Faculty of Engineering and Information Technology
University of Technology, Sydney

Coupled Similarity Analysis in Supervised Learning

A thesis submitted in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

by

Chunming Liu

October 2015

CERTIFICATE OF AUTHORSHIP/ORIGINALITY

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Signature of Candidate

_____

Acknowledgments

First and foremost, I would like to express my deepest appreciation to my supervisor, Prof. Longbing Cao, for his professional guidance, selfless help and continuous support throughout my PhD study and research. I feel very lucky to have had him as my advisor and I will always remember our many discussions and his invaluable ideas. I sincerely thank him.

I am grateful to my colleagues and friends, Can Wang, Junfu Yin and Xuhui Fan, for their selfless support, and especially to Can Wang, for her advice regarding the technical aspects of my thesis.

My sincere gratitude is extended to my team leader, Zhigang Zheng, for his ongoing support in the Australian Taxation Office project. I am also grateful for the excellent ongoing help that I received from my colleagues and team members, David Wei, Mu Li, and Wei Cao.

I would like to thank all the staff in our Advanced Analytics Institute (AAI). Without their generous support this dissertation would not have been possible.

Last but not the least, I would like to thank my family: my wife and my daughter, for their unconditional love and support throughout my PhD candidature.

Chunming Liu

May 2015 @ UTS

Contents

List of Figures

List of Tables

Abstract

In supervised learning, the distance or similarity measure is widely used in a lot of classification algorithms. When calculating the categorical data similarity, the strategy used by the traditional classifiers often overlooks the inter-relationship between different data attributes and assumes that they are independent of each other. This can be seen, for example, in the overlap similarity and the frequency based similarity. While for the numerical data, the most used Euclidean distance or Minkowski distance is restricted in each single feature and assumes the features in the dataset have no outer connections. That can cause problems in expressing the real similarity or distance between instances and may give incorrect results if the inter-relationship between attributes is ignored. The same problems exist in other supervised learning, such as the classification tasks of class-imbalance or multi-label. In order to solve these research limitations and challenges, this thesis proposes an insightful analysis on coupled similarity in supervised learning to give an expression of similarity that is more closely related to the real nature of the problem.

Firstly, we propose a coupled fuzzy kNN to classify imbalanced categorical data which have strong relationships between objects, attributes and classes in Chapter 3. It incorporates the size membership of a class with attribute weight into a coupled similarity measure, which effectively extracts the inter-coupling and intra-coupling relationships from categorical attributes. As it reveals the true inner-relationship between attributes, the similarity strategy we have used can make the instances of each class more compact when

measured by the distance. That brings substantial benefits when dealing with class imbalance data. The experiment results show that our supposed method has a more stable and higher average performance than the classic algorithms.

We also introduce a coupled similar distance for continuous features, by considering the intra-coupled relationship and inter-coupled relationship between the numerical attributes and their corresponding extensions. As detailed in Chapter 4, we calculate the coupling distance between continuous features based on discrete groups. Substantial experiments have verified that our coupled distance outperforms the original distance, and this is also supported by statistical analysis.

When considering the similarity concept, people may only relate to the categorical data, while for the distance concept, people may only take into account the numerical data. Seldom have methods taken into account the both concepts, especially when considering the coupling relationship between features. In Chapter 5, we propose a new method which integrates our coupling concept for mixed type data. In our method, we first do discretization on numerical attributes to transfer such continuous values into separate groups, so as to adopt the inter-coupling distance as we do on categorical features (coupling similarity), then we combine this new coupled distance to the original distance (Euclidean distance), to overcome the shortcoming of the previous algorithms. The experiment results show some improvement when compared to the basic and some variants of kNN algorithms.

We also extend our coupling concept to multi-label classification tasks. The traditional single-label classifiers are known to be not suitable for multi-label tasks anymore, owing to the overlap concept of the class labels. The most used classifier in multi-label problems, ML-kNN, learns a single classifier for each label independently, so it is actually a binary relevance classifier. As a consequence, this algorithm is often criticized. To overcome this drawback, we introduce a coupled label similarity, which explores the inner relationship between different labels in multi-label classification according

to their natural co-occurrence. This similarity reflects the distance of the different classes. By integrating this similarity with the multi-label kNN algorithm, we improve the performance significantly. Evaluated over three commonly used verification criteria for multi-label classifiers, our proposed coupled multi-label classifier outperforms the ML-kNN, BR-kNN and even IBLR. The result indicates that our supposed coupled label similarity is appropriate for multi-label learning problems and can work more effectively compared to other methods.

All the classifiers analyzed in this thesis are based on our coupling similarity (or distance), and applied to different tasks in supervised learning. The performance of these models is examined by widely used verification criteria, such as ROC, Accuracy Rate, Average Precision and Hamming Loss. This thesis provides insightful knowledge for investors to find the inner relationship between features in supervised learning tasks.

Chapter 1

Introduction

1.1 Background

Supervised learning is one of the most fundamental tasks in machine learning. Supervised Learning aims at mapping a data item into one or several predefined classes. In supervised learning, we have both training examples with labels and testing examples without labels. A training example is an ordered pair $< x, y >$ where x is an instance with features (or attributes) and y is a class label. A testing example is an instance x without a label. The goal is to predict labels for the testing examples using the training examples. Let X be the set of instances and Y be the set of labels, then a classifier is a function $f : X \to Y$.

Many domains currently use computers to capture the details of business transactions such as banking records, credit card records, stock market records, retail sales, telecommunications and many other transactions. Such transactions can be used to uncover useful patterns and relationships. Based on real world data like these, Supervised Learning has been expanded to various fields. For example, the implementation of a decision support system for better business decisions; the classifying of patients with potential fatal diseases; the application of data mining approaches in planning regional health-care systems, and so on. In general, the supervised learning problems

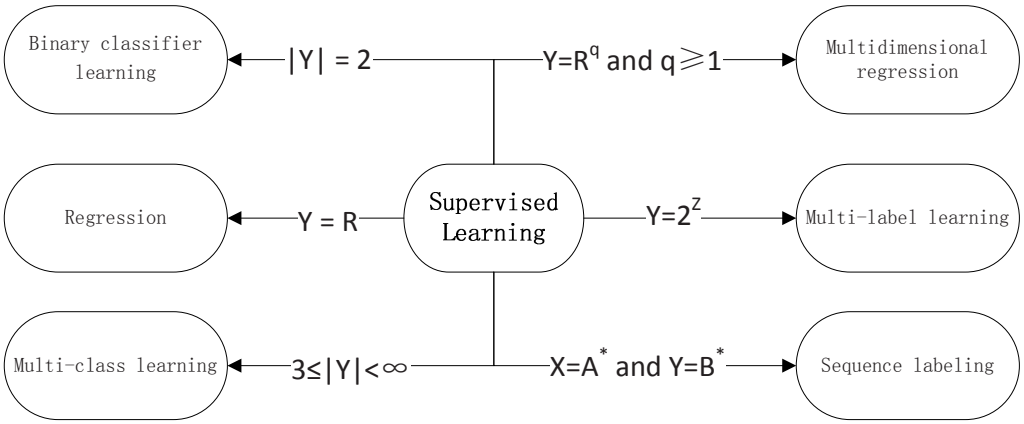

Figure 1.1: Categories of Supervised Learning

can be categorized as in Fig. 1.1.

There are many sub-fields in supervised learning. In recent years, the issue of assuming all the data features are independent while ignoring the inner relationship between them has attracted much attention by many data mining researchers. We all know that in real world applications, such as members in social networking and investors in capital markets, we almost always see quantitative features which are coupled to each other (Cao, Ou & Yu 2012). Accordingly, it is very important from both theoretical and practical perspectives to have a thorough understanding about the effect of the co-relationship among data attributes and labels in supervised learning. We illustrate this need in terms of Supervised Learning aspects below.

Nearest Neighbor Pattern Classification

The nearest-neighbor method is supposed to be the simplest of all algorithms for predicting the class of a testing instance. Its training phase is simple: just simply store all the training instances. To make a prediction, first, the method calculates the distance of the testing instance to every training instance. Second, it chooses the k closest training instances to the testing instance as its nearest neighbors, where $k \geq 1$ is a fixed integer. Last, it finds out the most common label among these neighbors, and this label is

the method's prediction for this testing instance.

The Nearest Neighbor Rule: As first defined in (Cover & Hart 1967), the concept of the Nearest Neighbor Rule is given below:

A set of n pairs $(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)$ is given, where the x_is take values in a metric space X upon which is defined a metric d, and the y_is take values in the set $\{C_1, C_2, \cdots, C_M\}$. Each y_i is considered to be the index of the category to which the ith individual belongs, and each x_i is the outcome of the set of measurements made upon that individual.

A new pair (x, y) is given, where only the measurement x is observable by the statistician, and it is desired to estimate y by utilizing the information contained in the set of correctly classified points. If $min(d(x_i, x)) = d(x'_n, x)$ where $i = \{1, 2, \cdots, n\}$, we call $x'_n \in \{x_1, x_2, \cdots, x_n\}$ a nearest neighbor to x.

The nearest neighbor rule decides x belongs to the category y'_n of its nearest neighbor x'_n.

Similarity and Distance Metrics

The concept of similarity or distance is a fundamental concept in many domains. A lot of algorithms in clustering, outlier detection, classification and regression compute the similarities between instances. Hence the choice of a particular similarity measure can turn out to be a major cause of success or failure for the algorithm. For the supervised learning tasks, the choice of a similarity measure can be as important as the choice of data representation or feature selection. In general, we take

$$\text{Similarity} = \frac{1}{1 + \text{Distance}} \tag{1.1}$$

as the relation between the concept "similarity" and "distance" in this thesis.

The terms of similarity metric and distance metric are often used to refer to any measure of affinity between two objects. In mathematics, a metric must conform to the following four criteria:

1. $d(x, y) \geq 0$; (non-negativity)

2. $d(x, y) = 0$ only if $x = y$; (identity)

3. $d(x, y) = d(y, x)$; (symmetry)

4. $d(x, z) \geq d(x, y) + d(y, z)$; (triangle inequality)

where $d(x, y)$ refers to the distance between two objects x and y.

In general, the similarity metric can be broadly divided in two categories: similarity measures for continuous data and similarity measures for categorical data. We use the concept of "distance" for continuous data rather than "similarity". The notion of distance measure for continuous data is straightforward due to the inherent numerical ordering. For example, the values pair (100kg, 120kg) is more like each other than to (100kg, 20kg), in other words, 100kg is closer to 120kg than to 20kg.

The Euclidean Distance is a method most widely used to compute the distance between two multivariate points. For two points $P = (x_1, x_2, \cdots, x_n)$ and $Q = (y_1, y_2, \cdots, y_n) \in \mathbb{R}^n$, the Euclidean distance between P and Q is defined as:

$$d(P, Q) = \sqrt{(x_1 - y_1)^2 + \cdots + (x_n - y_n)^2} = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{1.2}$$

Another metric used frequently is the Minkowski distance. The Minkowski distance is a metric on Euclidean space which can be considered as a generalization of both the Euclidean distance and the Manhattan distance. The Minkowski distance of order p between two points P and Q is defined as:

$$d_p(P, Q) = (\sum_{i=1}^{n} |x_i - y_i|^p)^{1/p} \tag{1.3}$$

Minkowski distance is typically used with $p = 1$ or $p = 2$. The latter is the Euclidean distance, while the former is sometimes known as the Manhattan distance.

However, the notion of similarity for categorical data is not as straightforward as for continuous data and hence is a major challenge. This is due

to the fact that the different values that a categorical feature takes are not inherently ordered and hence a notion of direct comparison between two categorical values is not possible. For example, we cannot tell whether the word "Cloudy" is in the middle of the words "Sunny" and "Rainy" or not. We only know that "Cloudy $\neq$ Sunny $\neq$ Rainy", while no "Cloudy $>$ Sunny" or "Cloudy $<$ Sunny". Thus, it is hard to compare different categorical values.

A commonly used similarity measure for categorical data is the overlap metric, which assigns a 1 if the values are identical and a 0 if the values are not identical. That is to say, assign the distance a 0 if the values are the same while assigning the distance a "∞" if the values are not identical. Then for two multivariate categorical data points, the similarity between them will be expressed by the number of features in which they match. The overlap measure does not distinguish between the different values taken by a feature. All matches as well as mismatches, are treated as equal importance and assigned a value 1 or 0. Jaccard (Jaccard 1912) coefficient is derived from the overlap and is adopted in several partitional and hierarchical clustering algorithms.

A variant of the overlap similarity is called the binary method, where each bit indicates the presence or absence of a possible feature value. Then the similarity between two objects is determined by the similarity between two corresponding binary vectors. The most popular measures for binary vectors belong to two families S_θ and T_θ (Gower & Legendre 1986). The transformation of data objects into binary vectors is a major drawback in which a lot of information may be removed.

The frequency based method, such as the cosine similarity, is another solution for measuring the similarity between categorical data. This method first counts the occurrence time of each categorical value (in its feature), then uses the number of occurrences to replace the original categorical value, and hence can transfer an object into a vector. The similarity of two categorical objects is then calculated using these vectors. Fig. 1.2 is an example of such transfer.

Original Categorical Data Table

	A_1	A_2	A_3	A_4	A_5
u_1	Morning	Movie	coffee	rainy	happy
u_2	Morning	Music	coffee	sunny	happy
u_3	Morning	Movie	tea	cloudy	sad
u_4	Afternoon	Book	tea	rainy	happy
u_5	Afternoon	Movie	cola	sunny	sad
u_6	Afternoon	Book	tea	cloudy	sad
u_7	Afternoon	Music	coffee	sunny	happy
u_8	Night	Movie	cola	cloudy	happy
u_9	Night	Book	tea	cloudy	happy
u_{10}	Night	Movie	tea	rainy	sad

Frequency Vectors Table

	A'_1	A'_2	A'_3	A'_4	A'_5
u_1	3	5	3	3	6
u_2	3	2	3	3	6
u_3	3	5	5	4	4
u_4	4	3	5	3	6
u_5	4	5	2	3	4
u_6	4	3	5	4	4
u_7	4	2	3	3	6
u_8	3	5	2	4	6
u_9	3	3	5	4	6
u_{10}	3	5	5	3	4

Figure 1.2: Example of Frequency Vectors

Cosine similarity is a measure of similarity between two vectors of an inner product space that measures the cosine of the angle between them. As the cosine of 0° is 1, then it is less than 1 for any other angle. It is thus a judgement of orientation and not magnitude: two vectors with the same orientation have a Cosine Similarity of 1, two vectors at 90° have a Cosine Similarity of 0, and two vectors diametrically opposed have a similarity of -1, independent of their magnitude. Cosine Similarity is particularly used in positive space, where the outcome is neatly bounded in [0,1].

Given two vectors of features, V^1 and V^2, the Cosine Similarity is represented as:

$$\text{Sim\_Cosine}(V^1, V^2) = \frac{V^1 \cdot V^2}{||V^1||\ \ ||V^2||} = \frac{\sum_{i=1}^{n} V_i^1 \times V_i^2}{\sqrt{\sum_{i=1}^{n}(V_i^1)^2} \times \sqrt{\sum_{i=1}^{n}(V_i^2)^2}}, \quad (1.4)$$

Class-Imbalance Classification

In classification, a dataset is said to be imbalanced if the number of instances which represents one class, usually the one that refers to the concept of interest (Chawla, Japkowicz & Kotcz 2004) and called positive or minority class, is smaller than that from other classes. That is to say, the number of negative (majority) instances outnumbers the amount of positive class

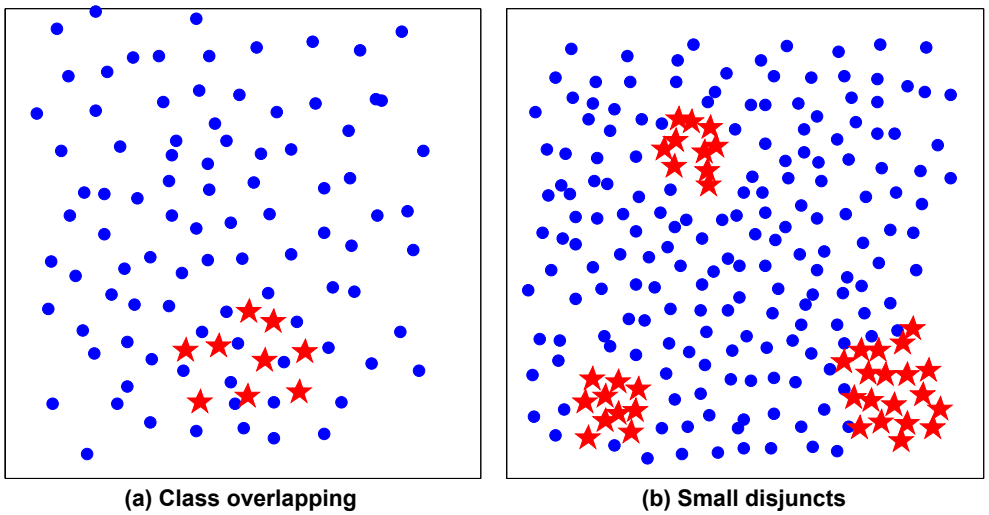

Figure 1.3: Example of Difficulties in Imbalanced Data Sets

instances. This problem is of great interest because it turns up in many real world classification problems, such as remote-sensing (Williams, Myers & Silvious 2009), pollution detection (Lu & Wang 2008), risk management (Huang, Hung & Jiau 2006), fraud detection (Cieslak, Chawla & Striegel 2006), and especially medical diagnosis (Mazurowski, Habas, Zurada, Lo, Baker & Tourassi 2008, Freitas, Costa-Pereira & Brazdil 2007, Kiliç, Uncu & Türksen 2007, Celebi, Kingravi, Uddin, Iyatomi, Aslandogan, Stoecker & Moss 2007, Peng & King 2008).

For an imbalanced dataset, the traditional classifiers have a bias toward the classes with a greater number of instances. The rules that correctly predict those instances are positively weighted in favor of the accuracy metric, whereas specific rules that predict instances from the minority class are usually ignored or treated as noise. In such a way, minority class instances are more often misclassified than those from other classes. The reason for this problem lies in several difficulties related to the imbalance:

- Small sample size [see Fig. 1.3(a)]: Imbalanced data sets generally do not have enough minority class examples.

(a) (b)

Figure 1.4: Examples of Multi-label Images

- Overlapping or class separability [see Fig. 1.3(a)]: When it occurs, discriminative rules are hard to induce and more general rules are induced that misclassify the minority class instances.

- Small disjuncts [see Fig. 1.3(b)]: The presence of small disjuncts occur when the concept represented by the minority class is formed of subconcepts (Weiss & Provost 2003).

Multi-Label Classification

In supervised learning, traditional single-label classification is concerned with learning from a set of examples that are associated with a single label l from a set of disjoint labels L, where $|L| > 1$. If $|L| = 2$, it is called a binary classification problem, while if $|L| > 2$, the problem is called a multi-class classification problem. However, in several real world domains, one single training instance is often associated with a label set Y, and $|Y| \geq 1$. Such data are called multi-label data, and these cause an class overlap problem. For example, in text or music classification, one document may be related to the topic of government and health, and a song may be labeled as rock and blues (McCallum 1999, Schapire & Singer 2000); In medical diagnosis, a disease may belong to multiple categories, and one single gene may have multiple functions and yield multiple labels (Clare & King 2001).

Table 1.1: Frequency of Co-occurrence

	morning	***afternoon***	***evening***	**Total**
Sunny	44	47	9	100
Cloudy	48	45	7	100
Rainy	8	8	84	100
Total	100	100	100	

Fig.1.4(a) and (b) show images that have been classified as a beach scene in traditional image classification. However, it is clear that image (a) can also be labeled as "Australia" and "Sunset", while image (b) can also be labeled as "city" and "tourist". We can see that every instance in a multi-label dataset is not a fuzzy member of each class due to ambiguity, but is a full member of each class due to multiplicity.

1.2 Limitations and Challenges

As there is no inherent order in the different values that a categorical feature takes, people always use the Overlap Similarity or similarity based on frequency to calculate the similarity between categorical values. But as they assume the features are independent of each other, that is to say, each part of the similarity is calculated only in one specific feature, this will create some problems. For example, considering a categorical dataset D, which has two categorical features: "Weather" and "Time", the feature "Weather" takes three possible values: {"Sunny", "Cloudy", "Rainy"}, and the feature "Time" takes three values: {"morning", "afternoon", "evening"}. Table 1.1 shows the frequency of the co-occurrence of the two features.

Based on the feature values given by dataset D, the overlap similarity between the two instances (Cloudy, morning) and (Cloudy, afternoon) is $\frac{1}{2}$, and the overlap similarity between (Rainy, morning) and (Rainy, afternoon) is also $\frac{1}{2}$. But the frequency distribution in Table 1.1 shows that (Cloudy, morning) and (Cloudy, afternoon) are frequent combinations, while (Rainy,

morning) and (Rainy, afternoon) are rare combinations in the dataset. Hence, the overlap measure is too simplistic to give equal importance to all matches and mismatches. While for frequency-based measures, the value of "Cloudy" and "Rainy" in the feature "Weather" occurs in both 100 times, and the value of "morning" and "afternoon" in feature "Time" occurs 100 times, so the distance of (Cloudy, morning) and (Rainy, afternoon) will be 0 for the frequency-based methods. That is to say, the frequency-based method also ignores the relationship between different features. Although there is no inherent order in categorical data, this example shows that there exists some hidden information in categorical data sets which can be used to define whether two values are more or less similar. We call this kind of data a Non-IID data (Cao 2014*b*).

The same problem exists in numerical data. As we know, the Euclidean and Minkowski distance (Gan, Ma & Wu 2007) is the widely used distance measure for numerical data, but they are often problematic when analyzing the numerical data because they assume that all the continuous features are independent, and hence fail to capture the genuine relationship between features. However, in the real world, people in social networking or the investors of capital markets always find quantitative features related to each other (Wang, Cao, Wang, Li, Wei & Ou 2011). Pearson's correlation coefficient (Gan et al. 2007) can measure the agreement of shapes between variables, but it is only suitable for the linear relationship between two variables. So developing an effective representation method for analyzing continuous variables is important for numerical featured supervised learning.

