be universal. Whether this is a downside depends on the context, as feature selection is often directly followed by classifying the data anyway.

An algorithm from a different context but nonetheless useful for comparison is Pairwise Cost in Semisupervised Discriminant Analysis (PCSDA), introduced by Wan et al. [21]. This method concerns itself with feature extraction on data used for semi-supervised learning, i.e. partially unlabeled data. This is common in face recognition settings [22], a major focus for the contribution of Wan et al. PCSDA tries to find a projection matrix that minimizes Bayesian risk for each class pair while reducing dimensionality. It does so by combining Linear Discriminant Analysis [23] with a weighting function that considers class imbalance as well as an adjustable cost matrix, one of the major advantages of this approach. In fields where reduction of necessary measurements is a central motivation for dimensionality reduction, feature extraction is not a viable alternative to feature selection. While PCSDA could potentially be generalized to apply to fully labeled data, it is too specific to be transferred to feature selection without a significant rethinking of its methods.

Our contribution, wRaR, is based on an existing feature selection method, RaR [24]. As such, it inherits many of its benefits and is thus particularly well-suited for feature selection on mixed datasets. In addition, it takes both relevance and redundancy of features into account.
As wRaR is based on weighting class-specific measurements and in contrast to the mentioned feature selection methods, it can be supplied with additional weighting instructions that will be applied on top of imbalance-related weighting. This is particularly useful in applications where specific classes have a high cost attached, such as environmental conservation [25], network intrusion detection [26], or medical diagnosis [27]. Due to the operating principles of its weighting, wRaR is equivalent to RaR in binary classification problems as explained in section 4.2.2, therefore it is primarily appropriate for imbalanced datasets with more than two classes. Using it on other types of datasets would not be different from applying RaR.

Algorithms	Outputs feature subset	Binary classification	Multiclass classification	Adjustable cost
[17]	✓	✓	✗	✗
FAST[19]	✓	✓	✗	✗
PCSDA[21]	✗	✓	✓	✓
wRaR	✓	✗	✓	✓

Table 1: Comparison of dimensionality reduction algorithms targeting class imbalance

The differences between the mentioned dimensionality reduction methods are summarized in section 3. PCSDA works only on partially labeled data, and is thus not directly comparable to wRaR. In addition, PCSDA is a feature extraction method and thus does not output a feature subset, so the use cases are different as well. The two other feature selection methods work on binary classification problems only, while wRaR tackles class imbalance only in multiclass problems. This makes it hard to compare these approaches.
In contrast to the other approaches, wRaR and PCSDA are both based on weighting, therefore an adjustable cost can be supplied to them.

4 Weighted Relevance and Redundancy Scoring

In this section we will introduce and explain our contribution. Our algorithm is based on Relevance and Redundancy Scoring [24], which we will also outline.

4.1 Relevance and Redundancy Scoring

The underlying feature selection algorithm (RaR) is based on the definition of relevance and redundancy of different features. The relevance of feature subset $S \subseteq F$ toward a target Y is defined as

$$rel_D(S) = D(P(Y \mid S) \parallel P(Y)) \tag{1}$$

where D is a divergence function. Likewise, the redundancy between a subset $S \subseteq F$ and an individual feature f is defined as

$$red_D(f, S) = D(P(f \mid S) \parallel P(f)). \tag{2}$$

The objective now is to select a subset of features optimal for predicting the target, therefore we need to select features that are very relevant, yet only minimally redundant to each other. RaR thus follows the general approach of maximizing relevances and minimizing redundancy used in many feature selection algorithms, e.g. mRmR [28]. Since it is computationally expensive to calculate the relevance or redundancy for each possible combination, it is advisable to employ a heuristic estimating this value. The measure is based on a divergence function of conditional and marginal probability distributions and as such, one of its essential properties is information monotonicity. Therefore, for a given feature subset $S \subseteq F$ and a target y, it holds true for these measures rel and red that

$$\forall S' \subseteq S : rel_D(S') \leq rel_D(S) \tag{3}$$

$$\forall S' \subseteq S : red_D(S', f) \leq red_D(S, f). \tag{4}$$

As a result, we can calculate the redundancy for random subsets, as well as the redundancy between random features and random subsets, and use this relation to extrapolate desired values. Relevances can be decomposed into estimates for each individual feature, while the calculated redundancies can be used to estimate the scores when iteratively selecting features for a ranking. Furthermore, by using subsets instead of feature-to-target relevances, RaR is able to recognize multi-feature correlations. A feature might have a low relevance on its own, yet have a high relevance toward the target in the context of another feature, resulting in a relevance score for these two features that is higher than their individual relevance scores.

4.2 General idea

Relevance and redundancy scores are based on a divergence function. In order to estimate the divergence for $p(Y \mid S)$, random slices conditioned on subset S and equal in size are picked, and the resulting divergences averaged. This is based on an an algorithm described by Keller et al., originally used to find High Contrast Subspaces for density-based outlier ranking [29]. Since we are comparing conditional distributions conditioned on S with regards to Y in comparison to its marginal distribution, we can evaluate how much S affects specific classes. In the context of imbalanced classes, we can influence feature selection to prefer features that would lead to a more equally distributed accuracy. Given a score $score_c$ of how much a feature f benefits a class c for every class $c \in C$, we can reward features with a high $score_{c_1}$ for a very infrequent class c_1 and prioritize such features over ones with a high score in a very frequent class. This can be accomplished by applying a weighting that is generated based on the existing imbalance in the dataset, and then adjusting the relevance score.
Given a target vector Y and a set of classes C. We can define our counterweight for all $c \in C$ as

$$w(c) = \frac{1}{p(Y = c)}. \tag{5}$$

This is a direct inversion of the class distribution, which will result in a higher weight for infrequent classes and a lower weight for frequent classes. It is strictly inversely proportional to the share of a class within the dataset.

4.2.1 Class-wise statistical dependence

If we observe the conditional probabilities of all slices for a given combination of subset S and our target Y, we can make out certain properties. In multiclass problems, some classes could stay at a consistent probability over all slices, while only as few as 2 have a change in probability and as such, are the main contributors to the final divergence value. If this is the case for all possible slices, i.e. slicing does not affect the occurrence of a class c in any way, we can define Z as an event with $p(Z) = p(Y = c)$ and say that S as a random variable and Z are statistically independent, since $p(Z \mid S = s) = p(Z)$ for all outcomes s of S. Note that this is different from the fact that $p(Z \mid S) = p(Z)$, which is already given by the law of total probability. In simpler terms, this means that we are looking at the average change between the conditional distribution and the marginal distribution, not the change between the averaged conditional probability and its marginal one, which is evidently equal to zero.
A statistical independence of this event and the subset as random variable is extremely unlikely. If that were the case, each outcome of S would have to be duplicated enough to reproduce the distribution of $p(Z)$ and its complement $p(\overline{Z})$

and thus would have to have a count of distinct values orders of magnitudes lower than its overall size. This is a simple side effect of using finite data to simulate a random variable, resulting in statistical independence being much easier to prove for categorical features. While this ideal state is an interesting thought experiment, our primary aim is to estimate how statistically dependent S and Z are. However, it is useful to see the limits of our slicing. Theoretically, it is easy disprove this statistical independence for almost any dataset by choosing a non-contiguous slice, e.g. a slice consisting of every value that is a multiple of 5. This is something RaR and, by extension, HiCS do not do, at least not for non-categorical features. The estimate of statistical dependence calculated by these algorithms is based only on random contiguous slices, therefore it misses some potential, although unorthodox, compositions of feature data that might show a strong predictive value. As classifiers such as Decision Trees do not commonly utilize non-contiguous feature splits either, this is not a relevant disadvantage, but it is important to keep in mind that due to these edge cases, contrast may not be a good estimate of mutual information in all cases.

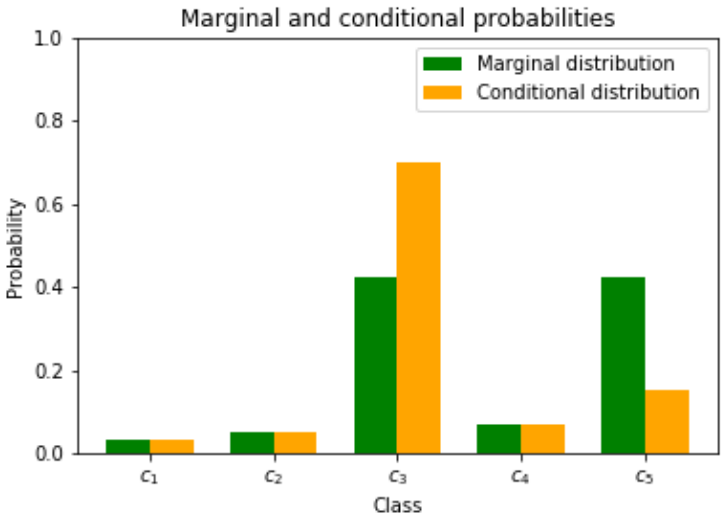


Figure 1: A typical slice for a dataset with five classes

To illustrate low statistical dependence of specific classes, observe the probabilities in fig. 1. In the marginal distribution, c_3 and c_5 are fairly frequent while c_1, c_2, and c_4 are very rare classes, i.e. the dataset is imbalanced. For an example slice, the conditional probability distribution diverges significantly for c_3 and c_5, but not at all for the other classes. In other words, there is no information in this particular slice that would help us discriminate classes more precisely concerning the infrequent classes c_1, c_2, or c_4. If similar behavior repeats with every other slice we test, it is reasonable to assume this particular subset or feature does

not contain a lot of information about the infrequent classes. As we want to reward features with more predictive value on classes that would otherwise suffer immensely from algorithms not taking class imbalance into account, the particular feature or subset on which the distribution in fig. 1 is conditioned should receive a lower relevance score. Doing so establishes a bias toward a feature subset that would result in a more homogeneous distribution of accuracy among the classes, one of the primary objectives of this thesis.

4.2.2 Class Divergence

As laid out in section 4.2.1, we can define $Y = c$ as a separate event instead of using Y as random variable, and thus analyze the statistical dependence of $Y = c$ and our subset. In a similar fashion as with the relevance between Y and S, we can define the relevance of S toward any class $c \in C$. Given that M is a set of slices conditioned based on S and $m \in M$ a slice from this set, we define a function

$$\delta_C(Y, c, m) = D(P(Y = c \mid m) \parallel P(Y = c)) \tag{6}$$

The mean of this function over all slices $m \in M$ will be the class relevance

$$rel_C(S, Y, c) = \frac{1}{|M|} \sum_{m \in M} \delta_C(Y, c, m). \tag{7}$$

Now, to elaborate on the meaning of eq. (6) and eq. (7). With $\delta_C(Y, c, m)$, we calculate the divergence for a conditional probability distribution consisting of $p(Y = c)$ and $p(\overline{Y = c})$. As an example, a slice as it appears in fig. 1 has no value for class c_1, because $p(Y = c_1)$ does not change and neither does its complement. There is a significant change in probability for class c_3 and c_5, but that does not affect the overall probability of $p(\overline{Y = c})$. Regardless of what divergence function is used, as part of its definition it satisfies the property that $D(P \parallel Q) = 0$ iff $P = Q$, therefore our class-wise divergence function for that particular slice will equal to zero as well. After calculating the mean of each class-wise relevance over all slices, we have a score $rel_C(S, Y, c)$ for all classes $c \in C$. Since we want to reward high divergences in infrequent classes, we compute a scoring by combining these values with the weighting from eq. (5) in a weighted average. We define our scoring function as

$$score(S, Y) = \frac{1}{\sum\limits_{c \in C} w(c)} \cdot \sum_{c \in C} w(c) \cdot rel_C(S, Y, c). \tag{8}$$