For class imbalance classification tasks, the re-sampling methods, which synthesize minority or remove majority instances in order to force the data balance, inevitably result in the problems of information loss or the addition of noise. The key issue which has not been analyzed comprehensively by the existing classification algorithms, such as kNN, is the coupling relationships between different feature values and class labels. This problem exists not only in categorical data, but in numerical and even mixed type data,

and reveals when computing the similarity or distance between instances. Considering such couplings (Wang et al. 2011) have been shown to be very important for capturing the non-IID (Cao 2014*b*) issues in the real world data, in which objects and object properties (features) are coupled rather than being independently and identically distributed as we usually suppose them to be. This is particularly important for the analysis of big data with complex behavior and social network data with diverse interactions.

In multi-label classification tasks, although ML-kNN is simple and powerful, there are some shortcomings in its processing strategy. ML-kNN uses the popular Binary Relevance (BR) strategy (Vembu & Gärtner 2011), which may artificially introduce outliers, and then tends to degrade the performance of the classifiers. Another problem of ML-kNN is the estimation of the posterior may be inaccurate due to the fact that it is highly possible that only a few samples are available for a given number of nearest neighbors with a certain number of labels. Furthermore, its ignorance of the inner relationship between labels when dealing with every single label is another issue which limits its usage.

Detailed introductions and evaluations of the related work are given in Chapter 2. Below, we summarize and list the main limitations and challenges of current research work in Supervised Learning.

- In supervised learning, when calculating the distance or similarity between categorical values, as there is no inherent order in the different categorical values, people usually use two types of measures: value-based and frequency-based. The value-based methods, such as the most frequently used Overlap Similarity, which assigns a 1 if the values are identical and a 0 if the values are not identical, treats all values indiscriminately. This can lead to the loss of the important information hiding in the categorical values. While for the frequency-based strategy, as most methods only calculate the times of the value occurrence, it can only reveal the general aspect of the value importance. Moreover, rather awkwardly, it treats different values but with same occurrence

times as the equally important object. How to represent the genuine similarity between categorical values is a big issue in machine learning.

- For numerical data learning, when calculating the distance or similarity between continuous values, the Euclidean and Minkowski distance (Gan et al. 2007) are the widely used distance measures. But no matter whether it is the Euclidean distance or the Minkowski distance, their definition is restricted to the calculation in a specific single feature, that is to say, it only considers the local distance while ignoring the fact that the influence comes from other features. These two distance measures fail to capture the genuine relationship between numerical values. Pearson's correlation coefficient (Gan et al. 2007) can measure the agreement of shapes between variables, but it is only suitable for the linear relationship between two variables. So how to develop an effective representation measure to analyze the similarity between continuous variables and make it more accurate to reality, is a challenge for numerical featured supervised learning.

- The Re-sampling method is one of the most used strategies for the class imbalance supervised learning tasks. It can force the data to become balanced by synthesizing minority instances or just deleting majority instances. The worth of this method is the re-sampled data can then be used by any traditional classification strategies, as it represents a data pre-processing method. However, when using the over-sampling on categorical featured data, as there is no inherent order in the categorical values, people cannot synthesize a real new "middle" value between two different categorical values as they can on numerical values. So the newly created instance may turn out to be a noise point. If the old value is just copied and the new one is not synthesized, it can cause over-fitting. The under-sampling method will inevitably cause a problem of information loss, especially on categorical data, for there is no obvious "border" for categorical values.

- The algorithm modifying method is another frequently used strategy for imbalance classification tasks. Unlike data re-sampling, this method does not alter the original data, and instead only modifies the existing algorithms to adapt them to the imbalanced data. A worthwhile feature of this method is that it can keep the original data unchanged, that is to say, the original information contained in the data is unchanged. The problem is that this method is mostly developed for imbalanced numerical data, and is seldom used on imbalanced categorical data, not to mention the mixed type imbalance data which contains both numerical features and categorical features. Although researchers try to avoid such kinds of data, the imbalanced data with mixed type features is popular in the real world. Most methods which try to deal with numerical attributes and categorical attributes are biased. People should work to find a solution to handle this kind of data properly.

- Multi-label classification is an important issue in supervised learning. People usually transform the multi-label data into single-label data or modify the existing algorithms, such as the simple and powerful ML-kNN, to adapt the multi-label situation to deal with these multi-label problems. But as ML-kNN uses the popular Binary Relevance (BR) strategy (Vembu & Gärtner 2011), that may artificially introduce outliers, this can tend to degrade the performance of the classifiers. Another problem exists in the nearest neighbors finding process. The ML-kNN and its variants rarely consider that the influence comes from other labels when searching the nearest neighbors with a certain label. It is highly possible that only a few samples are available. How many real neighbors indeed there are, is an issue should be explored thoroughly.

- In supervised learning, researchers used to calculate the relations based on the hypothesis that the data are "Independent and identically distributed", and this is called *IID* data. However in real world data, such

as social media data, sales transaction data and financial-market data, we can find so many data that breaks this rule. When handling these data, the traditional hypothesis will limit the performance of classifiers, and we have to build a new strategy to reveal the inner relationship inside the data, not only value to value, but the relationship between different features, and the relationship between class labels and features.

1.3 Research Issues and Objectives

Based on the aforementioned research limitations, in order to propose a comprehensive view about the effect of considering the coupling relationship in supervised learning, we will focus in this thesis on the following research issues and set our research objectives as:

- **Coupling similarity analysis for categorical, numerical and mixed type data**: When computing the similarity between categorical featured instances, only considering the feature value or feature value frequency is not enough. We try to find out the interactions within each categorical feature and the inter-coupling between different features, so as to produce a new similarity metric that can extract the similarity hiding in different levels in categorical data, from feature values to features and labels. For numerical features, based on the discrete group, we try to find an effective representation method to capture the genuine relationship between continuous features. As most previous works related to similarity are on categorical data, while works related to distance computing are on numerical data, we try to find a solution to combine these two measures and incorporate our coupling strategy, so as to take advantage of all the goodness and improve the similarity calculation.

- **Coupling analysis in Class-Imbalance**: In the class imbalance classification tasks, the under sampling and upper sampling methods balance the data by synthesizing or removing instances. This can cause

the information loss or the addition of noise. As the distance or similarity based algorithms always ignore the relationship between features when computing the similarity or distance between instances, we seek to find a solution which does not alter the original data but can still capture the inner relationship between features, and hence make the minority class instances more compact in relation to each other when revealed by distance.

- **Coupling analysis in Multi-Label**: ML-kNN is a widely known algorithm in multi-label classification. But as it is a binary relevance (BR) (Vembu & Gärtner 2011) classifier and may not calculate the posterior accurately, it cannot be widely used in real world problems. To address these problems, we try to propose a novel multi-label kNN learning method, which will incorporate the coupling similarity into labels when finding the nearest neighbors. Our supposed method will avoid the use of the BR strategy and improve the posterior accuracy by extending the concept of the nearest neighbors.

In the following chapters, we will analyze in detail and solve all of the above research issues.

1.4 Research Contributions

Current algorithms in Supervised Learning seldom consider the inter-relation between different features or between features and labels, let alone for categorical data or numerical data. They treat the features as isolated ones. This is not appropriate. In this thesis, by introducing the coupling relationship into several aspects of Supervised Learning, we prove the effectiveness of our coupling strategy. Our interest not only focuses on the coupling categorical similarity calculation, the coupling numerical distance calculation, the coupling on mixed type data, but also extends the coupling concept to class imbalance and multi-label classification tasks.

Specifically, in Coupling on Categorical data (Chapter 3), 14 data sets are taken from the UCI Data Repository (Bache & Lichman 2013), KEEL data set repository (Alcalá, Fernández, Luengo, Derrac, García, Sánchez & Herrera 2010), and a real Student Learning data taken from the records of an Australian university's database. The result shows that our method can achieve a much better AUC improvement (2.08% to 12.09%) on these data. In dealing with the Coupling in Numerical data issues (Chapter 4), several experiments are performed on 15 UCI data sets to show the effectiveness of our proposed coupled representation scheme for numerical objects. The result figures indicate that Coupled-kNN remarkably outperforms the original k-NNs for all the data sets, which implies that exploiting the relationship between different numerical features is effective. To verify the effectiveness of our coupling strategy on mixed type data (Chapter 5), we choose 10 data sets from the commonly used UCI, KEEL data and some real world data, which are all contain both numerical and categorical features. The experiment results show that our HC-kNN has a more stable and higher average performance than some other algorithms, such as the kNN, kENN, CCWkNN, SMOTE-based kNN, Decision Tree and NaiveBayes, when applied for class-imbalanced data which have both numerical and categorical features. A total of 8 commonly-used multi-label data sets (Table 6.6) are tested in Chapter 6. Overall, our proposed CML-kNN outperform all the compared methods on all three measures (the Hamming loss, one error and average precision (Schapire & Singer 2000)). The average ranking of our method on these data sets using three different metrics is (1.50, 1.50, 1.50), while the second best algorithm, BSVM, only achieves (2.50, 2.38, 2.25). The BR-kNN performs the worst, which only achieves (4.13,4.25,4.75). The improvement showed by quantitative experiment results confirms the advantage of our coupling strategy again.

Specifically, the key contributions of this thesis are as follows:

- By exploring the feature's weight, our supposed coupled fuzzy kNN can distinguish the feature's importance in the inter-feature coupling

calculation.

- Our supposed coupled fuzzy kNN captures the intra-feature coupling, namely the interactions within each categorical feature and inter-feature couplings between different categorical features, and produces a similarity metric that can extract the hidden similarity at different levels from one feature to different features and class labels.

- We extend the coupling strategy to continuous data. We explore the inter-coupled relationship among different numerical features, and this is measured by the relationship between the discrete groups of different features.

- The proposed coupling distance on numerical data outperforms the traditional distance measure, and the experiment results confirm the improvement.

- By assigning the corresponding size memberships to each class, we distinguish classes according to their sizes to handle the class-imbalance issue in a fuzzy way.

- In mixed type data with both categorical and numerical features, by doing the data discretization first, we extend the coupling concept to mixed type data. We explore the coupled interactions within each feature and between different features to produce a relatively more accurate similarity measurement for mixed-type data.

- We propose a novel multi-label learning algorithm based on lazy learning and the inner relationship between class labels.

- We introduce a new coupled label similarity for the multi-label kNN algorithm. Rather than only selecting the neighbors with a specific label, we use the coupled label similarity to include more similar neighbors into the process, and this is more natural in terms of selecting nearest neighbors.

- In our supposed coupled ML-kNN algorithm, we introduce a new frequency array strategy which is based on the extended coupling nearest neighbors.

1.5 Thesis Structure

The thesis is structured as follows:

Chapter 2 provides a literature review of the supervised learning problems which are analyzed in this thesis, such as the nearest neighbor algorithms, the similarity metrics, the class-imbalance and multi-label classification, and it also points out the advantages and disadvantages.

Chapter 3 proposes a novel coupled similarity based classification approach to cater for the class imbalance issue. By incorporating the fuzzy membership and the coupled similarity into one of the popular classifiers: kNN, we propose a new coupled fuzzy kNN (ie. CF-kNN). The experimental results show that CF-kNN outperforms the baselines. The classifiers incorporated with the proposed coupling strategy also perform better than their original versions.

Chapter 4 proposes a new algorithm based on the coupling analysis on numerical features to capture the inner relationship of continuous features. By doing data discretization, we introduce the inter-coupling relationship between different numerical features into the distance calculation process. Experiments on benchmark data sets show that our proposed method achieves a superior performance to the traditional way.

Chapter 5 provides a solution for mixed type data classification. In this chapter, we propose a hybrid coupled k-nearest neighbor classification algorithm (HC-kNN) for data which contains both numerical and categorical features, by doing discretization on numerical features first to adapt the inter coupling similarity as we do on categorical features, then combining this coupled similarity to the original similarity or distance, and finally to overcome the shortcomings of the previous distance measurement.

Chapter 6 extends the coupling concept to multi-label classification and supposes a good solution to overcome the disadvantages of the traditional ML-kNN algorithm. In this chapter, we present a new multi-label classification method which is based on the lazy learning approach to classify an unseen instance according to its k nearest neighbors. By introducing the coupled label similarity between class labels and extending the concept of nearest neighbors, the proposed method exploits the correlation among multi-label data and produces a good experiment result.

Chapter 7 concludes our research and outlines future directions related to this topic.

Appendix A shows a list of publications during my PhD studies. Appendix B lists the main denotations in this thesis.

Figure 1.5 shows the research profile of this thesis.

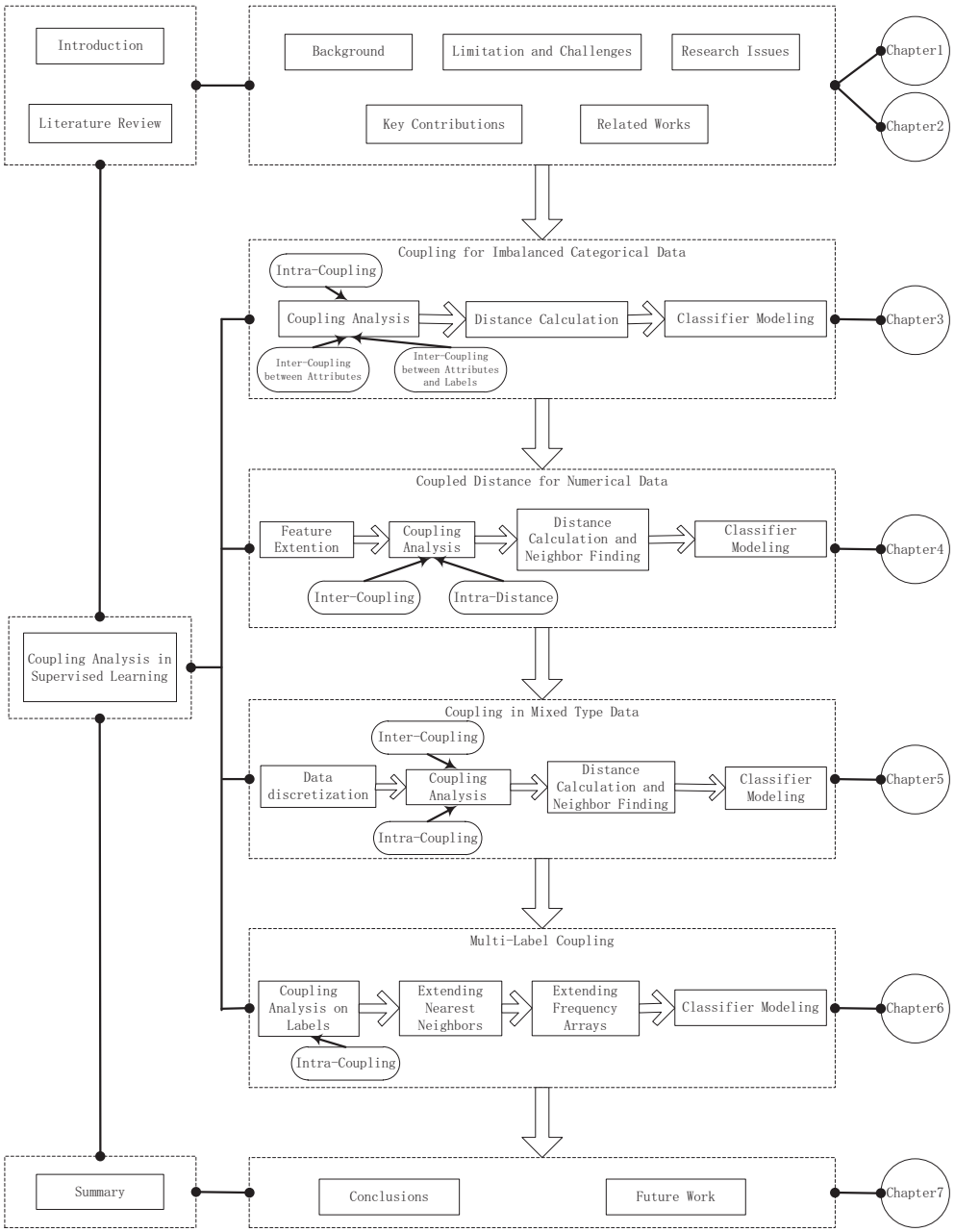

Figure 1.5: The Profile of Work in This Thesis

Chapter 2

Literature Review

This chapter introduces the related work about some supervised learning. Section 2.1 introduces some of basic nearest neighbor algorithms in classification, such as the traditional kNN, the ROC-kNN and the Fuzzy-kNN. Section 2.2 introduces the popular similarity metrics used for categorical data in supervised learning. It focuses on presenting the Context-free Similarity and Context-sensitive Similarity in categorical data. Section 2.3 presents the classic algorithms in class-imbalance classification tasks, and illustrates the basic concept of external method, internal method, cost-sensitive method and the ensemble-based method. Section 2.4 provides the popular handling methods for multi-label classification: problem transformation method and algorithm adaptation method. For each method, Section 2.4 also explains the main concept and the basic methodology. Finally, Section 2.5 summarizes the topics in this chapter.

2.1 Nearest Neighbor Classifier

The Nearest Neighbor Classifier is one of the most straightforward classifiers in machine learning techniques. Put simply, the classification is achieved by identifying the nearest neighbors to a query instance and using the most frequently occurred class in those neighbors as the class of the query. Although

being introduced decades before, this method still has particular importance today because the issues of poor run-time performance is not a problem any more with today's computational power. As Nearest Neighbor Classifier can be implemented simply and can be used to many different field directly, we choose it as the basic classifier for the whole thesis.

2.1.1 kNN

As mentioned previously, the strategy underlying the Nearest Neighbor Classification is quite straightforward: instances are classified based on the class of their nearest neighbors. It is often useful to take more than one neighbor into account so the technique is more commonly referred to as k Nearest Neighbor (kNN) Classification. In some research, it also called Memory-Based Classification, Lazy Learning, Instance-Based Classification or Case-Based Classification.

An example of kNN is as shown in Figure 2.1. It depicts a 3 Nearest Neighbor Classifier on a two-class problem in a two-dimensional feature space, and the point q_1 and q_2 are the query instances. We can see that the decision for q_2 is straightforward - all its three nearest neighbors are in class C_2, so it is classified as a C_2. For the query q_1, the situation is a bit more complicated as it has two neighbors in class C_1 and one in class C_2. This can be simply resolved by the majority voting metric which would classify it to be a class C_1.

So kNN classification has two stages once the number of neighbors (k) is defined: the first is the determination of the nearest neighbors and the second is the determination of the class according to those neighbors.

Assume that we have a training dataset $D = \{x_1, x_2, \cdots, x_n\}$, and the instances in D are described by a set of features $A = \{a_1, a_2, \cdots, a_m\}$. Each training instance is labeled with a class label $y_j \in Y$. Our objective is to classify an unknown instance q. For each $x_i \in D$ we can calculate the distance

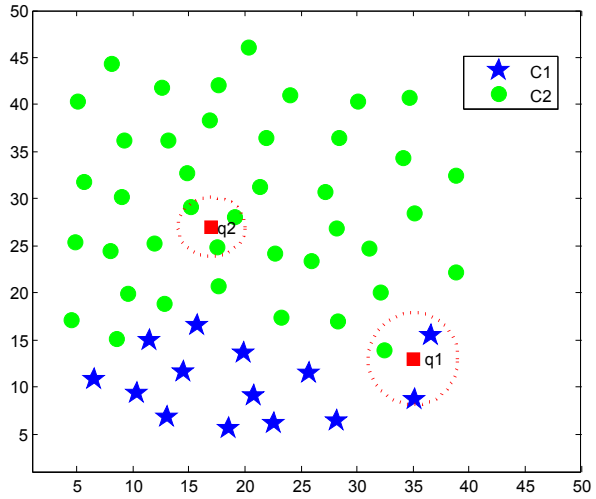

Figure 2.1: An Example of 3-Nearest Neighbor Classification

between q and x_i as follows:

$$d(q, x_i) = \sum_{a \in A} \omega_a \delta(q_a, x_{ia}), \tag{2.1}$$

There are a lot of possibilities for this distance metric, and we take a basic version for continuous and discrete attributes here:

$$\delta(q_a, x_{ia}) = \begin{cases} 0 & \text{a is discrete and } q_a = x_{ia} \\ 1 & \text{a is discrete and } q_a \neq x_{ia} \\ |q_a - x_{ia}| & \text{a is continuous} \end{cases} \tag{2.2}$$

The k nearest neighbors are selected based on this distance metric. Then there are a variety of ways in which the k nearest neighbors can be used to determine which class q belongs to. The most straightforward approach is to assign the majority class among the nearest neighbors to the query.

It will often make sense to assign more weight to the nearer neighbors in deciding the class of the query. A fairly general technique to achieve this is distance weighted voting where the neighbors get to vote on the class of the query case with votes weighted by the inverse of their distance to the query.

$$Vote(y_j) = \sum_{t=1}^{k} \frac{1}{d(q, x_t)^n} f(y_j, y_t), \tag{2.3}$$

Thus the vote assigned to class y_j by neighbor x_t is divided by the distance to that neighbor, i.e. $f(y_j, y_t)$ returns 1 if the class labels match and 0 otherwise. In equation 2.3 the n is normally equal to 1, but if $n > 1$, it can be used to further weaken the influence of much distant neighbors.

Another approach to voting is based on Shepard's work (Shepard 1987), i.e:

$$Vote(y_j) = \sum_{t=1}^{k} e^{-\frac{d(q,x_t)}{h}} f(y_j, y_t), \tag{2.4}$$

In his method, he uses an exponential function rather than inverse distance to evaluate the weight of the neighbors.

2.1.2 ROC-kNN

Although kNN has been applied for classification in many domains, it tends to suffer from poor classification accuracy when there are many features or few instances, or the data is very noisy, such as the gene data. Some researchers (Theilhaber, Connolly, Roman-Roman, Bushnell, Jackson, Call, Garcia & Baron 2002, Wu, Xing, Myers, Mian & Bissell 2005) adopt kNN in these kinds of domains, but got a generally inferior performance.