This score replaces the relevance score $rel_D(S)$ defined in eq. (1).
A very important property to consider is the fact that our new score equals the existing divergence measure if $|C| \leq 2$. For such an amount of classes, $P(Y) =$

$P(Y = c)$ and $P(Y \mid m) = P(Y = c \mid m)$ for all $c \in C$; the probability distributions are the same. Since $\delta_C(Y, c, m)$ will then be the same for every class, we are calculating the same $rel_C(S, Y, c)$. The weighted mean of two identical values equals the same value, so we can say that, given $|C| < 2$, $score(S, Y) = rel_D(S, Y)$. Therefore while our new measure will work on any dataset, class imbalance will only be considered in **multiclass** problems. This corresponds to the notion that, given a probability distribution such as in fig. 1, we could not measure the absence of change for an individual class as proposed, because every change in one probability must strictly affect the other.

To show that the calculated score is still valid within the context of RaR, we must prove that our weighted mean is still a valid divergence function. While computing a weighted mean using rel_C at the very end is useful to have a class-dependent relevance value, we can simplify it to have a score for each slice. Since rel_C is a sum of $\delta_C(Y, c, m)$, we can apply the weight to each slice individually. Over all classes $c \in C$, we can also divide by the sum of all weights on a slice-by-slice basis. Therefore we can define a class-weighted divergence

$$D_C(P(Y \mid m) \parallel P(Y)) = \frac{1}{\sum\limits_{c \in C} w(c)} \sum_{c \in C} w(c) \cdot \delta_C(Y, c, m). \tag{9}$$

This function fulfills all requirements to be a divergence function, as we will show below. As with rel_C before, we still divide by $|M|$ to compute the mean as we simply moved the order of operations. Therefore we can say that

$$score(S, Y) = \frac{1}{|M|} \sum_{m \in M} D_C(P(Y \mid m) \parallel P(Y)). \tag{10}$$

In order to be a valid divergence function, D_C must fulfill the properties outlined in definition 3. Now, for explanation purposes D_C and δ_C have been defined with a specific condition in mind, but since δ_C simply computes a divergence of two probability distribution using its parameters, it could easily be written as $\delta_C, m(P \parallel Q)$ to conform to a divergence function's typical form of $D(P \parallel Q)$.
D_C is a composition of divergence functions using a weighted mean. We know that $\forall c \in C : w(c) = \frac{1}{P(Y=c)} > 0$, therefore each summand must be greater than or equal to zero, and thus the quotient as well. Since the weights can never be zero, D_C will be zero if and only if δ_C is zero. δ_C is defined as a divergence function, for which this property must already be true, therefore it must be true for D_C as well. Since D_C fulfills all properties, it is a valid divergence function and as such a valid component of RaR. This specialized version of RaR will be called wRaR for the rest of this thesis.

4.3 Pseudocode

After we have described the mathematical basis in section 4.2.2, we would like to illustrate how our new weighted approach (wRaR) works using pseudocode. This is a specialization and as such, represents part of the process of the estimation of feature relevance. It is the computation of $rel(S_i)$, this value is then used to define constraints for individual feature relevances (c.f. [24]).

```
function RELEVANCE(S, Y, weights)
    M ← generate random slices of P(Y | S)
    C ← unique values of Y
    scores ← empty map of lists
    for s ∈ S do
        for c ∈ C do
            divergence_c ← D(P(Y = c | s) ‖ P(Y = c)) ▷ eq. (6)
            scores[c].push(divergence_c)
        end for
    end for
    averages ← empty map
    for c ∈ C do
        averages[c] ← mean(scores[c])
    end for
    relevance ← Σ_{c∈C} weights[c] · averages[c]
    return relevance
end function
```

4.4 Extension to Constraint Solving

The relevance of S and Y calculated by our weighted approach is used by RaR to create constraints for each feature in S. This is based on the fact that, given that $r(f)$ is the individual relevance of feature f, we can say that [24]

$$rel(S) \leq \sum_{f_i \in S} r(f_i). \tag{11}$$

These constraints are defined for each random subspace S_i. After multiple iterations, we should have multiple constraints for each feature, allowing us to deduce $r(f)$. These are merely lower bounds, so we aim to minimize the individual relevances. This objective function estimating $r(f)$ for all $f \in \mathcal{F}$ given M iterations is defined as [24]

$$\min_{r(f)} \Big[\sum_{f \in \mathcal{F}} r(f) + \sum_{f \in \mathcal{F}} (r(f) - \mu)^2 \Big] \text{ s.t. } rel(S_i) \leq \sum_{f \in S_i} r(f) \mid i = 1, \ldots, M, \tag{12}$$

where μ is the mean of all relevances, i.e. $\mu = \frac{1}{|\mathcal{F}|} \sum_{f \in \mathcal{F}} r(f)$.
Since wRaR is calculating a relevance value between subset S and each individual class c as laid out in eq. (7), it would make sense to apply this optimization function to each individual class relevance before taking the weighted mean. While over all subsets $S_i \in \mathcal{F}$, the features most beneficial for our imbalanced dataset should receive a boosted relevance, the heuristic will always only be able to measure a limited amount of subsets and unintended inaccuracies might occur. Hence it would be advisable to create a slightly different constraint to optimize on, resulting in

$$\min_{r_c(f)} \Big[\sum_{f \in \mathcal{F}} r_c(f) + \sum_{f \in \mathcal{F}} (r_c(f) - \mu_c)^2 \Big] \text{ s.t. } rel_C(S_i, Y, c) \leq \sum_{f \in S_i} r_c(f) \mid i = 1, \ldots, M, \tag{13}$$

to be run separately for each class $c \in C$. After estimating a value $r_c(f)$ for all features and all classes, we take the weighted mean of these values to compute a final $r(f)$.