A wide range of proposals have been made to improve kNN. Such as (Hastie & Tibshirani 1996) did in their research. They propose alternative ways of computing the distance function. Rafiul Hassan and Maruf Hossain (Hassan, Hossain, Bailey & Ramamohanarao 2008) introduced a new method to derive a distance function for kNN based on feature weighting. The weight for each feature is calculated by considering the area under the Receiver Operating Characteristics (ROC) curve (Green, Swets et al. 1966). The area under the ROC curve (AUC) actually represents the probability that a randomly chosen positive example is correctly ranked with greater suspicion than a randomly chosen negative example. Moreover, this probability of correct ranking is the same quantity estimated by the non-parametric Wilcoxon statistic (Bradley 1997).

The intuitive outline of the technique is as follows: For a given dataset

D of n instances comprising m features: $x_1, x_2, x_3, \cdots, x_m$, each feature x_i (where $1 \leq i \leq m$) has some discriminative power, i.e., the influence of each feature on the classification accuracy can be measured. The ROC curve is plotted for a series of pairs which are each formed by a threshold value for the "classifier" feature x_i and the corresponding class label y_i. Then, when calculating the distance of a new test instance from a training example, the distance measure is modified using the AUC score as weight for that feature. (Hassan et al. 2008)

Previous work (Mamitsuka 2006, Hossain, Hassan & Bailey 2008, Ferri, Flach & Hernández-Orallo 2002) has established the use of an ROC curve for feature ranking and selection, to identify the discriminative features in the context of data and for each feature, the AUC is calculated.

For calculating the distance of a new test instance from a training instance, the ROC-kNN modifies the standard distance measure using the AUC score as a weight. Using all values of a feature to derive the ROC curve whose area will be calculated, may not be the best way to measure the weight or "importance" of a feature. Thus, the power of a feature may need to be evaluated within some context or sub-population. A natural way to form such a context or range for the ROC calculation, is to consider the two points between which the distance is being computed. The detail of ROC-kNN algorithm is shown in Algorithm 2.1:

2.1.3 Fuzzy-kNN

Another problem encountered in using the kNN classifier is that normally each of the sample vectors is considered equally important in the assignment of the class label to the input vector. This frequently causes difficulty in those places where the sample sets overlap. Atypical vectors are given as much weight as those that are truly representative of the classifiers. What's more, once an instance is assigned to a class, there is no indication of its "strength" of membership in that class. Some researchers (Keller, Gray & Givens 1985) address this problem in the kNN algorithm by incorporating

Algorithm 2.1 : ROC-KNN Algorithm

Require: $D = \{(x_1, Y_1), \cdots, (x_n, Y_n)\}$: The matrix of n training examples with the last column being the class, $\tau = (\tau_1, \cdots, \tau_n)$: The test sample, k: The number of neighbours, p: The order for Minkowski distance function, ε: The percentage of training instances to be covered when weighting each feature

Ensure: C: The class label for the test sample τ

1: **for** each labeled instance $(x_i, Y_i), (i = 1, \cdots, n)$ **do**
2: **for** each feature a_j , $1 < j <$ Number of features, m **do**
3: $A_j = CalculateROC(\{x_j, Y\}, \varepsilon, x_i, \tau)$
4: **end for**
5: Calculate $\Delta(x_i, \tau) = (\sum_{j=1}^{m}(A_j \cdot |x_i[a_j] - \tau[a_j]|^p))^{\frac{1}{p}}$
6: **end for**
7: $D_\tau^k = k$ nearest instances to τ
8: $C \leftarrow$ most frequent class in D_τ^k
9: **return** C
10: **end**

fuzzy set theory into the kNN rule.

Fuzzy sets were introduced by Zadeh in 1965 (Zadeh 1965). Since that time researchers have found numerous ways to utilize this theory to generalize existing techniques and to develop new algorithms in pattern recognition and decision analysis (Wang & Chang 1980, Bezdek 1981, Bezdek & Pal 1992, Klir & Yuan 1995).

Fuzzy Sets

Given a universe U of objects, a conventional crisp subset A of U is commonly defined by specifying the objects of the universe that are members of A. An equivalent way of defining A is to specify the characteristic function of

A,$\mu_A : U \to \{0, 1\}$ where for all $x \in U$

$$\mu_A(x) = \begin{cases} 1, & x \in A \\ 0, & x \notin A \end{cases} \tag{2.5}$$

Fuzzy sets are derived by generalizing the concept of a characteristic function to a membership function $u : U \to [0, 1]$. An example of a fuzzy set is the set of real numbers much larger than zero, which can be defined with a membership function as follows:

$$\mu(x) = \begin{cases} x^2/(x^2 + 1), & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{2.6}$$

Numbers that are not at all larger than zero are not in the set ($u = 0$), while numbers which are larger than zero are partially in the set based on how much larger than zero they are. Thus the impetus behind the introduction of fuzzy set theory was to provide a means of defining categories that are inherently imprecise (Bezdek 1981).

The advantage provided by fuzzy sets is that the degree of membership in a set can be specified, rather than just the binary is or is not a member. This can be especially advantageous in pattern recognition, where frequently objects are not clearly members of one class or another. Another good point of fuzzy theory used in kNN is it will specify to what degree the object belongs to each class, which is useful in supervised learning.

Fuzzy kNN Algorithm

The basis of the Fuzzy kNN algorithm is to assign membership as a function of the vector's distance from its k-nearest neighbors and those neighbors' memberships in the possible classes. The fuzzy algorithm must also search the labeled sample set for the k-nearest neighbors. Beyond obtaining these k samples, the procedures differ considerably. The Fuzzy kNN algorithm is as shown in Algorithm 2.2.

$$u_i(x) = \frac{\sum_{j=1}^{k} u_{ij}(1/||x - x_j||^{2/(m-1)})}{\sum_{j=1}^{k}(1/||x - x_j||^{2/(m-1)})} \tag{2.7}$$

Algorithm 2.2 : Fuzzy-kNN Algorithm

Require: $W = \{x_1, x_2, \cdots, x_n\}$ be the set of n labeled samples, u_{ij} be the membership in the ith class of the jth vector of the labeled sample set, $k : (1 < k < n)$ be the number of neighbours

Ensure: $u_i(x)$: The class label for the test sample x

```
i = 1
DO UNTIL (k nearest neighbors to x found)
    Compute distance from x to x_i
  IF (i ≤ k) THEN
    Include x_i in the set of k nearest neighbors
  ELSE IF (x_i closer to x than any previous nearest neighbor) THEN
    Delete the farthest of the k-nearest neighbors
    Include x_i in the set of k-nearest neighbors
  END IF
END DO UNTIL
i = 1
DO UNTIL (x assigned membership in all classes)
    Compute u_i(x) using Eq.(2.7)
    i = i + 1
END DO UNTIL
END
```

2.1.4 Summary

Nearest neighbor algorithms are mostly used for numerical data. When calculating the distance or similarity between continuous values, the Euclidean and Minkowski distance (Gan et al. 2007) are the widely used distance measures. But no matter Euclidean distance or Minkowski distance, their definition restricted the calculation in a specific attribute, that is to say, only considers the local distance while ignoring the influence comes from other objects or features. These two distance measures fail to capture the genuine relationship between numerical values. Pearson's correlation coefficient (Gan

et al. 2007) can measure the agreement of shapes between variables, but it is only suitable for the linear relationship between two variables.

So how to develop an effective representation measure for analyzing the similarity between continuous variables and make it more accurate to reality, is a challenge for numerical featured supervised learning.

2.2 Similarity for Categorical Data

We can not measure the difference between two categorical values as we do on numerical data. In categorical data classification tasks, two measurements will produce totally different classification results. In order to handle such tasks and get a better result, we have to choose the measurement more carefully and more reasonably. Whether or not the measurement used by the classifier take into account the relationship between features, is the key issue we should think about in this thesis. Based on how they utilize the context of the given attributes, the similarity measures for categorical data can be generally categorized into two different groups: Context-free and Context-sensitive, respectively. Fig. 2.2 illustrates the categories of categorical similarity measures.

2.2.1 Context-free Similarity

The conventional measures of distance are context-free (Michalski 1980), that is to say, the distance between any two data objects A and B is a function of these two objects only, and does not have any relationship of these objects to other data objects, i.e.,

$$\text{Similarity}(A, B) = f(A, B) \tag{2.8}$$

1) Supervised Methods

The most simple similarity measure for categorical data is the overlap metric, which assigns a 1 if the values are identical and a 0 if the values are not identi-

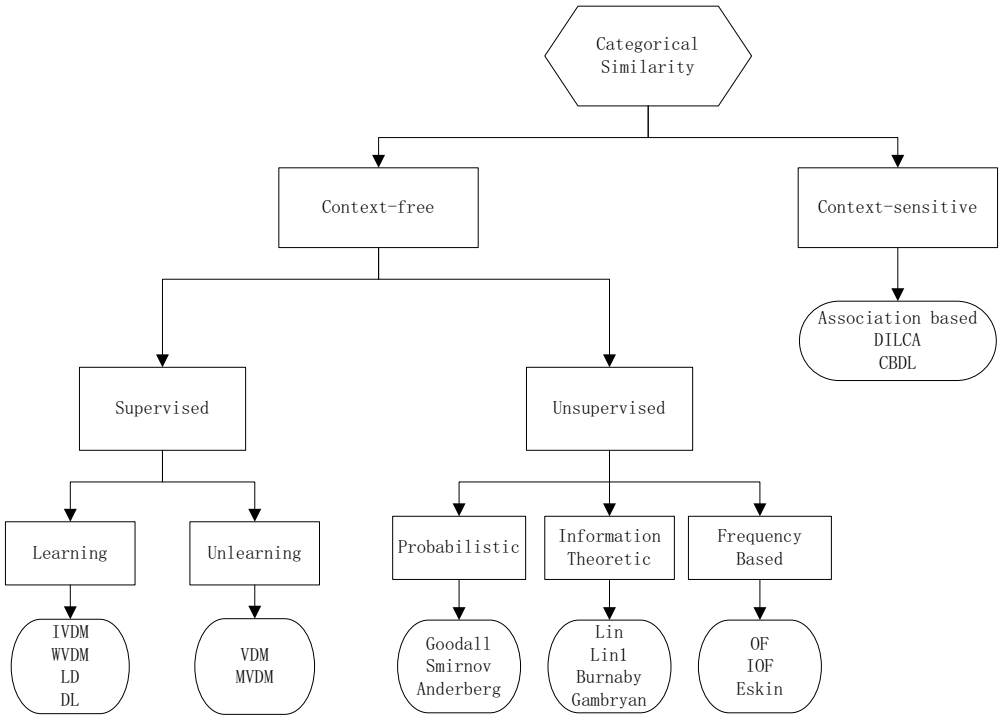

Figure 2.2: Categories of Categorical Data Similarity Measures

cal. That is to say, assign the distance a 0 if the values are same while assign the distance a "∞" if the values are not identical. Then for two multivariate categorical data points, the similarity between them will be expressed by the number of attributes in which they match. The overlap measure does not distinguish between the different values taken by an attribute, all matches as well as mismatches, are treated as equal and assign a value 1.

The value Difference Metric (VDM) proposed by (Stanfill & Waltz 1986) is a supervised non-learning approach, which gives more weight to an attribute value that is useful in discriminating the class. The disadvantage of this measure is that the computed distance is not symmetric and hence it is a non-metric measure. Cost and Salzberg (Cost & Salzberg 1993) modified VDM and proposed Modified Value Difference Metric (MVDM) which is symmetric.

Wilson and Martinez (Wilson & Martinez 1997) proposed two approaches for the categorical similarity: IVDM and WVDM, respectively. In IVDM, the continuous values are discretized into s equal width intervals, where s is an integer supplied by the user. The IVDM interpolate probabilities from only at the midpoints of each range. Instead of only at the midpoints of each range, but interpolates the probabilities from adjacent values, the WVDM samples the value of probability at each value occurring in the training set for each attribute.

Cheng et al. (Cheng, Li, Kwok & Li 2004) proposed the use of adaptive dissimilarity matrices for evaluating the dissimilarities between the categorical values. They computed all the pairwise distances between categorical attribute values, and optimized the error function by using gradient descent method.

Jierui Xie (Xie, Szymanski & Zaki 2010) proposed a new learning algorithm in which learns a dissimilarity measure for categorical data. In their algorithm, each categorical feature is mapped onto a continuous feature space whose values are real numbers. This learning algorithm is guided by the classification error based on kNN, and repeatedly updates the assignmen-

t of categorical symbols to real values to minimize the classification error. By utilizing the classification error this algorithm explores the relationship between categorical symbols.

2) Unsupervised Methods

Unsupervised measures are based on either frequency or entropy. Unsupervised approaches can be categorized into three approaches: Probabilistic, Information-theoretic and Frequency based approaches, respectively. The Probabilistic approaches take into account the probability of a given match taking place; the Information-theoretic approaches based on entropy considers the information content of a variable with respect to the dataset; the Frequency based approaches count the categorical value occurrence times.

To define the unsupervised similarity measures, we define these notations: Dataset $D = \{u_1, u_2, \cdots, u_n\}$ has n data points, where each point u_i is an object of m attribute values, each of the m attributes X^i is categorical. The attribute values for X^i are drawn from a set of n_i discrete values given as $(x_1^i, x_2^i, \cdots, x_{n_i}^i)$, which constitute the domain of X^i. Let $f(x_i)$ be the frequency of symbol x_i of attribute X in the dataset, then $p(x_i) = f(x_i) = t$.

Gambaryan (Gambaryan 1964) proposed a measure related to information entropy, which gives more weight to matches where the number of matches is between frequent and rare.

$$S(x_i, x_j) = \begin{cases} -[p(x_i)log2p(x_i) + (1 - p(x_i)) - log2(1 - p(x_i))], & \text{if } x_i = x_j \\ 0, & \text{otherwise} \end{cases} \tag{2.9}$$

Goodall (Goodall 1966) proposed a statistical measure in which less frequent attribute values make greater contribution to the overall similarity than frequent attribute values. Modified version of Goodall is proposed in (Boriah, Chandola & Kumar 2008), which defines,

$$S(x_i, x_j) = \begin{cases} 1 - \sum_{x_k \in X, p(x_k) < p(x_i)} p^2(x_k), & \text{if } x_i = x_j \\ 0, & \text{otherwise} \end{cases} \tag{2.10}$$

Smirnov (Smirnov 1968) not only considers the frequency, but also takes the distribution of the other attribute values into account.

$$S(x_i, x_j) = \begin{cases} 2 + \frac{n-f(x_i)}{f(x_i)} + \sum_{x_k \in X|\{x_i\}} \frac{f(x_k)}{n-f(x_k)}, & \text{if } x_i = x_j \\ \sum_{x_k \in X|\{x_i, x_j\}} \frac{f(x_k)}{n-f(x_k)}, & \text{otherwise} \end{cases} \tag{2.11}$$

Burnaby (Burnaby 1970) defines

$$S(x_i, x_j) = \begin{cases} 1, & \text{if } x_i = x_j \\ \frac{\sum_{x_k \in X} 2log(1-p(x_k))}{log(\frac{p(x_i)p(x_j)}{(1-p(x_i))(1-p(x_j))}) + \sum_{x_k \in X} 2log(1-p(x_k))}, & \text{otherwise} \end{cases} \tag{2.12}$$

Occurrence Frequency (Jones 1972) gives lower similarity to mismatches on less frequent symbols and higher similarity on mismatches on more frequent symbols.

$$S(x_i, x_j) = \begin{cases} \frac{1}{1+log(\frac{n}{f(x_i)})log(\frac{n}{f(x_j)})}, & \text{if } x_i \neq x_j \\ 1, & \text{otherwise} \end{cases} \tag{2.13}$$

Inverse Occurrence frequency (Jones 1972) assigns higher similarity to mismatches on less frequent symbols.

$$S(x_i, x_j) = \begin{cases} \frac{1}{1+log(f(x_i))log(f(x_j))}, & \text{if } x_i \neq x_j \\ 1, & \text{otherwise} \end{cases} \tag{2.14}$$

Lin (Lin 1998) gives more weight to matches on frequent values and lower weight to mismatches on infrequent values:

$$S(x_i, x_j) = \begin{cases} 2logp(x_i), & \text{if } x_i = x_j \\ 2log(p(x_i) + p(x_j)), & \text{otherwise} \end{cases} \tag{2.15}$$

Das and Mannila (Das & Mannila 2000) proposed ICD (Iterated contextual Distances) algorithm in which they suggest that attribute similarity and data objects similarity are interdependent. Lipika Dey (Ahmad & Dey 2007*b*) proposed a method to compute distance between two categorical values of

the same attribute in unsupervised learning for the categorical dataset. Can etc. (Wang et al. 2011) introduced a coupling similarity categorical data when doing data clustering. These methods even suggest that the distance between two categorical values of an attribute can be calculated with respect to other attributes.

2.2.2 Context-sensitive Similarity

Recently the researchers have come up with Context-sensitive measures of similarity, i.e.,

$$\text{Similarity}(A, B) = f(A, B, E) \tag{2.16}$$

where the similarity between A and B depends not only on A and B, but also on the relationship of A and B to other data objects represented by E.

Quang et al. (Le & Ho 2005) proposed an association based dissimilarity measure for categorical data, where the dissimilarity between two values of an attribute is estimated by using relations between other attributes under the condition of giving these two values. For data sets whose attributes are highly independent, these correlation based similarity measures are not suitable. Moreover for data sets with high dependency between attributes the associative similarity measures are absolutely suitable.

Correntropy proposed by (Gunduz & Principe 2009) is a kernel based similarity measure which non-linearly maps the input space into some higher dimensional feature space in which inner products are computed efficiently. Correntropy similarity measure includes the statistical distribution information and time structure of signals in a single measure.

Zeinab et al. (Khorshidpour, Hashemi & Hamzeh 2010) proposed a novel approach for distance learning based on the Context information. Their approach of context selection includes computing the Dependency Score between a given attribute and all other attributes. Dependency score is the information theoretic measure derived from entropy. The higher the value of Relevance Score indicates the more relevant the attribute. After choosing the context of an attribute, KL divergence method (Kullback &

Leibler 1951, Kullback 1959) is used to compute dissimilarity between probability distributions.

Dino et al. (Ienco, Pensa & Meo 2012) presented a new methodology to compute a context-based distance between values of a categorical variable and apply this technique to hierarchical clustering of categorical data. They introduced a new method called DILCA, to compute the distance between any pair of values of a specific categorical attribute. They provided two methods for the selection of a suitable context. The first one is parametric method and the second one is a fully automatic one. How the context is defined for the given attribute plays an important role in finding the distance between attribute values of the target attribute. For the context selection, they used the symmetric uncertainty measure (mutual information) which is derived from entropy (Cover 1991). In the first step they selected a relevant subset of the whole attributes set that they use as the context for a given attribute and in the second step they compute dissimilarity between a pair of values of the same attribute using the context defined in the first step.

The similarity measures discussed in this section are pictorially depicted in Fig. 2.2.

2.2.3 Summary

In supervised learning, when calculating the distance or similarity between categorical values, as there is no inherent order in the different categorical values, people usually use two types of measures: value-based and frequency-based. The value-based methods, such as the most frequently used Overlap Similarity, which assigns a 1 if the values are identical and a 0 if the values are not identical, treated all values indiscriminately. That will lose the importance information hide in the categorical values. While for the frequency-based strategy, as most methods only calculate the times of the value occurrence, it can only reveal the general aspect of the value importance. Moreover, an awkward thing will occur that it treats different values but with same occurrence times as the same important object. How to repre-

sent the genuine similarity between categorical values is a big issue in machine learning and it will affect the learning result greatly. Taking into account the relationship between two values from one feature, two values from two features even from features and class labels will be a genuine expression for categorical data.

2.3 Class-Imbalance Classification

In recent years, the problem of imbalance has been presented in many real world classification problems, for example, the oil spills detection (Kubat, Holte & Matwin 1998), face recognition (Liu & Chen 2005), medical diagnosis (Mazurowski et al. 2008), fault diagnosis (Yang, Tang, Shintemirov & Wu 2009, Zhu & Song 2010), anomaly detection (Khreich, Granger, Miri & Sabourin 2010, Tavallaee, Stakhanova & Ghorbani 2010), and e-mail foldering (Bermejo, Gámez & Puerta 2011). Due to the importance and the wide usage of the imbalance problem, a number of solutions have been proposed these years. Generally, depending on how to deal with the class imbalance, these proposals can usually be categorized into three groups: the external methods, the internal methods and the ensemble methods, respectively.

2.3.1 External Methods

The external methods, also called Data Level Methods, add a preprocessing step to force the data distribution balance so as to decrease the effect of the skewed class distribution in the learning process (Batista, Prati & Monard 2004, Chawla, Bowyer, Hall & Kegelmeyer 2002, Chawla et al. 2004). It is usually a positive solution (Batista et al. 2004, Fernández, García, del Jesus & Herrera 2008) and the main advantage is that they are independent to the subsequent classifiers.

1) Under-sampling

Random under-sampling (Kotsiantis & Pintelas 2003) is a non-heuristic method that aims to balance the class distribution through randomly eliminating the majority class instances. It is the easiest way to get data balanced. The major drawback of random under-sampling is it will discard useful data that may be important for the learning process. There is another problem for this approach. We all know that one of the purposes of machine learning is to estimate the distribution of the target population. As the target distribution is unknown, we try to estimate the population distribution using a sample distribution. Statistics prove that as long as the sample is drawn randomly, the sample distribution can be used to estimate the population distribution from where it was drawn. Once we performed under-sampling of the majority class, however, the sample is no longer random.

Given two instances E_i and E_j belonging to different classes, and $d(E_i, E_j)$ is the distance between E_i and E_j; a (E_i, E_j) pair is called a Tomek link if there is not an instance E_1, which exists $d(E_i, E_1) < d(E_i, E_j)$ or $d(E_j, E_1) < d(E_i, E_j)$. If two instances form a Tomek link, then either one of these instances is noise or both instances are borderline. Tomek links can be used as an under-sampling method and to eliminate only the instances belonging to the majority class.

Kubat and Matwin (Kubat, Matwin et al. 1997) randomly draw one majority class instance and all instances from the minority class and put these instances in E'. Afterwards they use a 1-NN over the instances in E' to classify the instances in E. Every misclassified instance from E is moved to E'. After several repetitions, the E' will be the under-sampled dataset and can be used in the subsequent learning tasks. The idea behind this method is to eliminate the instances from the majority class that are distant from the decision border, since these sorts of instances might be considered less relevant for learning.

2) Over-sampling

Random over-sampling is a non-heuristic method that aims to balance class distribution through the random replication of minority class instances. It is easy to achieve data balance, but several researchers (Chawla et al. 2002, Kubat et al. 1997) find that random over-sampling can increase the likelihood of over-fitting, since it makes exact copies of the minority class instances. For example, a symbolic classifier might construct rules that are apparently accurate, but actually cover only one replicated instance. Another problem for over-sampling is it will increase the computational complexity, especially when the dataset is already fairly large.

SMOTE (Chawla et al. 2002), the Synthetic Minority Over-sampling Technique, generates synthetic minority instances to over-sample the minority class. Its main idea is to form new minority class instances by interpolating between several minority class instances that lie together. For every minority instance, its k nearest neighbors of the same class are calculated, then some instances are randomly selected from the neighbors according to the over-sampling rate. After that, the SMOTE method generates new synthetic instances along the line between the minority instance and its selected nearest neighbors. Based on this strategy, the over-fitting problem is avoided and the newly generated instances will make the decision boundaries of the minority class spread further into the majority class space.

By calculating the distances among all instances, the Modified synthetic minority over-sampling technique (MSMOTE) (Hu, Liang, Ma & He 2009) divides the instances of the minority class into three groups: safe, border and latent noise instances, respectively. Unlike the SMOTE method, the MSMOTE applies different strategy on the previously divided groups to generate new instances: for the safe instances, the algorithm randomly selects data point from the kNN (same as SMOTE); for border instances, it only selects from the nearest neighbors; while for latent noise instances, it selects nothing.

3) Feature Selection

Some researchers extend the feature selection to the class imbalance problems. Zheng et al. (Zheng, Wu & Srihari 2004) propose their strategy of feature selection used for imbalance problem. They suggest a feature selection framework, which selects features for positive and negative classes separately and then explicitly combines them. The authors show simple ways of converting existing measures so that they separately consider features for negative and positive classes.

2.3.2 Internal Methods

The internal methods, also known as algorithm level methods, modify the existed algorithms to handle the class imbalance situation (Quinlan 1991, Zadrozny & Elkan 2001, Wu & Chang 2005).

Generally, a common strategy to deal with the class imbalance problem is to choose an appropriate inductive bias. For decision trees, one approach is to adjust the probabilistic estimate at the tree leaf (Quinlan 1991, Zadrozny & Elkan 2001) and another approach is to develop new pruning techniques (Zadrozny & Elkan 2001). For SVMs, one method is to use different penalty constants for different classes (Lin, Lee & Wahba 2002), and another method is to adjust the class boundary (Wu & Chang 2003). For association rule mining, multiple minimum supports for different classes are specified to reflect their varied frequencies in the database (Liu, Ma & Wong 2000).

In recognition-based one-class learning, such as neural network training (Japkowicz 2001) and SVMs (Manevitz & Yousef 2002), they modeled a system with only instances of the target class in the absence of the counter instances. This approach does not try to partition the hypothesis space with boundaries that separate positive and negative instances, but it attempts to make boundaries which surround the target concept. It measures the amount of similarity between a query object and the target class, where a threshold on the similarity value is introduced. In some domains, the one-class approach

is reported to be superior to discriminative (two-class learning) approaches (Japkowicz 2001). The threshold in this approach represents the boundary between the two classes. A strict threshold means that positive data will be sifted out, while a loose threshold will include considerable negative samples. Hence, how to set up an effective threshold is crucial for the performance of these methods.

2.3.3 Cost-sensitive Methods

Cost-sensitive methods combine both external and internal methods to incorporate misclassification costs for each class in the learning phase (Margineantu 2002, Chawla, Cieslak, Hall & Joshi 2008, Freitas et al. 2007). A cost matrix encodes the penalty of classifying instances from one class as another class. Assume that $Cost(i, j)$ denotes the cost of predicting an instance from class i as class j, then $Cost(+, -)$ is the cost of misclassifying a positive (minority class) instance as the negative (majority class) instance and $Cost(-, +)$ is the cost of misclassifying a negative (majority class) instance as the positive (minority class) instance. As the minority instances are usually more important than the majority ones in the class imbalance problems, the cost of misclassifying a positive instance should outweigh the cost of misclassifying a negative one, i.e. $Cost(+, -) > Cost(-, +)$, and making a correct classification should not get any penalty, i.e. $Cost(+, +) = Cost(-, -) = 0$. The target of cost-sensitive learning process is then turned to minimize the number of high cost errors and the total misclassification cost. The cost-sensitive learning can be further categorized into three kinds:

- **Weighting the data space**. The distribution of the training set is modified with regard to misclassification costs. That makes the modified distribution biased for the costly classes. This approach can be explained by the Translation Theorem derived in (Zadrozny, Langford & Abe 2003). Against the normal space without considering the cost item, let us call a data space with domain $X \times Y \times C$ as the cost-space, where X is the input space, Y is the output space and C is the cost

associated with mislabeling that instance. If we have instances drawn from a distribution D in the cost-space, then we can have another distribution D' in the normal space that

$$D'(X,Y) = \frac{C}{E_{X,Y,C\sim D}[C]} D(X,Y,C). \tag{2.17}$$

Here $E_{X,Y,C\sim D}[C]$ is the expectation of cost values. According to the translation theorem, those optimal error rate classifiers for D' will be optimal cost minimizers for D. Hence, when we update sample weights integrating the cost items, choosing a hypothesis to minimize the rate of errors under D' is equivalent to choosing the hypothesis to minimize the expected cost under D.

- **Making a specific classifier learning algorithm cost-sensitive.** For example, in the context of decision tree induction, the tree-building strategies are adapted to minimize the misclassification costs. The cost information is used to: determine whether a subtree should be pruned (Bradford, Kunz, Kohavi, Brunk & Brodley 1998) or choose the best attribute to split the data (Ling, Yang, Wang & Zhang 2004, Riddle, Segal & Etzioni 1994).

- **Using Bayes risk theory to assign each sample to its lowest risk class.** For example, a typical decision tree for a binary classification problem assigns the class label of a leaf node depending on the majority class of the training samples that reach the node. A cost-sensitive algorithm assigns the class label to the node that minimizes the classification cost (Domingos 1999, Zadrozny & Elkan 2001).

Converting instance-dependent costs into instance weights is also known as cost-sensitive learning by instance weighting (Abe, Zadrozny & Langford 2004), and is a data-level method. The other two methods in cost-sensitive which adapt the existing learning algorithms, are algorithm-level method. Cost-sensitive learning assumes that a cost-matrix can be known in the process, however, the cost matrix is often unavailable for some given data sets.

2.3.4 Ensemble Based Methods

In addition to these approaches, another group of techniques emerges when the use of ensembles of classifiers is considered. Ensemble methods (Polikar 2006, Rokach 2010) are designed to increase the accuracy of a single classifier by training several different classifiers and combining their decisions to output a single class label. Ensemble methods are well known in machine learning and are applied to many real world domains (Oza & Tumer 2008, Silva, Lotric, Ribeiro & Dobnikar 2010, Yang & Chen 2011, Xu, Cao & Qiao 2011).

When forming ensembles, creating diverse classifiers but maintaining their consistency with the training set is a key factor to make them accurate. Diversity in ensembles has a thorough theoretical background in regression problems (bias-variance (Ueda & Nakano 1996) and ambiguity (Krogh, Vedelsby et al. 1995)); however, in classification, the concept of diversity is still formally ill-defined (Brow, Wyatt, Harris & Yao 2005). Even though, diversity is necessary (Tumer & Ghosh 1996, Hu 2001, Kuncheva 2005) and there exists several different ways to achieve it (Rokach 2009). AdaBoost (Schapire 1990, Freund & Schapire 1995) and Bagging (Breiman 1996) are the most common ensemble learning algorithms among them. There exists many variants and other different approaches (Kuncheva 2004).

Because of their accuracy-oriented design, ensemble learning algorithms that are directly applied to imbalanced data sets do not solve the problem that underlays the base classifier by themselves. However, their combination with other techniques to tackle the class imbalance problem have led to positive results. These hybrid approaches are in some sense algorithm level approaches (since they slightly modify the ensemble learning algorithm), but they do not need to change the base classifier, which is one of their advantages. The modification of the ensemble learning algorithm usually includes data level approaches to preprocess the data before learning each classifier (Chawla, Lazarevic, Hall & Bowyer 2003, Seiffert, Khoshgoftaar, Van Hulse & Napolitano 2010, Błaszczyński, Deckert, Stefanowski & Wilk 2010, Liu, Wu & Zhou 2009).

1) Bagging

Breiman (Breiman 1996) introduced the concept of bootstrap aggregating to construct ensembles. It consists of training different classifiers with bootstrapped replicas of the original training dataset. That is, a new dataset is formed to train each classifier by randomly drawing (with replacement) instances from the original dataset while maintaining the original dataset size. Hence, diversity is obtained with the re-sampling procedure by the usage of different data subsets. Finally, when an unknown instance is presented to each individual classifier, a majority or weighted vote is used to infer the class. Algorithm 2.3 shows the pseudocode for Bagging.

Algorithm 2.3 : Bagging Algorithm

Require: TS: Training set; T: Number of iterations; n: Bootstrap size; I: Weak learner

Ensure: Bagged classifier: $H(x) = sign(\sum_{t=1}^{T} h_t(x))$ where $h_t \in [-1, 1]$ are the induced classifiers

1: **for** $t = 1$ to T **do**
2: $S_t \leftarrow$ RandomSampleReplacement(n,TS)
3: $h_t \leftarrow \mathrm{I}(S_t)$
4: **end for**

Pasting small votes is a variation of Bagging originally designed for large data sets (Breiman 1999). Large data sets are partitioned into smaller subsets, which are used to train different classifiers. There exists two variants, Rvotes that create the data subsets at random and Ivotes that create consecutive data sets based on the importance of the instances; important instances are those that improve diversity. The way the data sets are created consists of the usage of a balanced distribution of easy and difficult instances. Difficult instances are detected by out-of-bag classifiers (Breiman 1996), that is, an instance is considered difficult when it is misclassified by the ensemble classifier formed of those classifiers which did not use the instance to be trained. These difficult instances are always added to the next data subset,

whereas easy instances have a low chance to be included.

2) Boosting

Boosting was introduced by Schapire (Schapire 1990). Schapire proved that a weak learner which is slightly better than random guessing can be turned into a strong learner in the sense of Probably Approximately Correct (PAC) learning framework. AdaBoost (Freund & Schapire 1995) is the most representative algorithm in this family, it was the first applicable approach of Boosting, and it has been appointed as one of the top ten data mining algorithms (Wu & Kumar 2009). AdaBoost is known to reduce bias besides from variance, and similarly to SVMs boosts the margins (Friedman, Hastie, Tibshirani et al. 2000, Rudin, Daubechies & Schapire 2004). AdaBoost uses the whole dataset to train each classifier serially, but after each round, it gives more focus to difficult instances, with the goal of correctly classifying instances in the next iteration that were incorrectly classified during the current iteration. Hence, it gives more focus to instances that are harder to classify, the quantity of focus is measured by a weight, which initially is equal for all instances. After each iteration, the weights of misclassified instances are increased; on the contrary, the weights of correctly classified instances are decreased. Furthermore, another weight is assigned to each individual classifier depending on its overall accuracy which is then used in the test phase; more confidence is given to more accurate classifiers. Finally, when a new instance is submitted, each classifier gives a weighted vote, and the class label is selected by majority. We show the pseudocode for AdaBoost in Algorithm 2.4.

2.3.5 Evaluation

It is well known that the traditional classifier's evaluation criterion, accuracy rate, is not suitable for this imbalance situation any more. For instance, let us consider a dataset whose imbalance ratio is 1:1000 (i.e., for each instance in the positive class, there will be 1000 negative class instances). A classifier

Algorithm 2.4 : AdaBoost Algorithm

Require: Training set $TS = \{x_i, y_i\}$, $i = 1, 2, \cdots, N$; and $y_i \in \{-1, +1\}$; T: Number of iterations; I: Weak learner

Ensure: Boosted classifier: $H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$ where h_t, α_t are the induced classifiers (with $h_t(x) \in \{-1, +1\}$) and their assigned weights, respectively

1: $D_1(i) \leftarrow 1/N$ for $i = 1, 2, \cdots, N$
2: **for** $t = 1$ to T **do**
3: $h_t \leftarrow I(S, D_t)$
4: $\varepsilon_t \leftarrow \sum_{i, y_i \neq h_t(x_i)} D_t(i)$
5: **if** $\varepsilon_t > 0.5$ **then**
6: $T \leftarrow t - 1$
7: **return**
8: **end if**
9: $\alpha_t = \frac{1}{2} ln(\frac{1-\varepsilon_t}{\varepsilon_t})$
10: **for** $i = 1$ to N **do**
11: $D_{t+1}(i) = D_t(i) \cdot e^{-\alpha_t h_t(x_i) y_i}$
12: **end for**
13: Normalize D_{t+1} to be a proper distribution
14: **end for**

will obtain an accuracy as high as 99.9% if it just simply ignores all the minority instances but classifies all the instances into negative ones. Based on the confusion matrix for a two class problem in Table 2.1, we have

$$\text{AccRate} = \frac{TP + TN}{TP + FP + FN + TN} \tag{2.18}$$

But the accuracy rate does not distinguish between the numbers of correctly classified examples of different classes. One solution for this is to use the Receiver Operating Characteristic (ROC) graphic (Bradley 1997). This graphic allows the visualization of the trade-off between the benefits (TP) and costs (FP); thus, it evidences that any classifier cannot increase the number of true positives without the increment of the false positives. The area under

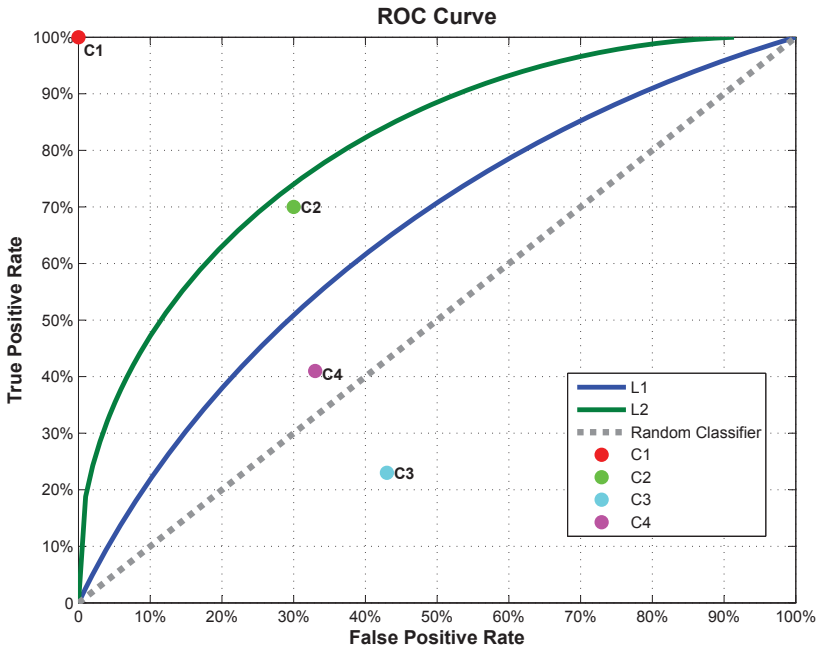

Figure 2.3: Example of ROC

the ROC curve (AUC) (Huang & Ling 2005) corresponds to the probability of correctly identifying which one of the two stimuli is noise and which one is signal plus noise. AUC provides a classifier's performance measure for the evaluation of which model is better on average. The bigger the AUC of a classifier has, the better average performance the classifier is.

Fig. 2.3 shows an example of the ROC space. Any classifier under the random classifier line will not be a good choice for classification tasks, such as classifier C_3; while any classifier over the random classifier line performs better, such as the classifier C_2 and C_4. The classifier C_1 represents the perfect classifier with all positive and negative instances correctly classified to their classes. The AUC measure is computed just by obtaining the area of the graphic. The bigger the AUC is, the better average performance the classifier has, in Fig. 2.3, as we have AUC(L_2)>AUC(L_1), hence L_2 will have a better average performance than L_1.

Table 2.1: Confusion Matrix for A Two-class Problem

	Condition positive	**Condition negative**
Test positive	True positive (TP)	False positive (FP)
Test negative	False negative (FN)	True negative (TN)

2.3.6 Summary

The Re-sampling method is one of the most used strategy for class imbalance supervised learning tasks. It can force the data become balance by synthesizing minority instances or deleting majority instances. The goodness of this method is the re-sampled data can then be used by any traditional classification strategies, as it is a data pre-processing method. However, when using the over-sampling on categorical featured data, as there is no inherent order in the categorical values, people cannot synthesize a real new "middle" value between two different categorical values as it does on numerical values. So the newly created instance may turn out to be a noise point. If just copy the old value while not synthesizes a new one, it will cause an over-fitting problem. The under-sampling method will inevitably cause the problem of information lost, especially on categorical data, for there is no obvious "border" for categorical values.

The algorithm modifying method is another frequently used strategy for imbalance classification tasks. Not like data re-sampling, this method does no alter for the original data, while only modifies the existing algorithms to adapt them to the imbalanced data. The good point of it is this method can keep the original data unchanged, that is to say, the original information contained in the data is unchanged. The problem of it is these methods are mostly developed for imbalanced numerical data, seldom used on imbalanced categorical data, not to mention the mixed type imbalance data which contains both numerical features and categorical features. Although researchers try to avoid such kind of data, the imbalanced data with mixed type features is popular in the real world. Any method which trying to dealing with numerical attributes and categorical attributes is biased. Due to the imbalance

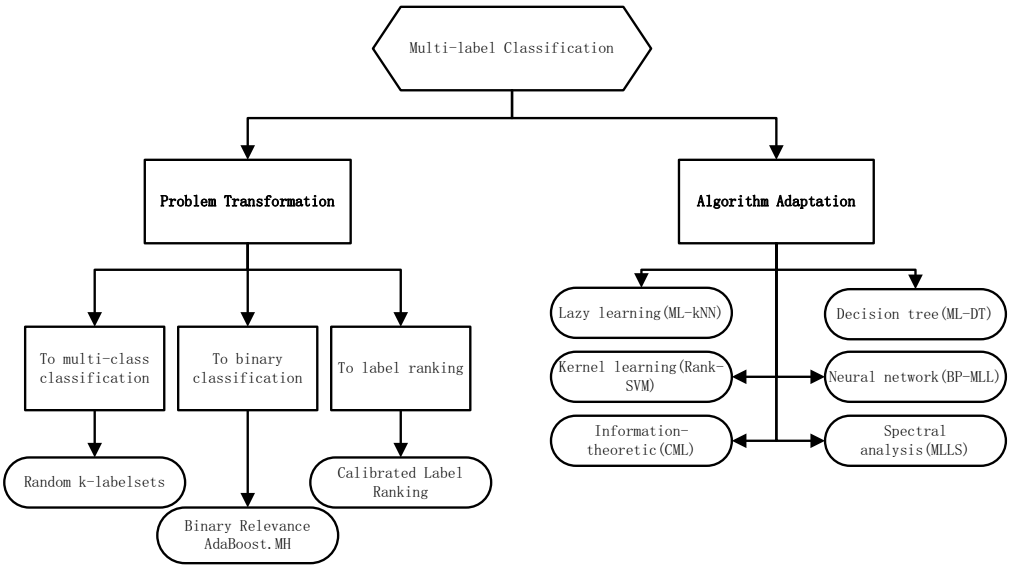

Figure 2.4: Categorization of Multi-label Classification Algorithms

problem is widely exists in the real world, no matter in numerical data or categorical data, even mixed type data, we want to find out a solution in this thesis to handle this issue properly.

2.4 Multi-Label Classification

Muti-label classification is a relatively new filed in Supervised Learning. As the traditional methods are no longer useful in dealing such issues, people revised them to adapt the new situation. In dealing with multi-label classification problems, many solutions have been proposed. As shown in Fig. 2.4, these methods can be broadly divided into two categories: problem transformation methods and algorithm adaptation methods, respectively.

2.4.1 Problem Transformation

The problem transformation methods transform the multi-label problem into other well-established learning scenarios, mostly into new multiple single-

label learning tasks, and these new tasks are then handled by the standard single-label classification algorithms. For example, the first-order approaches Binary Relevance (Boutell, Luo, Shen & Brown 2004), the Classifier Chains (Read, Pfahringer, Holmes & Frank 2011) which transform the task of multi-label learning into the task of binary classification, the second-order approach Calibrated Label Ranking (Fürnkranz, Hüllermeier, Mencía & Brinker 2008) which transforms the task of multi-label learning into the task of label ranking, the high-order approach AdaBoost.MH (Schapire & Singer 2000) and the Random k-labelsets (Tsoumakas, Katakis & Vlahavas 2011) which transforms the task of multi-label learning into the task of multi-class classification.

1) **Calibrated Label Ranking:** The basic idea of this algorithm is to transform the multi-label learning problem into the label ranking problem, where ranking among labels is fulfilled by techniques of pairwise comparison (Fürnkranz et al. 2008).

2) **Binary Relevance:** The basic idea of this algorithm is to decompose the multi-label learning problem into q independent binary classification problems, where each binary classification problem corresponds to a possible label in the label space (Boutell et al. 2004).