4.5 Relationship to other divergences

Although not essential for our algorithm, it could be interesting to investigate the relationship between our new measure (eq. (6)) and the one used in RaR or, more broadly, the difference between the divergence on a random variable Y and the divergence on event $Y = c$. Given that $\delta^\star_C$ is defined as a specialization of δ_C that uses the Kullback-Leibler divergence. We can say that

$$D_{KL}(P(Y \mid m) \parallel P(Y)) = \sum_{c \in C} \delta^\star_C(Y, c, m) - Pr(\overline{Y = c} \mid m) \cdot \log \frac{Pr(\overline{Y = c} \mid m)}{Pr(\overline{Y = c})}. \tag{14}$$

As they are not strictly related to wRaR, we have outlined this and other observations in the appendix, section 8.1.

5 Implementation Details

wRaR estimates relevance and redundancy for feature selection by averaging random subslices of random feature subsets. This is a heuristic over an enormous problem space and as such, the approach has certain technical limitations. There are a number of parameters regarding subspace search that wRaR inherits from RaR. These have not changed and can be found in [24].

The new additions with wRaR have some parameters as well, these affect the weighting of each class. An essential part of wRaR is the fact that one can supply a custom weighting function. In its current implementation, a list of weights for each class can be passed to the algorithm, which will be multiplied with the existing weights, i.e. for input ω the resulting function w' will be

$$w'(c) = \frac{\omega(c)}{p(Y = c)}. \tag{15}$$

This represents an adjustable bias toward classes with a higher $\omega(c)$, as high relevance toward these classes will result in an increased weighted mean. This makes sense when a particular class has a higher cost than others, and requires some fine-tuning. This also means that wRaR may be a suitable feature selection algorithm even when a multiclass dataset is not imbalanced.
Another parameter is the weight exponent, i.e. x where

$$w'(c) = w(c)^x = \frac{1}{p(Y = c)^x}. \tag{16}$$

It is a simple shortcut over supplying a specific ω of the same effect. The exponent significantly increases the weight of an infrequent class as share of the sum of all weights, providing a much higher counterweight.

wRaR has been fully implemented in Python. In the reasoning phase of RaR, it utilizes the Gurobi Optimizer [30]. The code is made available online under the MIT License[1].

[1] `https://github.com/KDD-OpenSource/wRaR`

6 Experiments

We will evaluate wRaR on a variety of datasets. As related algorithms only work on binary class datasets or are otherwise not applicable in this context, we will focus on evaluating how wRaR performs compared to RaR [24]. RaR as a general feature selection algorithm already outperforms many other methods, so it serves as a good baseline. RaR and wRaR have been tested with the same parameters, that is, a maximum subset size of 5, 200 Monte Carlo iterations, and an α value of 0.1. To ensure a representative result, we employ threefold cross-validation and have tested the generated feature ranking with multiple different classifiers. The results are compiled by taking the k best features of each ranking, training a classifier on these, and calculating the macro-averaged F_1 score described in definition 7.
wRaR has been implemented in Python based on the findings laid out in section 4.2 and section 4.4. As a consequence, the classifiers tested are used as they are implemented in sklearn [31], with their default parameters.

6.1 Synthetic datasets

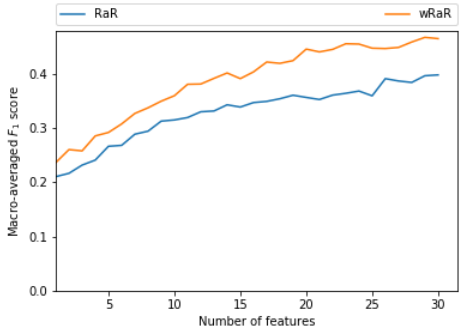
We have generated multiple synthetic datasets with a differing amount of classes and differing distributions given a gini coefficient as laid out in definition 4. To generate synthetic datasets, we have employed a feature generation approach based on NIPS [32], itself based on [33]. We only generate continuous features for evaluation, although categorization through discretization would also be possible. Before we present the results, let us first describe each synthetic dataset.

- *synthetic1*: This dataset has 5,000 instances and 100 features. It has been generated so that 30 features are independent, of which 20 have actual weight for the generated target label, while 70 features are linear combinations of the rest. It has 5 classes and a gini coefficient of approximately 0.465, and is thus fairly imbalanced.

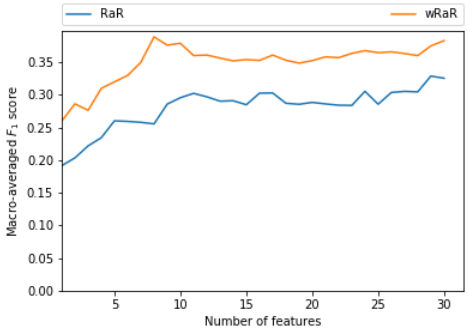
- *synthetic2*: This dataset also has 5,000 instances and 100 features. 70 of these are independent, all of which have weight, and another 30 are linear combinations. It has a lower gini coefficient of 0.37 and 5 classes.

- *synthetic3*: This dataset has only 80 features, 40 of them independent, 30 of which have weight, and 40 of them dependent. It has been generated to have a much higher imbalance with a gini coefficient of 0.61 and 5 classes.

- *synthetic4*: This dataset has been generated with 10,000 instances and a higher number of classes, 10, and a moderately high gini coefficient of 0.44. Apart from that, it has the same properties as *synthetic2*.



(a) Macro-averaged F_1 score of Naive Bayesian classifier



(b) Macro-averaged F_1 score of 1-Nearest Neighbor classifier

Figure 2: Scores for the *synthetic1* dataset on two classifiers

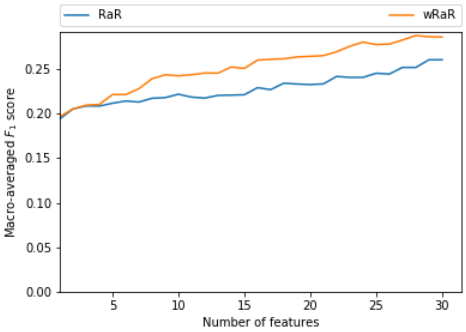


Figure 3: Score of Naive Bayesian classifier on *synthetic2*

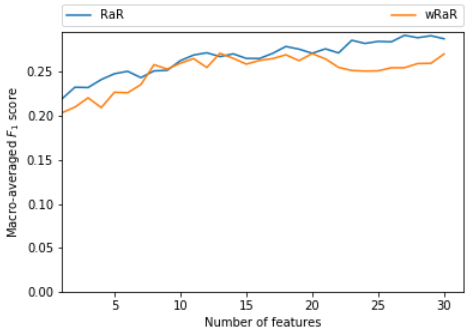
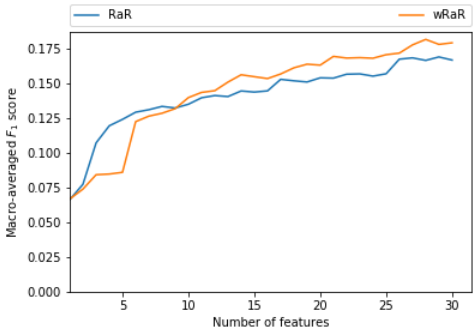
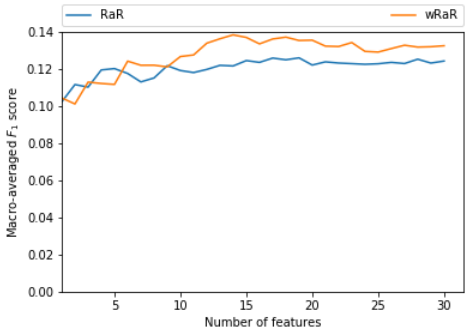


Figure 4: Score of 1-Nearest Neighbor classifier on *synthetic3*



(a) Macro-averaged F_1 score of Naive Bayesian classifier



(b) Macro-averaged F_1 score of 1-Nearest Neighbor classifier

Figure 5: Scores for the *synthetic4* dataset on two classifiers

As we can see in fig. 2, wRaR's feature ranking seems to outperform RaR's on *synthetic1*, a very imbalanced dataset, for every k best features. On *synthetic2*, a slightly less imbalanced dataset, wRaR outperforms by a smaller margin, while an extremely imbalanced dataset seems to lead to no discernible difference. The result on *synthetic3* might be an issue of too few samples in the most infrequent class, particularly when cross-validating. Nevertheless, while there is a general improvement, we consider *synthetic1* to be an outlier.
The performance on a dataset with 10 classes such as *synthetic4* appears to be not significantly better, but still with a decent margin.

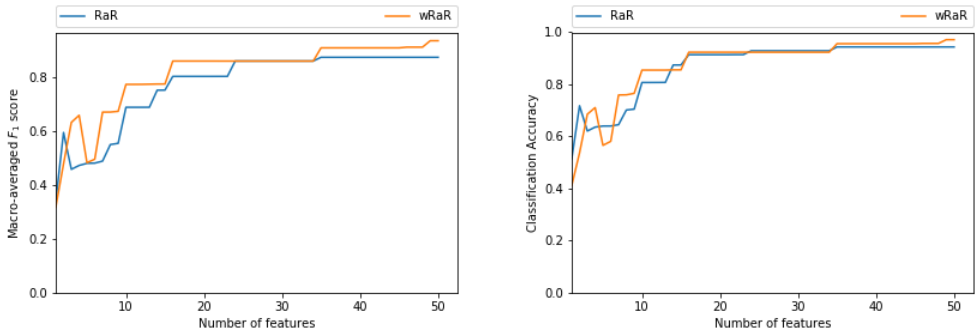
6.2 Real-world datasets

To test wRaR on real-world data, we are using multiple datasets from the UCI Machine Learning repository [34]. As we want to evaluate the performance on imbalanced datasets, we imbalance the given datasets artificially by removing samples. We pay special attention in this step in order to avoid discarding so many samples that classification is no longer sensibly possible. The datasets are listed here with their individual adjustments to achieve imbalance.

- Gas Sensor Array Drift (*gassensor*) [35]: This dataset has 6 classes, ca. 14,000 instances, 128 features, and was downsampled to achieve a gini coefficient of 0.38 with approximately 8,000 instances.

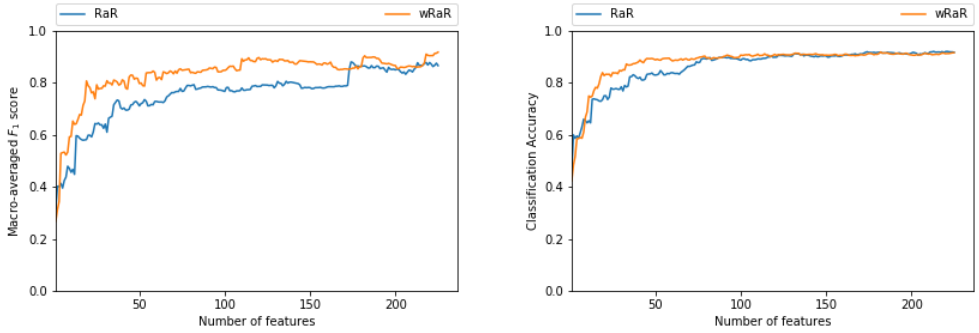
- ISOLET (*isolet*): This dataset originally had 26 classes with 240 samples each. We consider only 10 classes, and have downsampled the set resulting in a gini coefficient of 0.33. Note that, since *isolet* has an extremely high number of features of 616, the number of Monte Carlo iterations for both algorithms was increased to 500.