3) **AdaBoost.MH:** The basic idea of this algorithm is to transform the multi-label learning problem into one binary classification problem, where each multi-label example is mapped into q binary examples (Schapire & Singer 2000).

4) **Random k-Label sets:** The basic idea of this algorithm is to transform the multi-label learning problem into an ensemble of multi-class classification problems, where each component learner in the ensemble targets a random subset of Y' upon which a multi-class classifier is induced by the Label Powerset (LP) techniques (Tsoumakas et al. 2011).

2.4.2 Algorithm Adaptation

The algorithm adaptation methods are those methods that extend specific learning algorithms in order to handle multi-label data. For example, ML-kNN (Zhang & Zhou 2007), IBLR (Cheng & Hüllermeier 2009), AdaBoosting.MH (Schapire & Singer 2000), Rank-SVM (Elisseeff & Weston 2001), BP-MLL (Zhang & Zhou 2006) and Decision Tree (Clare & King 2001) are all state-of-the-art algorithm adaptation methods.

ML-kNN Algorithm

The k-nearest neighbor (kNN) algorithm (Cover & Hart 1967) has a long history in the data mining area for single-label classification. The kNN algorithm is an intuitive yet effective machine learning method for solving conventional classification problems. Simply to say, in kNN, a new instance is classified to the most common class by a majority vote of its k nearest neighbor instances. If integrated with a problem transformation method, it is easy to adapt the kNN algorithm for multi-label classification.

A number of multi-label learning methods are adapted from kNN (Brinker & Hüllermeier 2007, Spyromitros, Tsoumakas & Vlahavas 2008, Wieczorkowska, Synak & Raś 2006, Zhang & Zhou 2007). ML-kNN, as the first multi-label lazy learning approach, is based on the traditional kNN algorithm and the maximum a posterior (MAP) principle (Zhang & Zhou 2007). ML-kNN first finds out the nearest neighbors of the new instance, and then based on statistical information from the neighboring instance, maximum a posterior principle is applied to determine the label set for the new instance. The principle for the approach is that an instance's labels depend on the number of neighbors that possess identical labels.

For a test instance x, assume $N(x)$ represent the set of its k nearest neighbors in dataset D. For the j-th class label, ML-kNN chooses to calculate the following statistics:

$$C_j = \sum_{(x^*,Y^*)\in N(x)} [y_j \in Y^*] \tag{2.19}$$

where C_j records the number of x's neighbors with label y_j.

Let H_j denote that x has label y_j, and $P(H_j|C_j)$ represents the posterior probability that H_j holds under the condition that x has exactly C_j neighbors with label y_j. Correspondingly, $P(\neg H_j|C_j)$ represents the posterior probability that H_j does not hold under the same condition. According to the MAP rule, the predicted label set is determined by deciding whether $P(H_j|C_j)$ is greater than $P(\neg H_j|C_j)$ or not:

$$Y = \{y_j \qquad \text{if } \frac{P(H_j|C_j)}{P(\neg H_j|C_j)} > 1 \text{ and } 1 \leq j \leq q\} \tag{2.20}$$

Based on Bayes theorem, we have:

$$\frac{P(H_j|C_j)}{P(\neg H_j|C_j)} = \frac{P(H_j) \cdot P(C_j|H_j)}{P(\neg H_j) \cdot P(C_j|\neg H_j)} \tag{2.21}$$

Here, $P(H_j)$ represents the prior probability that H_j holds, while $P(\neg H_j)$ represents the prior probability that H_j does not hold. Furthermore, $P(C_j|H_j)$ represents the likelihood that x has exactly C_j neighbors with label y_j when H_j holds, while $P(C_j|\neg H_j)$ represents the likelihood that x has exactly C_j neighbors with label y_j when H_j does not hold. As shown in Eqs. 2.20 and Eqs. 2.21, it suffices to estimate the prior probabilities as well as likelihoods for making predictions.

ML-kNN fulfills the above task via the frequency counting strategy. Firstly, the prior probabilities are estimated by counting the number training examples associated with each label:

$$P(H_j) = \frac{s + \sum_{i=1}^{m}[y_j \in Y_i]}{s \times 2 + m} (1 \leq j \leq q) \tag{2.22}$$

where s is a smoothing parameter controlling the effect of uniform prior on the estimation. If $s = 1$, it results in Laplace smoothing. And accordingly, we have $P(\neg H_j) = 1 - P(H_j)$.

Secondly, the estimation process for likelihoods is somewhat involved. For the j-th class label y_j, ML-kNN maintains two frequency arrays k_j and k'_j each containing *k+1* elements:

$$\begin{cases} k_j[r] = \sum_{i=1}^{m}[y_j \in Y_i] \cdot [\delta_j(x_i) = r], & 0 \leq r \leq k \\ k'_j[r] = \sum_{i=1}^{m}[y_j \notin Y_i] \cdot [\delta_j(x_i) = r], & 0 \leq r \leq k \end{cases} \tag{2.23}$$

where $\delta_j(x_i) = \sum_{(x^*,Y^*)\in N(x_i)}[y_j \in Y^*]$

Analogous to C_j in Eq.(2.19), $\delta_j(x_i)$ records the number of x_i's neighbors with label y_j. Therefore, $k_j[r]$ counts the number of training examples which have label y_j and have exactly r neighbors with label y_j, while $k'_j[r]$ counts the number of training examples which don't have label y_j and have exactly r neighbors with label y_j. Afterwards, the likelihoods can be estimated based on elements in $k_j[r]$ and $k'_j[r]$:

$$\begin{cases} P(C_j|H_j) = \frac{s+k_j[C_j]}{s\times(k+1)+\sum_{r=0}^{k} k_j[r]} & 1 \le j \le q, 0 \le C_j \le k \\ P(C_j|\neg H_j) = \frac{s+k'_j[C_j]}{s\times(k+1)+\sum_{r=0}^{k} k'_j[r]} & 1 \le j \le q, 0 \le C_j \le k \end{cases} \tag{2.24}$$

Thereafter, by substituting Eq.(2.22) (prior probabilities) and Eq.(2.24) (likelihoods) into Eq.(2.21), the predicted label set in Eq.(2.20) naturally follows.

ML-kNN is widely used in multi-label classification and can sometimes outperform state-of-the-art approaches (Cheng & Hüllermeier 2009, Zhang & Zhou 2007). ML-kNN has two inheriting merits from both lazy learning and MAP principle: one is the decision boundary can be adaptively adjusted due to the varying neighbors identified for each new instance, and another one is that the class-imbalance issue can be largely mitigated due to the prior probabilities estimated for each class label. However, ML-kNN is actually a binary relevance classifier for it learns a single classifier h_i for each label independently. In other words, it does not consider the correlations between different labels. That drawback may cause problems in some cases.

2.4.3 Evaluation

Multi-label classification requires different metrics compared to those used in traditional single-label classification. A lot of criteria has been proposed for evaluating the performance of multi-label classification algorithms (Tsoumakas & Katakis 2007). Assume D is a multi-label dataset, and consists of $|D|$ multi-label examples (x_i, L_i), $i = 1..|D|$, $L_i \in L$. Let h be a multi-label classifier and $Y_i = h(x_i)$ be the set of labels predicted by h for example x_i.

There are three popular evaluation criteria which we adopted in this thesis for multi-label classification: the Hamming Loss, the One Error and the Average Precision (Schapire & Singer 2000). The definitions of these criteria are described as follows:

• The Hamming Loss: the fraction of the wrong labels to the total number of labels:

$$\text{HammingLoss}(h, D) = \frac{1}{|D|}\sum_{i=1}^{|D|}\frac{|L_i \Delta Y_i|}{|L|}, \tag{2.25}$$

where Δ stands for the symmetric difference of two sets and corresponds to the XOR operation in Boolean logic. It is a loss function, so the optimal value is zero.

• One error: computes how many times the top-ranked label is not relevant. The smaller the metric value, the better the classifier's performance:

$$\text{OneError}(f) = \frac{1}{|D|}\sum_{i=1}^{|D|}[|argmax_{\lambda \in L} f(x_i, \lambda) \notin Y_i|], \tag{2.26}$$

Note that, for single-label classification problems, the one-error is identical to ordinary error.

• Average precision: evaluates the average fraction of relevant labels ranked higher than a particular label $y \in Y_i$. The higher the value, the better performance:

$$\text{AvgPrec} = \frac{1}{|D|}(\frac{1}{|Y_i|}\sum_{y \in Y_i}\frac{|\{y'|rank_f(x, y') \leq rank_f(x_i, y), y' \in Y_i\}|}{rank_f(x_i, y)}), \tag{2.27}$$

The value is between 0 and 1.

2.4.4 Summary

Multi-label classification is an important issue in supervised learning. This type of classification is usually performed by transforming the multi-label data into single-label data or modifying the existing algorithms to adapt the

multi-label situation, such as the simple and powerful ML-kNN. But as ML-kNN uses the popular Binary Relevance (BR) strategy (Vembu & Gärtner 2011), which may artificially introduce outliers, and then tends to degrade the performance of the classifiers. Another problem exists in the nearest neighbors finding process. The ML-kNN and it's variants rarely consider the influence comes from other labels when counting the given number of nearest neighbors with a certain label. It is highly possible that only a few samples are available.

Do the real neighbors only have these few? Does there exist any connection between different labels? If exists, how to express such kind of connection? Will such connection bring improvement or cause degeneration? These all are issues we want to explore thoroughly in this thesis.

2.5 Summary

In current supervised learning, no matter for numerical or for categorical attributes, researches used to calculate the relations based on the hypothesis that the data are "Independent and identically distributed", which is called *IID* data. However in real world data, such as social media data, sales transaction data and financial-market data, we can find so many data that breaks the rule. When handling these data, the traditional hypothesis will limit the performance of classifiers, and we have to make up a new strategy to reveal the inner relationship inside the data. Not only value to value, but the relationship between two values come from two different features, the relationship between two values come from class labels and features, are all the key issues we take into account in this thesis.

Chapter 3

Coupled kNN for Imbalanced Categorical Data

3.1 Overview

3.1.1 Background

Classification (Kantardzic 2011) is a widely accepted machine learning and data mining technique of great practical importance. Based on training instances whose category memberships are known, it identifies which class a new instance belongs to by building appropriate classifiers.

The majority of classic classification algorithms, e.g. kNN, Decision Tree, Bayesian Networks and SVM (Wu & Kumar 2009), has been built for class-balanced data sets, i.e., each class of the dataset includes a comparable number of instances. By contrast, the classification analysis on the class-imbalanced data sets (i.e. the number of instances in one class is dramatically different from that of the other one) has received much less attention, especially for the categorical data described by categorical features. It has been observed that such algorithms do not perform as good on imbalanced data sets as on balanced data sets. Hence, classifying class-imbalanced data emerges and attracts increasing attention in recent years.

In general, existing class-imbalanced classification methods represent two sorts of efforts, either manipulating the data distribution by over or under sampling or modifying existing methods to fit class imbalance. Although sampling-based methods show to outperform the original algorithms in most situation, they do not introduce much improvement for kNN, especially on imbalanced categorical data. This may be partially explained by the maximum-specificity induction bias of kNN in which the classification decision is made by examining the local neighborhood of query instances, and therefore the global re-sampling strategies may not have pronounced effect in the local neighborhood under examination. In addition, sampling strategies inevitably change the inherent distribution of the original data, or even worse, lose information or add noise.

Instead, several distance or similarity-based classification algorithms are proposed, such as kENN(Yuxuan & Zhang 2011) and CCW-kNN(Liu & Chawla 2011), to adapt kNN to imbalanced data. However, they are more effective on data with numerical features.

Let us take some of the UCI Breast Cancer data (Table 3.1) as an example to illustrate the problems with the existing algorithms and show the challenge of classifying class-imbalanced categorical data.

As shown in Table 3.1, eleven instances are divided into two classes with four categorical features: age, tumor-size, inv-nodes and breast-quad. The value in the brackets indicates the frequency of the corresponding feature value. It is a class-imbalanced categorical dataset, since there are only three instances in class A while eight instances in class B. Here, we use the first instance $\{u_0\}$ as the testing dataset, and the rest $\{u_i\}_{i=1}^{10}$ as the training dataset.

If we adopt one of the most popular similarity measure for categorical data, the Overlap Similarity, the similarity between different categorical values v_i and v_j is defined in Equation 3.1:

$$\text{Sim\_Overlap}(v_i, v_j) = \begin{cases} 1, & \text{if } v_i = v_j \\ 0, & \text{if } v_i \neq v_j. \end{cases} \tag{3.1}$$

The Overlap Similarity between (u_0, u_1) is equal to that of (u_0, u_4), (u_0, u_6), (u_0, u_9) and (u_0, u_{10}), all are 0.5, while less than $Sim\_Overlap(u_0, u_7)$, which is the max value (0.75).

If we adopt the Frequency-based Cosine Similarity, which is defined as

$$\text{Sim\_Cosine}(V_i, V_j) = \frac{V_i \cdot V_j}{||V_i||\,||V_j||}, \tag{3.2}$$

where the V_i and V_j is the instances' corresponding frequency vectors, then the instances u_{10}, u_1, u_6 and u_7 will be the top 4 instances which are close to u_0, while u_2 is only the seventh close instance to u_0.

Under these scenario, u_0 will be labeled as class B rather than class A if we using the nearest neighbor algorithms, because there are always more nearest neighbors labeled as class B than as class A. Therefore, the classifier based on the traditional similarity fails to correctly classify u_0 in the class-imbalanced categorical data shown in Table 3.1.

The overlap similarity between two categorical values is to assign 1 if they are identical otherwise 0 if different. Further, for two multivariate categorical data points, the similarity between them will be proportional to the number of features in which they match. This will cause problems in some situations. For example, considering a categorical dataset D, which has only two features: color and size. Color takes three possible values: red, green, blue, and size takes three values: small, medium and large. Table 3.2 shows the frequency of co-occurrences of the two features.

Based on the feature values given by dataset D, the overlap similarity between the two instances (green, small) and (green, medium) is $\frac{1}{2}$, and the overlap similarity between (blue, small) and (blue, medium) is also $\frac{1}{2}$. But the frequency distribution in Table 3.2 shows that (green, small) and (green, medium) are frequent co-occurrences, while (blue, small) and (blue, medium) are very rare co-occurrences. Hence, the overlap measure is too simplistic by just giving the equal importance to matches and mismatches, and the co-occurrence information in categorical data reflects the interaction between features and can be useful to define what makes two categorical values more

or less similar. However, such co-occurrence information hasn't been incorporated into the existing similarity metrics including the frequency-based cosine similarity.

From the above examples, we know that for the class-imbalanced categorical data classification tasks, besides the class imbalance, another key complexity which has not been catered for in existing classification algorithms like kNN is the comprehensive coupling relationships between features and classes hidden in data while computing the similarity or distance. Considering such couplings has shown (Wang et al. 2011) to be very important for capturing the non-IIDness nature in the real world data, in which objects and object properties are coupled and personalized rather than independent and identically distributed as we usually assume. This is particularly important for big data analytics of complex behavioral and social data with diverse interactions.

3.1.2 Challenges and Solutions

Classifiers based on balanced data are not suitable for class-imbalanced classification tasks, especially cannot be used on the class-imbalanced categorical data. While for the classifiers developed for class-imbalanced categorical data, there are two shortcomings for existing methods. The first one is their ignoring of the co-relationships between data features, and hence not revealing the true relation inner data. This will definitely affect the accuracy of prediction result. The second problem is the distance measurement they used in their learning process cannot capture the truly distance or similarity between the objects. As the distance measurement is the key issue of such classifiers, we should not surprise if such a less accurate distance measurement cannot find out all the nearest neighbors.

So it is much challenging but essential to classify class-imbalanced non-IID categorical data. In fact, learning from the class-imbalanced data has also been identified as one of the top 10 challenging problems in data mining research (Yang & Wu 2006). It is very needed to develop a new similar-

ity measure for the categorical data to capture the coupling relationships between imbalanced classes and between categorical features in many real world domains, such as social networks, social media, recommender systems and behavioral applications.

In this chapter, we propose a novel coupled fuzzy nearest neighbor classification algorithm, CF-kNN for short, for class-imbalanced non-IID categorical data. The CF-kNN advances the idea of classic kNN substantially by addressing both class imbalance and couplings within data.

We performed our experiments on different real-life data sets from UCI(Bache & Lichman 2013), KEEL(Alcalá et al. 2010) and even an university database. According to the results, the CF-kNN outperforms the typical kNN algorithms, including classic kNN, kENN (which finds exemplar training samples to enlarge the decision boundary for the minority class), CCW-kNN (which learns the class weight for each training sample by mixture modelling), and SMOTE based kNN (which uses SMOTE to pre-process the dataset), showing its significant advantage in handling class-imbalanced categorical data by considering couplings. The improved performance of variants kNN algorithms which use our coupling strategy indicates that our method can capture the intrinsic interactions between categorical features.

The chapter is organized as follows. Section 3.1 briefly reviews the background of this chapter. Preliminary definitions are specified in Section 3.2. Section 3.3 explains our classification algorithm for the imbalanced data sets. The experimental results are discussed in Section 3.4 and the conclusion is summarized in Section 3.5.

3.2 Preliminary Definitions

Classification of the class-imbalanced categorical data can be formally described as follows: $U = \{u_1, \cdots, u_m\}$ is a set of m instances; $F = \{a_1, \cdots, a_n\}$ is a set of n categorical features; $C = \{c_1, \cdots, c_L\}$ is a set of L classes, in which each class has dramatically different numbers of instances. The

goal is to classify an unlabeled testing instance u_t based on the instances in the training set $\{u_i\}$ with known classes. For example, Table 3.1 exhibits a class-imbalanced dataset in which the training set consists of ten objects $\{u_1, u_2, \cdots, u_{10}\}$, four categorical features {"age", "tumor-size", "inv-nodes", "breast-quad"}, and two classes $\{A, B\}$. The testing dataset is $\{u_0\}$. There are only two instances in class A, while eight instances in class B. Our target is to find a suitable classification model to categorize u_0 into class A.

In the following sections, the *size* of a class refers to the number of instances in this class. When we say a class c_l is smaller (or larger) than class c_k, it means that the size of class c_l is smaller (or larger) than that of class c_k. A minority class has a relatively small size, while a majority class has a relatively large size. In addition, $|H|$ is the number of instances in dataset H.

3.3 Coupled kNN

In this section, a coupled fuzzy kNN algorithm (i.e. CF-kNN for short) is proposed to handle the class-imbalanced categorical data classification issues.

Compared to the classic kNN, our CF-kNN consists of three components: *weight assignment*, *coupling similarity calculation*, and *integration*. At the phase of weight assignment, we introduce a fuzzy class weight to handle the class-imbalanced issue: *Class Size Weight*. This weight provides the quantification on how small a class is. Simultaneously, at the step of coupling similarity calculation, we introduce the *Adapted Coupled Nominal Similarity* following the idea in (Wang et al. 2011) to describe the closeness between two different instances by considering both intra-feature and inter-features couplings and their combination. Finally, at the stage of integration, we propose the *Integrated Similarity* to measure the similarity between the instances by merging the adapted coupled nominal similarity and fuzzy class weight. The classification result of a testing instance is determined according to the integrated pairwise similarity. Below, we specify these building blocks one by

one.

3.3.1 Weights Assignment

In this part, we propose two weights: *Class Size Weight* and *Feature Weight* to characterize the structure of imbalanced classes and to capture the hidden information from the instances.

A) Class Size Weight

In a class-imbalanced dataset, there are usually several small classes that contain much less instances (i.e. minority), while a lot more instances are in some large classes (i.e. majority). However, what exactly does a small class mean or how to quantify a small class are not so explicit. As it would be too reductive to regard the smallest class as the minority, we use a fuzzy way (Ross 2009) to measure how small a class is, and make our approach suitable for multi-class problems. Accordingly, we have:

Definition 1 *The **Class Size Weight** $\theta(\cdot)$ denotes the degree of a class c_l that belongs to the minority. Formally, $\theta(\cdot)$ is defined as:*

$$\theta(c_l) = \sqrt{2^{-\frac{|c_l|}{m}}}, \tag{3.3}$$

where $|c_l|$ is the number of instances with class c_l and m is the total number of instances in the dataset. Accordingly, we have $\theta(c_l) \in (0.707, 1)$.

The Class Size Weight describes how small a class is. In special cases, $\theta(c_l)$ reaches the maximum if c_l has the smallest number of instances; $\theta(c_l)$ will down to the minimum if c_l is the largest class. For other medium classes, the corresponding Class Size Weight fall within $(\theta(c_l)^{min}, \theta(c_l)^{max})$. When a dataset is balanced with two classes, we have $\theta(c_l) = 0.841$. In Table 3.1, for instance, we have $\theta(c_A) = 0.758$, and $\theta(c_B) = 0.933$.

Later in measuring the similarity of instances, we will incorporate the Class Size Weight $\theta(\cdot)$ into the integrated similarity measure to balance the impact of class size when finding the nearest neighbors.

B) Feature Weight

As we know, less relevant features that provide little information for classification should be assigned low weights, while features that provide more reliable information should be assigned high weights. Towards this goal, the *mutual information* (MI)(Shannon 2001) between the values of a feature and the class of the training examples can be used to assign feature weights. Formally, we have:

Definition 2 *The **feature weight** describes the importance of each categorical feature f_j:*

$$\alpha_j = \sum_{v \in V_f} \sum_{c_j \in C} p(c_j, x_f = v) \cdot \log \frac{p(c_j, x_f = v)}{p(c_j) \cdot p(x_f = v)} \tag{3.4}$$

where $p(c_j)$ is the frequency of class c_j among the training set D and $p(x_f = v)$ is the frequency of value v for f among instances in D.

This equation assigns zero to features that provide no information about the class, and a value proportional to $\log(|C|)$ to features that completely determine the class (i.e., assuming a uniform distribution on classes). For example, in Table 3.1, we have the normalized feature weights: $\alpha_1 = 0.2639, \alpha_2 = 0.1528, \alpha_3 = 0.3750$, and $\alpha_4 = 0.2083$.