- Covertype (*covtype*): This dataset has more than 500,000 instances, therefore we will use it to test our runtime.



(a) Macro-averaged F_1 score (b) Accuracy score

Figure 6: Different scores of 5-Nearest Neighbors classifier for *gassensor*



(a) Macro-averaged F_1 score (b) Accuracy score

Figure 7: Different scores of 5-Nearest Neighbors classifier for *isolet*

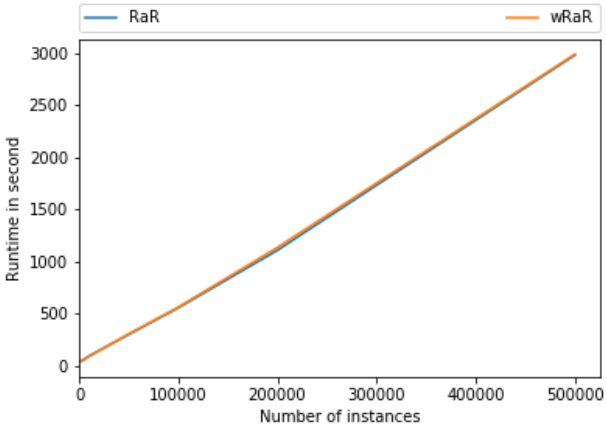


Figure 8: Runtime with growing number of instances

As we can see in fig. 6, wRaR outperforms RaR on *gassensor*. To illustrate the algorithm's different selection strategy, we have put the macro-averaged F_1 score in fig. 7a next to classification accuracy as seen in fig. 7b. While wRaR performed slightly better overall in this example, they are almost identical in terms of classification accuracy, whereas there is a large margin when looking at the F_1 score. This is because wRaR has preferred features that will benefit small classes more, boosting the F_1 score, while accuracy became secondary.
The same effect can be observed on *isolet*, as seen in fig. 7. There seems to be a slight bias toward wRaR, so we assume that we need to average more rankings to get a definitive statistic on *isolet*, but there is a clear trend toward an increased macro-averaged F_1 score when using wRaR.
As a sidenote, while *isolet* is not the best example, the benefit of feature selection in general is also visible when comparing to the F_1 score over all features. A 5-Nearesxt Neighbors classifier trained on all 616 features results in a macro-averaged F_1 score of approximately 0.88, the same classifier trained on the k best feature according to wRaR already reaches 0.91 at 110 features.

We have also measured the runtime of wRaR and RaR on *covtype*, a very large dataset. As we can see in fig. 8, adding weighting does not significantly impact runtime, i.e. it grows with the amount of instances in the same manner as RaR. We have tested them with 1,000, 5,000, 10,000, 50,000, 100,000, 200,000 and 500,000 instances, and wRaR ran on average 2-3% longer than RaR. This is a negligible amount and likely due to the additional optimizer systems, a step that is very fast in any case.

7 Conclusions

In this thesis we have presented a novel feature selection approach for imbalanced datasets. Owing to its foundation in RaR it is applicable to large datasets as well as ones with mixed features, and takes multivariate correlations into account. In addition, we have seen in our experiments that wRaR tends to outperform RaR on imbalanced datasets. Thus, we can say that wRaR can serve as a good feature selection algorithms in these cases. To be more exact, given that a dataset is imbalanced and has enough features to warrant dimensionality reduction, it is a good alternative when the target is not binary. Benefits tend to arise at a gini coefficient greater than 0.30. The upper limit where we can observe a clear benefit seems to be approximately 0.60. However, this might be a threshold above which our tested datasets did not have enough samples, so we intend to investigate and determine these limits more precisely in future work.
A clear disadvantage of wRaR is the fact that it is confined to multiclass problems. Due to this being rooted in its mathematical basis it is unlikely that this can be changed easily, so this will remain its primary application.

7.1 Future Work

Apart from investigating the algorithm's limits more closely, there are a couple of things to consider for future work.

To be able to better evaluate the effectiveness of wRaR, future work could modify related algorithms such as FAST and the one outlined in [17] to accept multiclass classification problems. Another field of interest would be how wRaR could incorporate more sophisticated cost functions as input, such as a separate cost for the false positives and false negatives of each class.

Finally, building on the relationship between class-wise divergence and general divergence described in section 4.5, future work could further explore these relationships and if there are rules that govern them that have not been identified yet.

8 Appendix

8.1 Relationship to Kullback-Leibler Divergence

If we want to derive the weighted function in eq. (8) from an existing divergence function, we can define $\delta_C^\star$ as a specialization of δ_C that uses the Kullback-Leibler divergence, and show their relationship. We can say that

$$D_{KL}(P(Y \mid m) \parallel P(Y)) = \sum_{c \in C} \delta_C^\star(Y, c, m) - Pr(\overline{Y = c} \mid m) \cdot \log \frac{Pr(\overline{Y = c} \mid m)}{Pr(\overline{Y = c})}. \tag{17}$$

As $\delta_C^\star$ is defined as the Kullback-Leibler divergence of $P(Y = c \mid m)$ and $P(Y = c)$, we now have a simple relationship between this binary divergence and the normal Kullback-Leibler divergence. In essence, we are subtracting the summand concerning the converse probability from each binary divergence, resulting in a sum of all log differences of the probabilities, the definition of Kullback-Leibler divergence. The relationship is fairly straightforward, nevertheless, proof of it can be found below in eq. (22).

A much more interesting relationship however can be seen on balanced datasets. We can simplify $D_C^\star$, which is D_C from eq. (9) but using $\delta_C^\star$, in a way that might not be obvious, and could be relevant for other works. Note that since our dataset is balanced, $q_c = \frac{1}{|C|}$ for all classes. For better readibility, we define $l = |C|$, and event $Y = c$ equal to Z_c.

$$\begin{aligned}
D_C^\star(P(Y \mid m) \parallel P(Y)) &= \frac{1}{\sum_{c \in C} w(c)} \sum_{c \in C} w(c) \cdot \delta_C^\star(Y, c, m) \\
&= \frac{1}{l^2} \sum_{c \in C} l \cdot \delta_C^\star(Y, c, m) \\
&= \frac{1}{l} \sum_{c \in C} D_{KL}(P(Z_c \mid m) \parallel P(Z_c)) \\
&= \frac{1}{l} \sum_{c \in C} p_c \log(p_c \cdot l) + (1 - p_c) \cdot \log((1 - p_c) \cdot l) \\
&= \frac{1}{l}\Big(D_{KL}(P(Y \mid m) \parallel P(Y)) + \sum_{c \in C} (1 - p_c) \cdot \log l + \sum_{c \in C} (1 - p_c) \cdot \log(1 - p_c)\Big) \\
&= \frac{1}{l}\Big(D_{KL}(P(Y \mid m) \parallel P(Y)) + (l - 1) \log l + \sum_{c \in C} \log(1 - p_c)^{(1 - p_c)}\Big) \\
&= \frac{1}{l}\Big(D_{KL}(P(Y \mid m) \parallel P(Y)) + (l - 1) \log l + \log(\prod_{c \in C} (1 - p_c)^{(1 - p_c)})\Big)
\end{aligned} \tag{18}$$

This can also be put as

$$D_{KL}(P(Y \mid m) \parallel P(Y)) = \\ l \cdot D_C^\star(P(Y \mid m) \parallel P(Y)) - \log l^{l-1} - \log\left(\prod_{c \in C}(1-p_c)^{(1-p_c)}\right) \quad (19)$$

As we can see, the relationship between the sum of Kullback-Leibler divergences for each class as event and the divergence of the entire distribution is much simpler. There is a constant added that is merely dependent on the number of classes. It would be something to consider for future work to find out if the last summand can be further simplified if averaged over all slices, i.e. transforming the term

$$\frac{1}{|M|}\sum_{m \in M} \log\left(\prod_{c \in C}(1-p_c)^{(1-p_c)}\right). \quad (20)$$

A potential approach might be using the fact that the expectancy of p_c over all slices equals q_c for a sufficiently accurate slicing process. If a better term is found, we could stipulate a simpler rule for the relationship of D_{KL} and $D_C^\star$ conditioned on an arbitrary subset S, as opposed to conditioned on m. For now the relationship can be described as

$$D_{KL}(P(Y \mid S \parallel P(Y)) = \\ \frac{1}{|M|}\sum_{m \in M}\left(l \cdot D_C^\star(P(Y \mid m) \parallel P(Y)) - \log\left(\prod_{c \in C}(1-p_c)^{(1-p_c)}\right)\right) - \log l^{l-1}. \quad (21)$$

Note that, since $\log l^{l-1}$ is constant over all slices, we can even add it at the very end, when we are averaging slices, to get the same result.

As an addendum, let us quickly prove the equivalence in eq. (17). In the following, individual probabilities of $P(Y \mid m)$ and $P(Y)$ for class c will be given as p_c and q_c, respectively. Given a slice $m \in M$ where M is a set of slices picked from S. The proof is as follows

$$\begin{aligned}
&\sum_{c \in C} D_{KL}(P(Y=c \mid m) \parallel P(Y=c)) - Pr(\overline{Y=c} \mid m) \cdot \log \frac{Pr(\overline{Y=c} \mid m)}{Pr(\overline{Y=c})} \\
&\quad = \sum_{c \in C} D_{KL}(P(Y=c \mid m) \parallel P(Y=c)) - (1-p_c) \cdot \log \frac{1-p_c}{1-q_c} \\
&\quad = \sum_{c \in C} p_c \cdot \log \frac{p_c}{q_c} + (1-p_c) \cdot \log \frac{1-p_c}{1-q_c} - (1-p_c) \cdot \log \frac{1-p_c}{1-q_c} \\
&\quad = \sum_{c \in C} p_c \cdot \log \frac{p_c}{q_c} \\
&\quad = D_{KL}(P(Y \mid m) \parallel P(Y)))
\end{aligned}$$

□ (22)

References

[1] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intelligent data analysis*, vol. 6, no. 5, pp. 429–449, 2002.

[2] H. Han, W.-Y. Wang, and B.-H. Mao, "Borderline-smote: a new over-sampling method in imbalanced data sets learning," *Advances in intelligent computing*, pp. 878–887, 2005.

[3] H. He, Y. Bai, E. A. Garcia, and S. Li, "Adasyn: Adaptive synthetic sampling approach for imbalanced learning," in *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, pp. 1322–1328, IEEE, 2008.

[4] N. V. Chawla, "C4. 5 and imbalanced data sets: investigating the effect of sampling method, probabilistic estimate, and decision tree structure," in *Proceedings of the ICML*, vol. 3, 2003.

[5] G. M. Weiss, K. McCarthy, and B. Zabar, "Cost-sensitive learning vs. sampling: Which is best for handling unbalanced classes with unequal error costs?,"

[6] S. J. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P. K. Chan, "Cost-based modeling for fraud and intrusion detection: Results from the jam project," in *DARPA Information Survivability Conference and Exposition, 2000. DISCEX'00. Proceedings*, vol. 2, pp. 130–144, IEEE, 2000.

[7] Y. Zhang and Z.-H. Zhou, "Cost-sensitive face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 10, pp. 1758–1769, 2010.

[8] N. V. Chawla, N. Japkowicz, and A. Kotcz, "Special issue on learning from imbalanced data sets," *ACM Sigkdd Explorations Newsletter*, vol. 6, no. 1, pp. 1–6, 2004.

[9] R. Bellman, *Dynamic programming*. Courier Corporation, 2013.

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

[11] S. Eguchi *et al.*, "A differential geometric approach to statistical inference on

the basis of contrast functionals," *Hiroshima mathematical journal*, vol. 15, no. 2, pp. 341–391, 1985.