3.3.2 Coupling Similarity

The similarity between instances is defined for the class-imbalanced categorical data. The usual way to deal with the similarity between two categorical instances is the cosine similarity on frequency and overlap similarity on feature value. However, as they are too rough to measure the similarity and do not consider the coupling relationships among features. Wang et al. (Wang et al. 2011) introduced a Coupled Nominal Similarity (CNS) for categorical data, which addresses both the intra-coupling similarity between values within a feature and the inter-coupling similarity among different features.

The proposed similarity measure picks up both explicit and implicit interactions between objects, features and feature values, and has been shown to outperform the SMS and the ADD(Ahmad & Dey 2007*a*) in the clustering learning. Here, we adapt the CNS and extend it to our classification algorithm as follows.

Definition 3 *Given a training dataset D, a pair of values* $v_j^x, v_j^y (v_j^x \neq v_j^y)$ *from feature* a_j. v_j^x *and* v_j^y *are defined to be intra-related in feature* a_j. *The* ***Intra-Coupled Similarity*** *(IaCS) between feature values* v_j^x *and* v_j^y *of feature* a_j *is formalized as:*

$$\delta^{Ia}(v_j^x, v_j^y) = \frac{RF(v_j^x) \cdot RF(v_j^y)}{RF(v_j^x) + RF(v_j^y) + RF(v_j^x) \cdot RF(v_j^y)}, \tag{3.5}$$

where $RF(v_j^x)$ *and* $RF(v_j^y)$ *are the occurrence frequency of values* v_j^x *and* v_j^y *in feature* a_j, *respectively.*

The Intra-Coupled Similarity reflects the interaction of two different categorical values in the same feature. The higher these similarities are, the closer such two values are. Thus, Equation (3.5) is designed to capture the value similarity in terms of occurrence times by taking into account the frequencies of categories. Besides, since $1 \leq RF(v_j^x), RF(v_j^y) \leq m$, then $\delta^{Ia} \in [1/3, m/(m+2)]$. For example, in Table 3.1, values *"left low"* and *"left up"* of feature *"breast-quad"* are observed 2 and 4 times respectively, so $\delta^{Ia}((\text{left low}), (\text{left up})) = (2*4)/(2+4+2*4) = 4/7$.

In contrast, the Inter-Coupled Similarity below is defined to capture the interaction of two different values in one feature according to the co-occurrence of other values from another feature.

Definition 4 *For a training dataset D and two different features* a_i *and* a_j $(i \neq j)$, *two feature values* $v_i^x, v_i^y (v_i^x \neq v_i^y)$ *from feature* a_i *and a feature value* v_j^z *from feature* a_j. v_i^x *and* v_i^y *are inter-related if there exists at least one pair value* (v_p^{xz}) *or* (v_p^{yz}) *that co-occurs in features* a_i *and* a_j *of instance* U_p. *The* ***Inter-Coupled Similarity*** *(IeCS) between feature values* v_i^x *and*

v_i^y of features a_i according to feature value v_j^z of a_j is formalized as:

$$\delta_{i|j}^{Ie}(v_i^x, v_i^y|v_j^z) = \frac{\min\left(F(v_p^{xz}), F(v_p^{yz})\right)}{\max(RF(v_i^x), RF(v_i^y))}, \tag{3.6}$$

where $F(v_p^{xz})$ and $F(v_p^{yz})$ are the co-occurrence frequency count function for value pair v_p^{xz} or v_p^{yz}, and $RF(v_i^x)$ and $RF(v_i^y)$ is the occurrence frequency in feature i.

Accordingly, we have $\delta_{i|j}^{Ie} \in [0, 1]$. The Inter-Coupled Similarity reflects the interaction or relationship of two categorical values from one feature but based on the connection to other feature. In Table 3.1, for example, as $\delta_{1|4}^{Ie}((40\text{-}49), (60\text{-}69)|(\text{left up})) = 0$ and $\delta_{1|4}^{Ie}((40\text{-}49), (60\text{-}69)|(\text{left low})) = 0$, while $\delta_{1|4}^{Ie}((40\text{-}49), (50\text{-}59)|(\text{left up})) = \min(1, 3)/\max(2, 6) = 1/6$, and $\delta_{1|4}^{Ie}((40\text{-}49), (50\text{-}59)|(\text{left low})) = \min(1, 3)/\max(2, 6) = 1/6$, so in feature 1 ("age"), the Inter-Coupled Similarity of value pair [(40-49),(50-59)] is much higher than the value pair [(40-49),(60-69)] based on feature 4 ("breast-quad").

By taking into account the feature weight, the *Adapted Coupled Object Similarity* between instances u_{i_1} and u_{i_2} is formalized as:

$$\begin{aligned} &AS(u_{i_1}, u_{i_2}) \\ &= \sum_{j=1}^{n}[\beta \cdot \alpha_j \delta_j^{Ia} + (1-\beta) \cdot \sum_{k=1, k\neq j}^{n} \delta_{j|k}^{Ie}] \\ &= \sum_{j=1}^{n}[\beta \cdot \alpha_j \delta_j^{Ia}(v_j^{i_1}, v_j^{i_2}) + (1-\beta) \cdot \sum_{k=1, k\neq j}^{n} \delta_{j|k}^{Ie}(v_j^{i_1}, v_j^{i_2}|v_k)], \end{aligned} \tag{3.7}$$

where $\beta \in [0, 1]$ is the parameter that decides the weight of intra-coupled similarity, $v_j^{i_1}$ and $v_j^{i_2}$ are the values of feature j for instances u_{i_1} and u_{i_2}, respectively. δ_j^{Ia} and $\delta_{j|k}^{Ie}$ are the intra-coupled feature value similarity and inter-coupled feature value similarity, respectively. It is remarkable to note that α_j is the feature weight defined in Equation (3.4), rather than $\alpha_j = 1/n$ assumed in (Yang, Cao & Zhang 2010).

3.3.3 Integration

Finally, we combine the weights with the coupling similarity, and propose an *Integrated Similarity* for classifying class-imbalanced categorical data. In this way, we are able to balance the influence of class imbalance and simultaneously take into account the inner-relationship between features. The features are not independent of each other anymore. The *Integrated Similarity* is defined as below:

Definition 5 *The **Integrated Similarity** represents the adapted coupled similarity measure by taking into account the feature weight, feature values' intra and inter coupling relationship as well as the class size information. Formally,*

$$IS(u_e, u_i) = \theta(C(u_i)) \cdot AS(u_e, u_i), \tag{3.8}$$

where u_e and u_i are the instances, respectively; $C(u_i)$ is the class of u_i; $\theta(\cdot)$ is the class size weight defined in Equation (3.3); and $AS(\cdot)$ is the adapted coupled object similarity defined in Equation (3.7).

As indicated by Equation (3.8), on one hand, the $\theta(\cdot)$ can capture the class size information, which is helpful to handle the class imbalance issues and suitable for multi-class classification tasks; On the other hand, the adapted similarity $AS(\cdot)$ includes not only the feature-class coupling information (feature weight), but it also capture the feature values' intra-feature coupling relationship and inter-feature coupling relationship based on different features. These coupling relationship reflect the inner relationship of real world data. Therefore, the similarity in our algorithm is more reasonable than that in the existing similarity calculation related algorithms for the imbalanced categorical data.

3.3.4 The CF-kNN Algorithm

As shown in Algorithm 1, the CF-kNN algorithm works as follows. Based on the idea of kNN, after obtaining the Integrated Similarity between testing

instance u_t and training set $\{u_i \in D\}$, we select k nearest neighbors to u_t from the training set that correspond to the k highest Integrated Similarity values. The class c_l which contains the most instances in the neighbors is the predicted class for u_t. For example, in Table 3.1, we have $IS(u_0, u_1) = 3.9785$, $IS(u_0, u_2) = 3.8054$ and $IS(u_0, u_7) = 3.8332$, and $\{u_1, u_2, u_7\}$ are the top three nearest neighbors to u_0. So u_0 will be categorized to its real class, namely class A (if $k = 3$).

Algorithm 3.1 : Coupled Fuzzy kNN Algorithm

Require: $U = \{u_1, \cdots, u_m\}$ is a set of m instances; $F = \{a_1, \cdots, a_n\}$ is a set of n categorical features; $C = \{c_1, \cdots, c_l\}$ is a set of l classes, in which each class has dramatically different numbers of instances; u_t is an instance without label

Ensure: The class label of u_t

1: **for** i=1 **to** l
2: $\theta(c_i) \leftarrow$ class size weight
3: **end**
4: **for** i=1 **to** n
5: $\alpha_i \leftarrow$ feature weight
6: **end**
7: **for** i=1 **to** m
8: $\delta_i^{Ia} \leftarrow$ Intra-feature Coupling
9: **for** j=1 **to** n
10: $\delta_{i|j}^{Ie} \leftarrow$ Inter-feature Coupling
11: **end**
12: $AS \leftarrow$ Adapted Coupled Object Similarity
13: $IS \leftarrow$ Integerated Similarity
14: **end**
15: Select k nearest neighbors to u_t
16: **Return** the most frequent class label in those k neighbors

3.4 Experiments and Evaluation

3.4.1 Data and Experimental Settings

To verify the performance of our algorithm, we choose some commonly used UCI, KEEL data sets and a real world dataset. Our motivation is that if an algorithm can show improvement on such data compared to the baselines, it has potential to differentiate itself from others in more complex data with strong coupling relationship. In total, 14 data sets are taken from the UCI Data Repository (Bache & Lichman 2013), KEEL data set repository (Alcalá et al. 2010), and the real Student Learning data taken from the records of an Australian university's database (If a student failed both in course L and course S, he or she will be labeled as "Positive", or else be labeled as "Negative"). In experiment 3, we use SMOTE on this student dataset and created 50 new data sets with minority class varies from 1% to 50%. A short description of all the data sets is provided in Table 3.3 and the proportion of minority class to the total instances is shown as *Minority(%)*. These data sets have been selected as they typically have an imbalance class distribution (the biggest Imbalance Rate is one is 101.04). For some data sets that have mixed type of features, such as D1, D2, D4 and D5, we conducted the CAIM discretization algorithm (Kurgan & Cios 2004) on numerical features first so as to convert them into categorical ones.

We conducted 10-fold cross validation 3-times to evaluate the performance of all the algorithms. In the experiments, we select not only variants of kNN, such as the classic k Nearest Neighbors(kNN)(Wu & Kumar 2009), kENN(Yuxuan & Zhang 2011), CCW-kNN(Liu & Chawla 2011) and SMOTE based kNN to compare with, but also the very popular classifiers C4.5 and NaiveBayes. To make algorithms more comparable, we further incorporate our coupling strategy into some kNN algorithms (the new ones are with a prefix of "$CF+$") to compare their performance. In all experiments, we set $k = 5$ to all those kNN-based classifiers, and the confidence levels for kENN is set to 0.1.

Due to the dominative effect of the majority class, the overall accuracy is not an appropriate evaluation measure for the performance of classifiers on imbalanced data sets, so we use the Receiver Operating Characteristic (ROC) curve and the Area Under the ROC Curve (AUC)(Fawcett 2006) to evaluate the performance. AUC indicates the overall classification performance. From the definition we know that the AUC of a perfect classifier equals to 1 while a bad one is less than 0.5. The higher AUC is, the better average performance the classifier is. So a good classification algorithm will has a higher AUC.

3.4.2 The Performance of CF-kNN

Table 3.4 shows the AUC results of our CF-kNN comparing to some of the state of the art algorithms. The top two results are highlighted in bold. Comparing to other approaches, our CF-kNN achieved the highest AUC and outperforms others in most of the data sets, especially in data sets with high imbalance rate. Also, our proposed CF-kNN always outperforms the classic kNN on all the data sets. This proves that considering the coupling relationships between features and classes by treating the data as non-IID when computing similarity or distance can capture the intrinsic data characteristics. Note that the SMOTE-based kNN does not always demonstrate significant improvement compared with kNN, sometimes even worse, such as in dataset D5 and D14. It means that only using SMOTE on imbalanced categorical data may not always bring improvement but sometimes noise.

From these results we can see that when the imbalance rates are higher than 11.5, our method can achieve a much better improvement (the least one is 2.08% and the highest one is 12.09%) on these data which does not incorporate much non-IIDness characteristics. On some specific data sets, such as D8, our methods also approach as good as CCW-kNN. That confirms again that our coupled fuzzy strategy is very effective for imbalanced non-IID classification tasks.

3.4.3 The Effect of Incorporating Couplings

This set of experiments aims to test the effect of incorporating the coupling similarity into other classification algorithms. For doing this, we created three comparison sets by integrating the proposed coupled fuzzy mechanism into kENN to form CF+kENN, CCWkNN to form CF+CCWkNN, and SMOTE based kNN to form CF+SMOTE based kNN, and compared their performance. All comparable algorithms are with the same parameter settings.

Table 3.5 shows the result of performance of these comparable algorithms with vs. without the coupling strategy. It shows that incorporating our new similarity metrics will get more or less improvement for those distance or similarity-based algorithms. More specifically, for kENN, the improvement is between 2.09% (on D4) and 6.60% (on D1), except for the data set D10; for CCWkNN, the improvement is vary from 1.25% (on D10) and 6.09% (on D1), except for the data set D12; while for SMOTE based kNN, it can get a 6.83% improvement on data set D3. This further verified the effectiveness of our supposed coupling strategy and can capture the intrinsic characteristics better than the existing methods, and especially suitable for class-imbalanced categorical data.

3.4.4 The Sensitivity to Imbalance Rate

To evaluate our coupled similarity on different imbalance rate, we do SMOTE on student data and create 50 new data sets, in which the minority class varies from 1% to 50% of the total instances. Fig. 3.1 shows the improvement of the basic algorithms which combined with our Coupled Fuzzy Similarity on different imbalance rate. As it shows in the figure, when minority class only takes up 10% or less of the total instances, both kNN and kENN (combined with coupling) can have an improvement of over 5.821%. Even for CCWkNN, the improvement can over 5.372%. But with the imbalance rate declining, this improvement falls simultaneously. When minority class comes to 35%

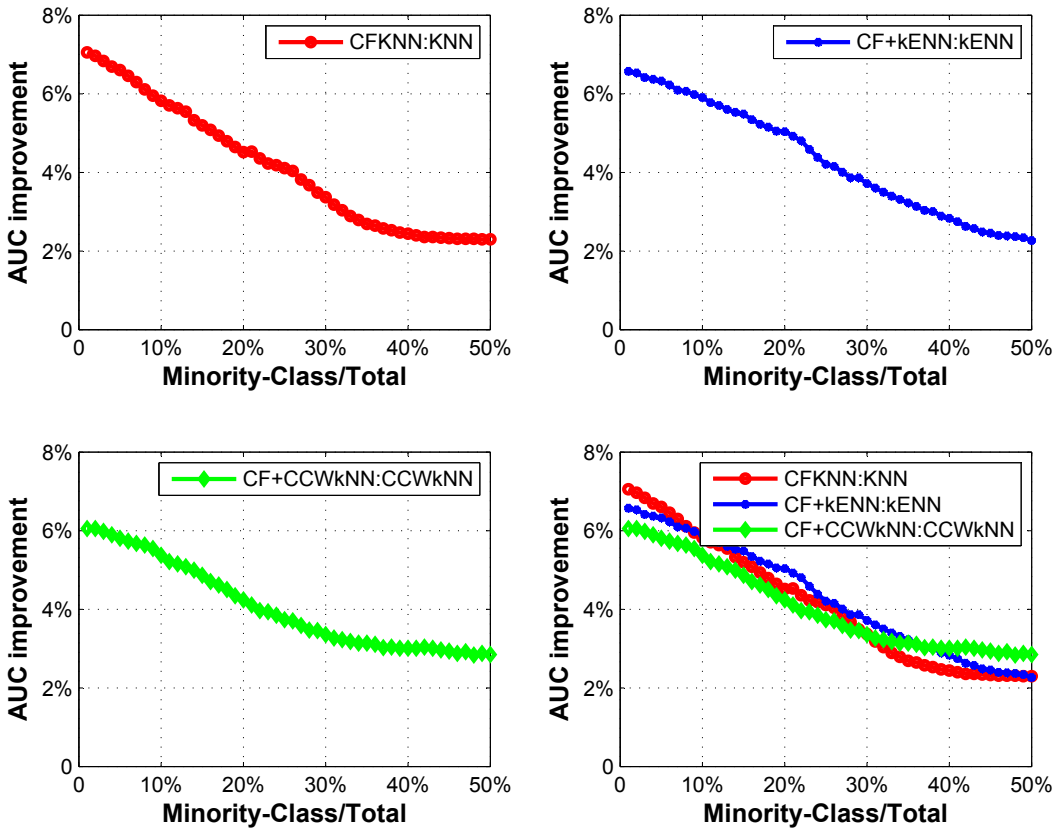

Figure 3.1: The Sensitivity of Coupling to Imbalance Rate

of the total records (which can be defined as "balanced" data) or over, the improvement will not be so outstanding and stay stable at about 2.2%. This experiment demonstrates that our strategy is sensitive to the imbalance rate, and it is more suitable for being used in the scenario with high imbalance rate, that is, imbalanced categorical Non-IID data.

3.5 Summary

Traditional classifiers mainly focus on dealing with balanced dataset and overlook the coupling relationship between attributes and classes. Classifying imbalanced categorical data is very challenging. We proposed a coupled fuzzy kNN to classify the class imbalanced categorical data which has strong relationships between attributes and classes. It incorporates the class size weight with feature weight into a coupling similarity measure, which effectively extracts the inter-feature coupling and intra-feature coupling relationships in categorical data. The experiment results show that our CF-kNN has a more stable and higher average performance than the classic kNN, kENN, CCWkNN, SMOTE-based kNN, Decision Tree and NaiveBayes do when applied on the class-imbalanced categorical data classification tasks.

[***Note***] A conference version (Liu, Cao & Yu 2014*a*) of this chapter has been accepted already and be published by IJCNN2014 as below:

- **Chunming Liu**, Longbing Cao, Philip S Yu (2014), Coupled fuzzy k-nearest neighbors classification of imbalanced non-IID categorical data. *in* 'Proceedings of the Neural Networks (IJCNN), 2014 International Joint Conference on (**WCCI14**)', IEEE, pp. 1122-1129. (**ERA ranking: A**)

Table 3.1: An Example from The UCI Dataset: Breast Cancer Data

ID	**age**	**tumor-size**	**inv-nodes**	**breast-quad**	**Class**	**Overlap Sim**	**Cosine Sim**
u_0	50-59	35-39	0-2	left low	A		
u_1	50-59 (6)	25-29 (2)	0-2 (8)	right up (1)	A	0.5	0.9905
u_2	60-69 (1)	30-34 (2)	0-2 (8)	central (2)	A	0.25	0.8681
u_3	40-49 (2)	25-29 (2)	0-2 (8)	left up (4)	B	0.25	0.8947
u_4	50-59 (6)	30-34 (2)	6-8 (1)	left low (2)	B	0.5	0.7274
u_5	30-39 (1)	10-14 (3)	0-2 (8)	right low (1)	B	0.25	0.8452
u_6	50-59 (6)	50-54 (1)	0-2 (8)	left up (4)	B	0.5	0.9834
u_7	50-59 (6)	35-39 (1)	0-2 (8)	left up (4)	B	0.75	0.9834
u_8	50-59 (6)	10-14 (3)	3-5 (1)	left up (4)	B	0.25	0.6817
u_9	40-49 (2)	10-14 (3)	0-2 (8)	left low (2)	B	0.5	0.9000
u_{10}	50-59 (6)	15-19 (1)	0-2 (8)	central (2)	B	0.5	1.0000

Table 3.2: An Example of Frequency of Feature Co-occurrences

	small	***medium***	***large***	**Total**
red	44	47	9	100
green	48	45	7	100
blue	8	8	84	100
Total	100	100	100	

Table 3.3: Data Sets Used in Experiment

Index	**Dataset**	**Source**	**#Instances**	**#Attribute**	**#Class**	**Minority Name**	**Minority(%)**
D1	Students	REAL	50000	32	2	Positive	0.98%
D2	kr-vs-k	KEEL	28056	6	18	five	1.68%
D3	Abalone	UCI	4177	8	29	Class15	2.47%
D4	Nursery	UCI	12960	8	5	very recom	2.53%
D5	Dermatology	UCI	366	34	6	P.R.P.	5.46%
D6	Zoo	UCI	101	17	7	Set6	7.92%
D7	Solar Flare	KEEL	1066	11	6	E	8.91%
D8	Connect-4	UCI	67557	42	3	draw	9.55%
D9	Primary Tumor	UCI	339	17	22	stomach	11.50%
D10	Soybean(Large)	UCI	307	35	19	brown-spot	13.03%
D11	Hayes-roth	UCI	160	5	3	3	19.38%
D12	Contraceptive	UCI	1473	9	3	Long-term	22.61%
D13	Adult	UCI	45222	14	2	>50K	23.93%
D14	Splice-junction	KEEL	3190	60	3	EI	24.04%

Table 3.4: The AUC Results for CF-kNN in Comparison with Other Algorithms

Dataset	Minority(%)	**CF-kNN**	kNN	kENN	CCWkNN	SMOTE	C4.5	Naive	**improvement**
D1	0.98%	**0.909**	0.845	0.849	0.854	**0.866**	0.857	0.857	4.97%-7.59%
D2	1.68%	**0.711**	0.661	0.672	**0.685**	0.682	0.669	0.669	3.87%-7.49%
D3	2.47%	**0.718**	0.672	0.680	**0.692**	0.688	0.683	0.682	3.75%-6.89%
D4	2.53%	**0.981**	0.922	**0.959**	0.948	0.933	0.958	0.934	2.35%-6.38%
D5	5.46%	**0.76**	0.715	0.720	**0.729**	0.678	0.716	0.724	4.28%-12.09%
D6	7.92%	**0.887**	0.842	**0.869**	**0.869**	0.854	0.857	0.859	2.08%-5.30%
D7	8.91%	**0.962**	0.910	0.920	0.937	0.930	**0.947**	0.925	1.62%-5.67%
D8	9.55%	**0.916**	0.864	0.876	**0.916**	0.910	0.910	0.888	0.00%-6.02%
D9	11.50%	**0.716**	0.685	0.701	0.695	0.701	0.698	**0.705**	1.60%-4.59%
D10	13.03%	**0.971**	0.932	0.957	0.961	**0.961**	0.942	0.954	1.01%-4.16%
D11	19.38%	**0.972**	0.932	0.943	**0.960**	0.942	0.959	0.952	1.26%-4.34%
D12	22.61%	**0.755**	0.718	0.729	0.725	**0.743**	0.726	0.736	1.64%-5.12%
D13	23.93%	**0.938**	0.904	0.915	0.910	0.910	**0.920**	0.919	1.95%-3.79%
D14	24.04%	**0.977**	0.938	0.940	0.947	0.907	**0.964**	0.953	1.36%-7.72%

Table 3.5: The Comparison of With and Without Coupling

Dataset	Minority(%)	kENN	CF+kENN	CCWkNN	CF+CCWkNN	SMOTE	CF+SMOTE
D1	0.98%	0.849	**0.905**	0.854	**0.906**	0.866	**0.922**
D2	1.68%	0.672	**0.715**	0.685	**0.726**	0.682	**0.725**
D3	2.47%	0.680	**0.724**	0.692	**0.733**	0.688	**0.735**
D4	2.53%	0.959	**0.979**	0.948	**0.967**	0.933	**0.990**
D5	5.46%	0.720	**0.766**	0.729	**0.771**	0.678	**0.718**
D6	7.92%	0.869	**0.922**	0.869	**0.918**	0.854	**0.908**
D7	8.91%	0.920	**0.975**	0.937	**0.989**	0.930	**0.985**
D8	9.55%	0.876	**0.928**	0.916	**0.965**	0.910	**0.963**
D9	11.50%	0.701	**0.742**	0.695	**0.732**	0.701	**0.741**
D10	13.03%	**0.957**	**0.957**	0.961	**0.973**	0.961	**0.975**
D11	19.38%	0.943	**0.990**	0.960	**0.974**	0.942	**0.995**
D12	22.61%	0.729	**0.764**	**0.725**	**0.725**	0.743	**0.776**
D13	23.93%	0.915	**0.957**	0.910	**0.946**	0.910	**0.951**
D14	24.04%	0.940	**0.981**	0.947	**0.984**	0.907	**0.947**

Chapter 4

Coupling Based Classification for Numerical Data

4.1 Overview

4.1.1 Background

In classification tasks, the data set with continuous features is one of the most important data types. Such data exists in diverse domains, such as financing, marketing, medicine and geography, etc. People usually use the information table (M. Kaytoue & Napoli 2011) to represent such continuous data. In an information table, it is comprised by columns and rows, which is named as "attributes" (or "features") and "objects", respectively. The table cell is corresponding to an attribute value of some object.