[12] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[13] C. Gini, "Variabilità e mutabilità," *Reprinted in Memorie di metodologica statistica (Ed. Pizetti E, Salvemini, T). Rome: Libreria Eredi Virgilio Veschi*, 1912.

[14] R. Baeza-Yates, B. Ribeiro-Neto, *et al.*, *Modern information retrieval*, vol. 463. ACM press New York, 1999.

[15] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, vol. 45, no. 4, pp. 427 – 437, 2009.

[16] N. Japkowicz, "The class imbalance problem: Significance and strategies," in *Proc. of the Int'l Conf. on Artificial Intelligence*, 2000.

[17] Z. Zheng, X. Wu, and R. Srihari, "Feature selection for text categorization on imbalanced data," *ACM Sigkdd Explorations Newsletter*, vol. 6, no. 1, pp. 80–89, 2004.

[18] G. Tsoumakas and I. Katakis, "Multi-label classification: An overview," *International Journal of Data Warehousing and Mining*, vol. 3, no. 3, 2006.

[19] X.-w. Chen and M. Wasikowski, "Fast: a roc-based feature selection metric for small samples and imbalanced data classification problems," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 124–132, ACM, 2008.

[20] A. P. Bradley, "The use of the area under the roc curve in the evaluation of machine learning algorithms," *Pattern recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.

[21] J. Wan, M. Yang, Y. Gao, and Y. Chen, "Pairwise costs in semisupervised discriminant analysis for face recognition," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 10, pp. 1569–1580, 2014.

[22] X. Zhu, "Semi-supervised learning literature survey," *Computer Science, University of Wisconsin-Madison*, vol. 2, no. 3, p. 4, 2006.

[23] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of human genetics*, vol. 7, no. 2, pp. 179–188, 1936.

[24] A. K. Shekar, T. Bocklisch, C. N. Straehle, P. I. Sánchez, and E. Müller, "Including multi-feature interactions and redundancy for feature ranking in mixed datasets," in *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2017, Macedonia, Skopje, September 18-22, 2017, Proceedings*, Lecture Notes in Computer Science, Springer, 2017.

[25] A. H. Fielding and J. F. Bell, "A review of methods for the assessment of prediction errors in conservation presence/absence models," *Environmental conservation*, vol. 24, no. 1, pp. 38–49, 1997.

[26] W. Lee, W. Fan, M. Miller, S. J. Stolfo, and E. Zadok, "Toward cost-sensitive modeling for intrusion detection and response," *Journal of computer security*, vol. 10, no. 1-2, pp. 5–22, 2002.

[27] M. Núñez, "The use of background knowledge in decision tree induction," *Machine learning*, vol. 6, no. 3, pp. 231–250, 1991.

[28] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005.

[29] F. Keller, E. Muller, and K. Bohm, "Hics: High contrast subspaces for density-based outlier ranking," in *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*, pp. 1037–1048, IEEE, 2012.

[30] I. Gurobi Optimization, "Gurobi optimizer reference manual," 2016.

[31] 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*, vol. 12, pp. 2825–2830, 2011.

[32] "Nips: Workshop on variable and feature selection," 2001.

[33] S. Perkins and J. Theiler, "Online feature selection using grafting," in *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pp. 592–599, 2003.

[34] M. Lichman, "UCI machine learning repository," 2013.

[35] A. Vergara, S. Vembu, T. Ayhan, M. A. Ryan, M. Homer, and R. Huerta, "Chemical gas sensor drift compensation using classifier ensembles," vol. s 166–167, p. 320–329, 05 2012.

German abstract

Ein häufiges Problem in der Klassifikation ist Ungleichgewicht von Klassen, ein Zustand in dem gewisse Klassen eines Datensatzes wesentlich häufiger auftreten als andere. In solchen Fällen würde das Klassifizieren aller Instanzen als die häufigste Klasse zu einer hohen Genauigkeit führen, doch solch eine Lösung ließe sich nicht als gute Klassifikation beschreiben. Mit anderen Worten sind die Kosten, eine Instanz falsch zu klassifizieren, für seltene Klassen und häufige Klassen jeweils unterschiedlich. Das Feld der kostensensitiven Klassifikation ist gut erforscht. Für hochdimensionale Daten ist jedoch Feature Selection als weiterer Vorbereitungsschritt ratsam, um statistisches Rauschen zu reduzieren, den Fluch der Dimensionalität zu lindern, und die Genauigkeit der Klassifikation generell zu verbessern. Es ist grundsätzlich akzeptiert, dass herkömmliche Feature-Selection-Algorithmen nicht vollständig für Datensätze mit unausgeglichener Klassenverteilung geeignet sind. Es sind bereits einige Algorithmen vorgestellt worden, die Klassenungleichgewicht berücksichtigen, diese fokussieren sich dabei auf binäre Klassifikationsprobleme. Wir stellen einen neuen Feature-Selection-Algorithmus vor, der Klassenungleichgewicht bei Klassifikationsproblemen mit mehr als 2 Klassen mithilfe einer Gewichtungsfunktion abschwächt. Zudem erlaubt uns diese Funktion, sie mit bekannten, extern begründeten, Missklassifikationskosten zu kombinieren, die vom Datensatz abhängen. Wir werten diesen Ansatz an Datensätzen der UCI-Datenbank für maschinelles Lernen aus, und vergleichen ihn mit existierenden Algorithmen.

Selbstständigkeitserklärung

Hiermit erkläre ich, die vorliegende Arbeit selbstständig angefertigt, nicht anderweitig zu Prüfungszwecken vorgelegt und keine anderen als die angegebenen Hilfsmittel verwendet zu haben. Sämtliche wissentlich verwendete Textausschnitte, Zitate oder Inhalte anderer Verfasser wurden ausdrücklich als solche gekennzeichnet.

Potsdam, 20. Juli 2017

Daniel Oliver Theveßen