If we taking the fragment data of Wine in UCI (Table 4.1) as an example, nine kind of wine objects are characterized by four numerical attributes (i.e. "Malic acid", "Ash", "Total phenols" and "Proanthocyanins") and divided into three classes(i.e. "C_1","C_2" and "C_3"). For instance, the "Total phenols" of wine object u_2 is 2.35, and the "Malic acid" of u_2 is 1.8. Based on this kind of representation, a quantity of data mining techniques and machine learning tasks (Plant 2012) have been performed. In these approaches, one

of the critical parts in such applications is to study the distance between different objects. As we know, for numerical data, a lot of distance metrics have been developed. The most frequently used distance metrics are known as the Euclidean and Minkowski metrics (Gan et al. 2007).

Given two points $P = (x_1, x_2, \cdots, x_n)$ and $Q = (y_1, y_2, \cdots, y_n) \in \mathbb{R}^n$, the Euclidean distance between P and Q is defined as:

$$\begin{aligned} d &= \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \cdots + (x_n - y_n)^2} \\ &= \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \end{aligned} \tag{4.1}$$

The Minkowski distance is a metric on Euclidean space. The Minkowski distance of order p between two points P and Q is defined as:

$$d_p = (\sum_{i=1}^{n} |x_i - y_i|^p)^{1/p} \tag{4.2}$$

Minkowski distance is typically used with $p = 1$ or $p = 2$. The latter is the Euclidean distance, while the former is known as the Manhattan distance.

In Table 4.1, since wine objects u_4 and u_8 have identical values of " Malic acid" and "Proanthocyanins", the normalized Euclidean distance between them is only 0.093 , which is much smaller than that between u_4, u_5 (i.e. 1.334) and also much smaller than that between u_4, u_6 (i.e. 0.182). From the Euclidean distance, we know that u_8, u_7 and u_2 are the nearest three points to u_4. It indicates that u_4 have a good chance to be classified into the same class of u_8 if we apply the Euclidean distance based nearest neighbor classifier with $k = 3$. However, in fact, u_4 belongs to "C_2" while u_8 is labeled as "C_3".

This example indicates that it is often problematic to analyze the numerical data by assuming all the attributes are independent to each other. The reason is that the traditional data representation schemes fail to capture the real inner relationship between attributes. It is very needed to develop an effective representation method to analyze numerical variables by considering the relationships among attributes.

4.1.2 Challenges and Solutions

For numerical data learning, when calculating the distance or similarity between continuous values, the Euclidean and Minkowski distance (Gan et al. 2007) are the widely used distance measures. But the definition of both Euclidean distance and Minkowski distance restrict the calculation in a specific attribute. These two definitions only consider the local distance while ignore the influence comes from other objects or features. These two distance measures fail to capture the genuine relationship between numerical values.

An increasing number of researchers point out that the independence assumption on attributes often leads to a mass of information loss, and several papers have addressed the issue of attribute interactions. Pearson's correlation coefficient (Gan et al. 2007) can measure the agreement of shapes between variables, but it is only suitable for the linear relationship between two variables. Jakulin and Bratko analyzed the attribute dependency by information gain (Jakulin & Bratko 2003). A rank-correlated measure (Calders, Goethals & Jaroszewicz 2006) has been proposed to mine frequent patterns, but it only considers the pair wise relationship in a local way and works on non intuitive ranks rather than attribute values. More recently, Wang et al. put forward the coupled nominal similarity in unsupervised learning (Wang et al. 2011), but its for categorical data. Plant presented the dependency clustering by mapping objects and attributes in a cluster-specific low-dimension space (Plant 2012), however, the interaction mechanism is embedded in the modeling process of clustering and not explicitly defined. Wang et al. (Wang, She & Cao 2013) proposed a framework of the coupled attribute analysis to capture the global dependency of continuous attributes, but it don't give out an explicit evaluation metric to measure the proportion of interaction comes from other attributes.

So how to develop an effective representation measure for analyzing the similarity between continuous variables and make it more accurate to reality, is a challenge for numerical featured supervised learning. Based on the

information table, we try to describe the inner relationship with the least information loss. Accordingly, we propose a coupled attributes analysis on numerical features to address the aforementioned research issues in this chapter.

The chapter is organized as follows. In Section 4.1, we briefly introduce the work related. Section 4.2 gives a detailed description of the new coupling concept we proposed. The experimental results are discussed in Section 4.3. Finally, the conclusion is discussed in Section 4.4.

4.2 Coupling Relationship on Numerical Attributes

4.2.1 Problem Statement

We formally define the classification problem on numerical data as this: Let X denotes the space of instances and $C = \{c_1, \ldots, c_n\}$ denotes the whole class set where $|C| = n$. $T = \{(x_1, C(x_1)), \ldots, (x_m, C(x_m))\}$ ($|T| = m$) is the training data set, whose instances are drawn identically and independently from the D. Each instance $x \in X$ is associated with a class $c_x \in C$. The goal of our classification is to get a classifier $h : X \to C$ that maps a feature vector to a class, while optimizing some specific evaluation metrics. For example, the information Table 4.1 composes of nine objects $\{u_1, u_2, \cdots, u_9\}$ and four attributes $\{a_1, a_2, a_3, a_4\}$, all the instances are divided into 3 classes $\{c_1, c_2, c_3\}$.

In the next part of this paper, we call the relationships within a numerical attribute as "intra-coupled" relationship while the relationships among different numerical attributes as "inter-coupled" relationship. Our target is to capture these two coupling relationships.

4.2.2 Data Discretization

In order to apply our strategy which compute the similarity between numerical features and categorical features, we do discretization on numerical attributes to transfer such continuous values into separate groups. As we are conducting the supervised classification tasks, we choose CAIM (class-attribute interdependence maximization) discretization algorithm (Kurgan & Cios 2004) which can capture the class-attribute interdependency information as our discretization method.

The CAIM uses class-attribute interdependency information as the criterion for the optimal discretization. For a given quanta matrix, the CAIM criterion measures the dependency between the class variable C and the discretization variable D for attribute F. It is defined as:

$$\text{CAIM}(C, D|F) = \frac{\sum_{r=1}^{n}(max_r^2/M_{+r})}{n}, \tag{4.3}$$

where n is the number of intervals, r iterates through all intervals, and max_r is the maximum value within the rth column of the quanta matrix, M_{+r} is the total number of continuous values of attribute F that are within the interval $(d_{r-1}, d_r]$.

The algorithm starts with a single interval that covers all possible values of a continuous attribute, and divides it iteratively. From all possible division points that are tried it chooses the division boundary that gives the highest value of the CAIM criterion.

The result we got from this discretization on numerical attributes is used in the following coupling relationship calculation stage. As the numerical value is continuous, we cannot apply our strategy on two numerical values comes from different attributes directly. So we use the discretization interval groups as the categories of the continuous values, then we can evaluate the similarity between these groups. For example, The Table 4.1 can then be transferred into a new group table, as shown in Table 4.2.

4.2.3 Similarity Calculation

In this part, the similarity between instances is defined for the numerical data. We use the Euclidean distance as the base distance in our algorithm, and the intra-coupling and the inter-coupling are the extension from the base distance. We use $Similarity = \frac{1}{1+distance}$ as the function between distance and similarity.

Definition 6 *Given a discretized training data set D, a pair of value groups $v_j^x, v_j^y (v_j^x \neq v_j^y)$ of feature a_j. v_j^x and v_j^y are defined to be intra-related in feature a_j. The **Intra Coupled Relationship** between group values v_j^x and v_j^y of feature a_j is formalized as:*

$$Ra^{Intra}(v_j^x, v_j^y) = \frac{F(v_j^x) \cdot F(v_j^y)}{F(v_j^x) + F(v_j^y) + F(v_j^x) \cdot F(v_j^y)}, \tag{4.4}$$

where $F(v_j^x)$ and $F(v_j^y)$ are the occurrence frequency of value group v_j^x and v_j^y in feature a_j, respectively.

The Intra Coupled Relationship just reflects the interaction of two value groups in the same feature. The higher these frequencies are, the closer such two groups are. Thus, Equation (4.4) is designed to capture the group similarity in terms of occurrence times by taking into account the frequencies of categories. Besides, since $1 \leq F(v_j^x), F(v_j^y) \leq m$, then $Ra^{Intra} \in [1/3, m/(m+2)]$. For example, in Table 4.2, groups "g_1^1" and "g_1^2" of feature $Malicacid$ are observed four and two times, so $Ra^{Intra}((g_1^1), (g_1^2)) = (4*2)/(4+2+4*2) = 4/7$.

In contrast, the Inter Coupled Relationship below is defined to capture the interaction of two groups from one same attribute but based on the connection to another group from another attribute.

Definition 7 *Given a discretized training data set D and two different features a_i and a_j $(i \neq j)$, two feature group pairs $v_i^x, v_j^y (i \neq j)$ from features a_i and a_j, respectively. v_i^x and v_j^y are defined to be inter-related if there exists at least one pair group value (v_p^{xy}) that co-occurs in features a_i and a_j of*

*instance U_p. The **Inter Coupled Relationship** between feature groups v_i^x and v_j^y of feature a_i and a_j is formalized as:*

$$Ra_{ij|k}^{Inter}(v_i^x, v_j^y) = \frac{\min\left(FC(v_p^{zx}), FC(v_p^{zy})\right)}{\max(F(v_i^x), F(v_j^y))}, \tag{4.5}$$

where $FC(v_p^{zx})$ and $FC(v_p^{zy})$ are the co-occurrence frequency count function for discretized value group pair v_p^{zx} or v_p^{zy}, and $F(v_i^x)$ and $F(v_j^y)$ is the occurrence frequency of related value group in attribute a_i. v_p^z is the discretized value group in attribute a_k.

Accordingly, we have $Ra_{i|j}^{Inter} \in [0, 1]$. The Inter-Coupled Relationship reflects the interaction or relationship of two groups from one same attribute but based on the connection to another group from another attribute. In Table 4.2, for example, as $Ra_{1|2}^{Inter}((g_1^1), (g_1^2)|(g_2^3)) = min(2,1)/max(4,2) = 0.25 < Ra_{1|2}^{Inter}((g_1^3), (g_1^2)|(g_2^1)) = min(1,1)/max(3,2) = 0.33$, so based on attribute a_2, the group pair $[(g_1^3), (g_1^2)]$ is more close to each other than the group pair $[(g_1^1), (g_1^2)]$.

4.2.4 Weight of Coupling

In order to evaluate the degree of coupling, we proposed two kinds of coupling weights on attribute and attribute-pair to evaluate the intra-coupling and inter-coupling, respectively. The bigger the weight, the strong connection there is.

Definition 8 *The **Intra-coupling weight** describes the importance degree of each discretized numerical feature a_j according to its group value distribution consistency with the distribution of classes. Formally, we have:*

$$\alpha_j = \begin{cases} \sqrt{\left(\sum_{i=1}^{m} \frac{Fre(x_{ij}, R^{C(u_i)})}{m \cdot |R^{C(u_i)}|}\right)} & if \quad |Uni(f_j)| > 1 \\ 0 & if \quad |Uni(f_j)| = 1 \end{cases} \tag{4.6}$$

where m is the total number of instances in the data set, x_{ij} is the group value of jth feature for instance u_i, $R^{C(u_i)}$ consists of all the instances which share the same class as instance u_i, and the according instance number is $|R^{C(u_i)}|$, while $Fre(x_{ij}, R^{C(u_i)})$ is a function that count the occurrences of x_{ij} in feature j of set $R^{C(u_i)}$, and $|Uni(f_j)|$ returns the discretization interval number in feature j.

The weight α_j indicates the distribution matching degree of the values in a feature to the class labels. For example, if a training data set has 6 instances with class labels of $\{C_1, C_1, C_2, C_2, C_2, C_1\}$ respectively, and feature f_1 has values of $\{A, A, B, B, B, A\}$ while feature f_2 has values of $\{M, M, M, N, N, N\}$, then the value distribution of feature f_1 is more consistent with the distribution of classes than f_2 does. The more consistent in distribution for the feature values to the class labels, the more important the feature is. If all the group values in a feature are the same, that is, $|Uni(f_j)| = 1$, then it means the intra-coupling is no use for the classification tasks, so we set the weight to be zero.

Accordingly, we can got the weight to evaluate the importance of our inter-coupling relationship:

Definition 9 *The **Inter-coupling weight** describes the importance degree of each discretized numerical feature pair (f_j, f_k) $(j \neq k)$ according to their group value pairs' distribution consistency with the distribution of classes. This weight measures the influence comes from another attribute. Formally, we have:*

$$\beta_{jk} = \begin{cases} \sum_{i=1}^{m} \sqrt{(\frac{Fre(x_i^{jk}, R^{C(u_i)})}{m \cdot |R^{C(u_i)}|})} & |Uni(f_{jk})| > 1 \\ 0 & |Uni(f_{jk})| = 1 \end{cases} \tag{4.7}$$

where m is the total number of instances in the data set, x_i^{jk} is the group value pair of the jth feature and the kth feature for instance u_i, $R^{C(u_i)}$ consists of all the instances which share the same class as instance u_i, and the according

instance number is $|R^{C(u_i)}|$*, while* $Fre(x_i^{jk}, R^{C(u_i)})$ *count the occurrences of group value pair* x_i^{jk} *in feature* j *and feature* k *of set* $R^{C(u_i)}$*, and* $|Uni(f_{jk})|$ *returns the unique group value pair amount in feature pair* (j, k)*.*

The weight β_{jk} indicates the distribution matching degree of the group value pairs of two features to the class labels. Same as Intra-coupling weight α_j, this weight measures the importance of different feature pairs, and determines the relationship comes from other attributes strong or weak. A difference with Intra-coupling weight is two high Intra-coupling weights, such as a high α_j and a high α_k, do not guarantee a high β_{jk}.

4.2.5 Integration

Finally, we aggregate the weights and the coupling relationship, and propose an *Integrated Coupled Similarity* for numerical data set.

The **Integrated Coupled Similarity** represents the adapted coupling similarity measure by taking into account the relationships inner an attributes and relationships comes from other attributes. Formally, we have

$$\begin{aligned} &IS(v_i^x, v_j^y) \\ &= \alpha_j Ra^{Intra}(v_j^x, v_j^y)(v_j^{i_1}, v_j^{i_2}) \cdot \sum_{k=1}^{n} \beta_{jk} Ra_{ij|k}^{Inter}(v_i^{i_1}, v_j^{i_2}) \\ &(k \neq j), \end{aligned} \tag{4.8}$$

where v_i^x and v_i^y are the numerical values in attribute i; α_j and β_{jk} denote the weight of coupling defined in Equation (4.6) and (4.7).

In this paper, we illustrate our supposed method by kNN. After obtaining the similarity between the instances u_e and $\{u_i\}$, we choose the k nearest neighbors that correspond to the k highest similarity values. The most frequently occurred class c_f in the k neighbors is the desired class for u_e.

Based on the above definition, we apply the coupling similarity between numerical data to classic nearest neighbor algorithm: kNN, and compared the result with classic kNN using Euclidean distance. And the supposed

Coupled kNN on numerical data is presented in Algorithm 4.1.

Algorithm 4.1 : Coupled kNN Algorithm on Numerical Data

Require: An instance u_t without label and a labeled numerical dataset $D\{u_1, u_2, ..., u_n\}$, $|D| = m$

Ensure: The class label of u_t

1: CAIM Data Discretization;
2: Transfer information table into new discretized information table
3: **for** $i = 1$ **to** n **do**
4: Calculate the Coupling Weights using Eq.(4.6) and (4.7)
5: **end for**
6: **for** $i = 1$ **to** m **do**
7: Calculate the Intra-Coupled and Inter-Coupled Relationship using Eq.(4.4) and Eq.(4.5)
8: **end for**
9: **for** $i = 1$ **to** m **do**
10: Calculate $d(u_t, u_i)$ based on the Coupled Relation
11: **end for**
12: Identify the k nearest neighbors $N(u_i)$ for u_i
13: **for** $j = 1$ **to** k **do**
14: Count the quantity of each class in $N(u_i)$
15: **end for**
16: **Return** the class label $L(u_t)$ of object u_t by voting

4.3 Experiments and Result

4.3.1 Data Sets and Settings

In this section, several experiments are performed on 15 UCI data sets (i.e. Table 4.3) to show the effectiveness of our proposed coupled representation scheme for numerical objects.

To verify the superiority of our proposed coupled method, we use the k-nearest neighbor (k-NN) algorithm (Fix & Hodges Jr 1951) to compare the classification quality. The k-NN classifier can be one of the most straightforward classifiers in machine learning techniques - classification is achieved by identifying the nearest neighbors to a query example and using those neighbors to determine the class of the query. This approach to classification is easy today because issues of poor run-time performance is not such a problem any more with the computational power that is available.

We carry out experiments on all 15 data sets and we use the 10-fold 3-repeat cross-validation and with $k = 3, 5, 7$, respectively.

4.3.2 Evaluation Criteria

Sensitivity, specificity and accuracy are widely used evaluation criteria to describe a classifier's performance. In particular, they are used to quantify how good and reliable a classifier is. As shown in Table 4.4, the confusion matrix has several terms that are commonly used along with them. They are true positive (TP), true negative (TN), false negative (FN), and false positive (FP). Specially, the three evaluation criteria can be described as:

Sensitivity evaluates how good the classifier is at detecting a positive one:

$$\text{Sensitivity} = \frac{TP}{TP + FN} \tag{4.9}$$

Specificity estimates how likely a negative one can be correctly ruled out:

$$\text{Specificity} = \frac{TN}{TP + FN} \tag{4.10}$$

Accuracy measures how correct a classifier identifies and excludes a given condition:

$$\text{Accuracy} = \frac{(TN + TP)}{(TN + TP + FN + FP)} \tag{4.11}$$

As we know, a better data representation approach corresponds to a better classification result, i.e. higher Accuracy, higher Precision, and higher Specificity (Figueiredo, Rocha, Couto, Salles, Gonçalves & Meira Jr 2011). Given

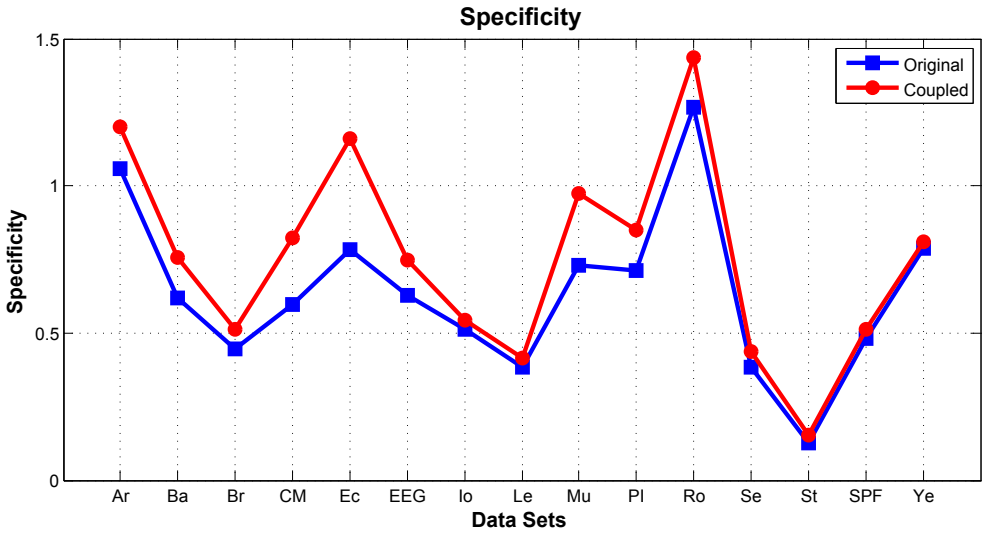

Figure 4.1: The Comparison of Specificity.

the importance and popularity of these criteria, we choose all these three criteria to evaluate the performance of our supposed method.

4.3.3 Experiments Result

The experiment results are shown in Figure 4.1 - Figure 4.3. For each evaluation criterion, the bigger the better.

From the Comparison of Accuracy figure, we can see that compared with Euclidean distance, the classifier combined with our coupled relation can get an outstanding improvement on all these data sets. For example, on data set "Bank-Marketing", the classifier based on Euclidean distance can get an accuracy about 82 %, while with coupling distance it can be as high as 93%.

The result figures indicate that Coupled-kNN remarkably outperforms the original k-NN for all the data sets, which implies that exploiting the relationship between different numerical features is effective, and especially for our Coupled-kNN, the improvement is significant. This also implies

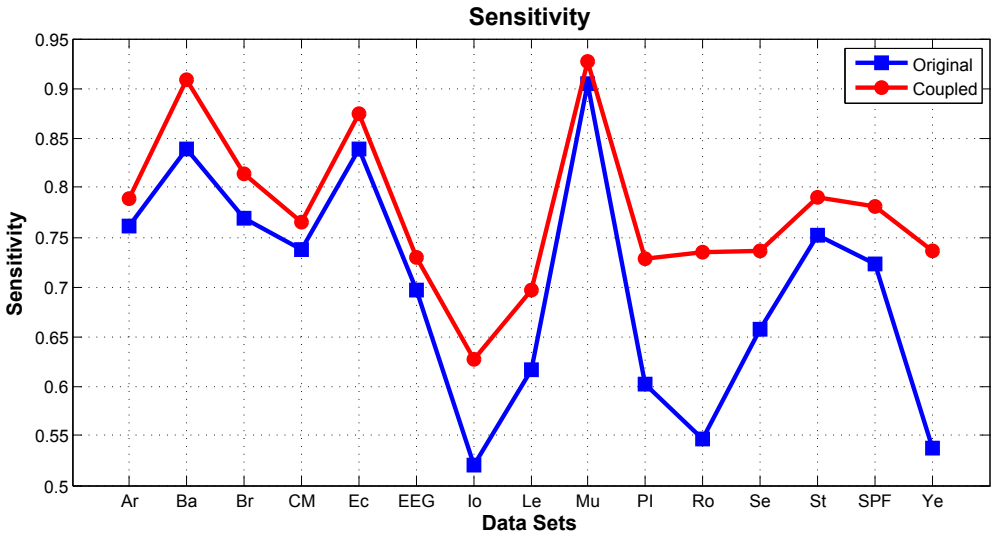

Figure 4.2: The Comparison of Sensitivity.

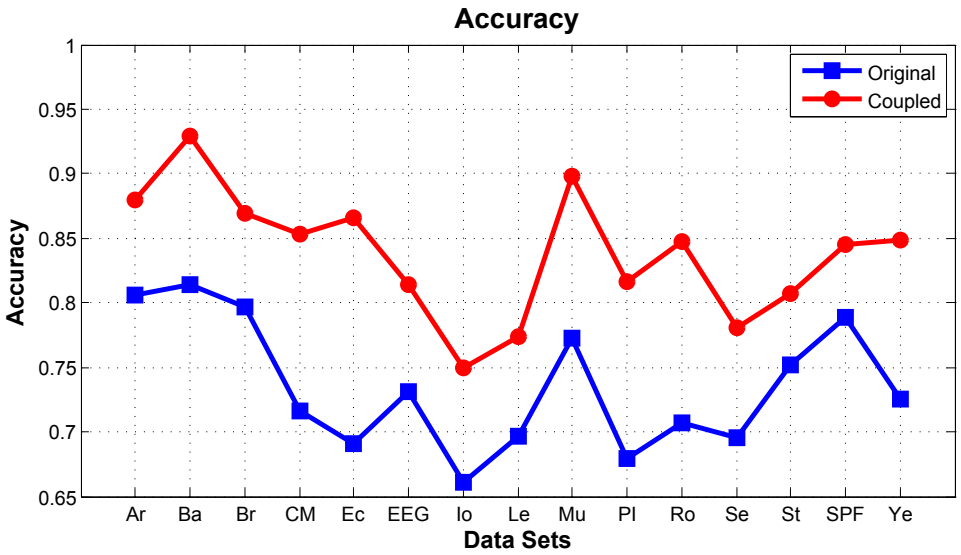

Figure 4.3: The Comparison of Accuracy.

that only consider the local value to value information is not sufficient, and the interactions between different attributes should also be considered when calculating the numerical distance.

Overall, our proposed Coupled-kNN outperforms the original Euclidean based k-NN on all three measures. The maximal relative improvement rate across all the data sets is 165.43%, while the minimal rate is 6.57%. All the results are supported by a statistical significant test at 93% significance level, which suggests the effectiveness and superiority of our proposed coupled method.

It is worth noting that although our proposed method runs the best on average, it does not mean that it is suitable for all kinds of data. For example, when used on data set "Statlog","Steel-Faults" and "Yeast", the Specificity result is not as good as on other data sets. It even got a similar result as the original k-NN did.

The reason is the coupled interactions between different attributes of these data sets are weak or loose, and our extended coupling similarity may introduce more noisy information than the useful information. Only around 32.5% pairs of attributes and their powers have significant coupling relationships for the data set "Statlog","Steel-Faults" and "Yeast", compared to the average percentage of around 74.8% on other data sets. But in terms of average performance, our method performs the best.

4.4 Summary

In this chapter, we have introduced a coupled relationship for objects by the concept of intra-coupled relationship and inter-coupled relationship on numerical attributes, by considering the inner relationship in and between the attributes. We also provide two weights to evaluate the proportion of intra-coupling and the inter-coupling, which offers a metric to measure the strongness of coupling between numerical values. Substantial experiments have verified that distance based classification algorithm combined with our

coupled relationship outperforms the original one, and that also supported by statistical analysis.

[***Note***] A conference version of this chapter has been submitted to and has been reviewed by **ICDM15** as below:

- **Chunming Liu**, Longbing Cao (2015) Coupling based Classification for Numerical Data, submitted to and been reviewed by the IEEE International Conference on Data Mining series (**ICDM15**).

Table 4.1: Example of Information Table: Wine in UCI

Wine	**Malic acid** (a_1)	**Ash** (a_2)	**Total phenols** (a_3)	**Proanthocyanins** (a_4)	**Class**
u_1	1.53	2.7	2.95	1.35	c_1
u_2	1.8	2.65	2.35	1.54	c_1
u_3	1.81	2.41	2.7	1.86	c_1
u_4	1.67	2.62	1.92	1.34	c_2
u_5	5.65	2.45	1.68	1.06	c_2
u_6	1.33	2.3	2.2	1.38	c_2
u_7	2.06	2.46	1.95	1.35	c_3
u_8	1.67	2.48	1.68	1.34	c_3
u_9	3.91	2.48	1.8	1.41	c_3

Table 4.2: Discretization of Information Table: Wine in UCI

Wine	**Malic acid** (a_1)	**Ash** (a_2)	**Total phenols** (a_3)	**Proanthocyanins** (a_4)	**Class**
u_1	g_1^1	g_2^3	g_3^3	g_4^2	c_1
u_2	g_1^2	g_2^3	g_3^3	g_4^3	c_1
u_3	g_1^2	g_2^1	g_3^3	g_4^3	c_1
u_4	g_1^1	g_2^3	g_3^2	g_4^1	c_2
u_5	g_1^3	g_2^1	g_3^1	g_4^1	c_2
u_6	g_1^1	g_2^1	g_3^3	g_4^2	c_2
u_7	g_1^3	g_2^2	g_3^2	g_4^2	c_3
u_8	g_1^1	g_2^2	g_3^1	g_4^1	c_3
u_9	g_1^3	g_2^2	g_3^1	g_4^2	c_3

Table 4.3: Numerical Data Sets from UCI

Data Set	**Feature Types**	**#Instances**	**#Attributes**	**#Class**	**Short As**
Arcene	Real	900	10000	2	Ar
Bank-Marketing	Real	45211	17	2	Ba
Breast-Cancer	Integer	699	10	2	Br
Climate-Model	Real	540	18	2	CM
Ecoli	Real	336	8	8	Ec
EEG-Eye-State	Integer, Real	14980	15	2	EEG
Ionosphere	Integer, Real	351	34	2	Io
Leaf	Real	340	16	40	Le
Musk-2	Integer	6598	168	2	Mu
Pima-Indians	Integer, Real	768	8	2	PI
Robot-Failures	Integer	164	90	5	Ro
Seeds	Real	210	7	3	Se
Statlog	Real	2310	19	7	St
Steel-Faults	Integer, Real	1941	27	2	SPF
Yeast	Real	1484	8	10	Ye

Table 4.4: The Confusion Matrix of Binary Classification

	Condition positive	**Condition negative**
Test positive	True positive (TP)	False positive (FP)
Test negative	False negative (FN)	True negative (TN)
Column Total	TP+FN	FP+TN

Chapter 5

Coupled Similarity for Mixed Type Data

5.1 Overview

5.1.1 Background

Classification analysis plays an important practical role in several domains, such as machine learning and data mining. Classification techniques have been widely used in retail, finance, banking, security, astronomy, and behavioral ecology, etc. (Kantardzic 2011).

In many research and application areas, the features of data sets are mixture of both categorical and numerical attributes (mixed data sets). If the objects are described by numerical attributes, their similarity measures reflect the direct relationship between data values. For example, the values pair (100kg, 120kg) are more similar than (100kg, 20kg), in other words, more close to each other. There are a variety of similarity metrics developed for numerical data, such as Euclidean and Minkowski distances (Gan et al. 2007), however, for categorical data, although several similarity measures such as the Jaccard coeffcient (Pang-Ning, Steinbach, Kumar et al. 2006), overlap, and Goodall similarity (Boriah et al. 2008) can be used, they are usually not

as straightforward and general as similarities for continuous data.

The classification analysis for the class-imbalanced data sets has received a lot attention, but they are mainly for numerical or categorical type data, and there are much less attention for the mixed type data described by numerical and categorical features. It is well known that the traditional algorithms do not perform as good on imbalanced data sets as on balanced data sets, so we illustrate the problems with the existing work and highlight the challenge of classifying class-imbalanced mixed type data below. Let's take a fragment of the UCI Nursery data (Table 5.1) as an example, eleven instances are divided into two classes with four categorical features: "parents", "has-nurs", "form" and "social". The value in the brackets indicates the frequency of the corresponding feature value. It is a class-imbalanced categorical dataset. Here, we use the first instance u_0 as the testing data instance, and the rest $\{u_1, u_2, \cdots, u_{10}\}$ as the training dataset. We know that the Overlap Similarity is defined as

$$\text{Sim\_Overlap}(v_i, v_j) = \begin{cases} 0, & \text{if } v_i \neq v_j \\ 1, & \text{if } v_i = v_j \end{cases}. \tag{5.1}$$

So in Table 5.1, the maximum similarity is $Sim\_Overlap(u_0, u_4)$, which is 0.75. While if we adopt the Frequency Based Cosine Similarity, which is defined as

$$\text{Sim\_Cosine} = \frac{V_i \cdot V_j}{||V_i||\,||V_j||}, \tag{5.2}$$

where the V_i and V_j is the instances' corresponding frequency vectors, then the instances u_5, u_2, u_8 and u_4 will be the top 4 instances which are close to u_0. Under this scenario, u_0 will be assigned to class B rather than class A if we using a nearest neighbor classifier such as kNN, because there are always more nearest neighbors labeled as class B than as class A.

From this example we can see that the main problem of categorical data is there is no inherent order in the different categorical values for we can not distinguish which one should be the Maximum and which one should be the Minimum, for example, there are no order for the words "Cloudy", "Sunny"

and "Rainy". To simplify the problem and inspired by the numerical distance measure, people using the simplest way, the Overlap Similarity, to define the similarity between two categorical values. As shown in Equation 5.1, it assigns a 1 if the values are identical and a 0 if the values are not identical. Then for two multivariate categorical data points, the similarity between them will be expressed by the number of attributes in which they match. The overlap measure does not distinguish between the different values taken by an attribute, all matches as well as mismatches, are treated as equal. This will cause problems in some situations. For example, considering a categorical dataset D', which has only two features: "Weather" and "Time". The feature "Weather" has three values: {"Sunny", "Cloudy", "Rainy"}, and feature "Time" has also three values: {"morning", "afternoon", "evening"}. Table 5.2 shows the frequency of co-occurrence of these two features.

Based on dataset D', the overlap similarity between the two instances (Cloudy,morning) and (Cloudy,afternoon) is $\frac{1}{2}$, and the overlap similarity between (Rainy,morning) and (Rainy,afternoon) is also $\frac{1}{2}$. But the frequency distribution in Table 5.2 shows that the first pair are frequent combinations, while the second pair are very rare combinations in the dataset. Hence, the overlap measure is too simplistic to give equal importance to all matches and mismatches. This example shows that there is some other information in categorical data sets that can be used to define what makes two values more or less similar.

In computing the frequency based cosine similarity, the vector comes from the frequency of a single value of a feature, so it ignores the information hiding in the co-occurrence of two features. For example in Table 5.2, as the frequency of "Cloudy" and "Rainy" are all 100, and the frequency of "morning" and "afternoon" are 100 too, so their corresponding frequency vectors are the same, and hence the frequency based cosine similarity can not distinguish the difference between different categorical values.

These examples show that traditional similarity measures are unable to capture the genuine relationships between imbalanced categorical data.

Learning from the class-imbalanced data has also been identified as one of the top 10 challenging problems in data mining research (Yang & Wu 2006). To handle the class imbalance classification tasks, various solutions have been proposed. In general, these methods can be broadly divided into two different approaches: data re-sampling methods and algorithm modifying methods (López, Fernández, García, Palade & Herrera 2013).

The data re-sampling methods focus on balancing the data, and the common strategies are to reduce the majority class examples (undersampling) or to add new minority class examples to the data (oversampling)(Chawla et al. 2002, Estabrooks, Jo & Japkowicz 2004), such as the SMOTE(Chawla et al. 2002). These re-sampling methods show to outperform the original algorithms in most situation, but they do not achieve much improvement for kNN on imbalanced data in some cases. One reason for this is because of the maximum-specificity induction bias of kNN in which the classification decision is made by examining the local neighborhood of query instances, and therefore the global re-sampling strategies may not have pronounced effect in the local neighborhood under examination; and another reason for this is the re-sampling strategies inevitably change the inherent relationships of the original data, or even worse, delete important information or add noise.

Unlike data re-sampling methods which change the original data relationships, the algorithm modifying methods alter the existing algorithms to make them more effective in dealing with imbalanced data, while keeping the original data unchanged. For example, kENN(Yuxuan & Zhang 2011), CCW-kNN(Liu & Chawla 2011), CCPDT(Liu, Chawla, Cieslak & Chawla 2010) and the cost-sensitive method (Zadrozny et al. 2003).

5.1.2 Challenges and Solutions

From the previous introduction, we know that both data re-sampling methods will add noise to or change the original information in the data, while for algorithm modifying methods , they do not consider the inner-interaction between different features, and seldom handling the mixed type data while

focusing on numerical or categorical data only. So it is a much challenging issue to handle mixed type data with the original data unchanged and consider the inner relationship between features in the same time.

In this chapter, we propose a novel hybrid coupled nearest neighbor classification algorithm for class-imbalanced mixed type data by addressing both the relationships between classes and between features.

The chapter is organized as follows. Section 5.1 briefly reviews the background and challenges of the topic. Preliminary definitions are specified in Section 5.2. Section 5.3 explains our classification algorithm in detail. The experimental results are discussed in Section 5.4. The conclusion of this chapter is summarized in Section 5.5.

5.2 Preliminary Definitions

Given a dataset D which contains m instances, $D = \{u_1, \cdots, u_m\} = \{AC\}$, where $A = \{a_1, \cdots, a_n\}$ is a set of n categorical and numerical features and $C = \{c_1, \cdots, c_L\}$ is a set of L class labels, in which each class has dramatically different numbers of instances. The goal is to classify an unlabeled testing instance u_t based on the instances in the training set D with known classes. For example, Table 5.1 exhibits a class-imbalanced dataset. The training set consists of ten objects $\{u_1, u_2, \cdots, u_{10}\}$, four features {"parents","hasnurs","form","social"}, and two classes {"A","B"}. There are only two instances in class A, while eight instances in class B. Our task is to find a suitable classification model to categorize u_0 into class A.

In the following sections, when we say a class c_l is smaller (or larger) than c_k, it means that the instances in class c_l is less (or more) than that in class c_k. A minority class has a relatively small size, while a majority class has a relatively large size.

5.3 Coupled Similarity for Mixed Type Data

In this section, a hybrid coupled kNN algorithm (i.e. *HC-kNN* for short) is proposed to classify the class-imbalanced mixed type data. Our HC-kNN consists of four parts: *data discretization*, *weight calculation*, *coupling similarity calculation*, and *integration*. At the first step of data discretization, we use CAIM discretization algorithm (Kurgan & Cios 2004) which can capture the class-attribute interdependency information on numerical features, so as to apply our coupling similarity to these numerical features. In the second part of weight calculation, we first introduce a Class Size Weight to handle the class-imbalanced issue, and this membership provides the quantification on how small a class is, then we use feature-class and feature-pair coupling relationship to assign every feature and feature-pair a proper weight. At the step of coupling similarity calculation, we present the *Adapted Coupled Nominal Similarity* to describe the closeness of two different instances. Finally, we integrated the coupling similarity with all the weights to measure the similarity between instances with mixed type features. Below, we will specify all the building blocks one by one.

5.3.1 Data Discretization

In order to apply our strategy which compute the similarity between numerical features and categorical features, we first do discretization on numerical attributes to transfer these continuous values into separate groups. As we are conducting the supervised classification tasks, we choose CAIM (class-attribute interdependence maximization) discretization algorithm (Kurgan & Cios 2004) which can capture the class-attribute interdependency information as our discretization method.

The method uses class-attribute interdependency information as the criterion for the optimal discretization. For a given quanta matrix, the CAIM criterion measures the dependency between the class variable C and the dis-

cretization variable D for attribute F. It is defined as:

$$\text{CAIM}(C, D|F) = \frac{\sum_{r=1}^{n}(\frac{max_r^2}{M_{+r}})}{n}, \tag{5.3}$$

where n is the number of intervals, r iterates through all intervals, and max_r is the maximum value within the rth column of the quanta matrix, M_{+r} is the total number of continuous values of attribute F that are within the interval $(d_{r-1}, d_r]$.

The discretization method starts with a single interval that covers all possible values of a continuous attribute, and divides it iteratively. From all possible division points that are tried it chooses the division boundary that gives the highest value of the CAIM criterion.

The result we got from this discretization on numerical attributes is only used in the next Weight Calculation stage and Inter-similarity calculation stage. The reason we do so is because we cannot compute the similarity between a numerical value and a categorical value directly for the numerical value is continuous. So we use the discretization intervals as the categories of the continuous values, then we can evaluate the co-occurrence between numerical data and categorical data.

5.3.2 Weight Calculation

In this part, we introduce three weight for our method: *the Class Size Weight*, *the Feature Weight* and *the Feature-pair Weight*.

A) Class Size Weight

In the class-imbalanced dataset, there are usually several small classes (i.e. minority) that contain much less instances, while a lot more instances are in some large classes (i.e. majority). However, what exactly does a small class mean? How do we quantify a small class? As it would be too reductive to regard the smallest class as the minority, we use the Class Size Weight to measure how small a class is according to its size. Accordingly, we have:

Definition 10 *The **Class Size Weight** $\theta(\cdot)$ denotes to what extend a class c_l belongs to the minority. Formally, $\theta(\cdot)$ is defined as:*

$$\theta(c_l) = \frac{1}{2^{\frac{|c_l|}{m}}}, \tag{5.4}$$

where $|c_l|$ is the number of instances in classes c_l and m is the total number of instances in the dataset. Accordingly, we have $\theta(c_l) \in (0.5, 1)$.

The sized membership of class describes how small a class is. In special cases, $\theta(c_l)$ will reach the maximum if c_l has the smallest number of instances; $\theta(c_l)$ will down to the minimum if c_l is the largest class. For other classes, the corresponding Class Size Weight falls within $(\theta(c_l)^{min}, \theta(c_l)^{max})$. When a dataset is balanced with two classes, where we have $\theta(c_l) = 0.707$. In Table 5.1, for instance, we have $\theta(c_A) = 0.871$, and $\theta(c_B) = 0.574$.

B) Feature Weight

Definition 11 *The **Feature Weight** describes the importance of each categorical feature (or discretized numerical feature) f_j by considering the value distribution consistency with the distribution of classes. Formally, we have:*

$$\alpha_j = \begin{cases} \sum_{i=1}^{m} \frac{Fre(x_{ij}, R^{C(u_i)})}{m \cdot |R^{C(u_i)}|} & if \;\; |Unique(f_j)| > 1 \\ \\ 0 & if \;\; |Unique(f_j)| = 1 \end{cases} \tag{5.5}$$

where m is the total number of instances in the dataset, x_{ij} is the j feature value for instance u_i, $R^{C(u_i)}$ consists of all the instances which share the same class as instance u_i, and the according instance number is $|R^{C(u_i)}|$, while $Fre(x_{ij}, R^{C(u_i)})$ is a frequency count function that count the occurrences of x_{ij} in feature j of set $R^{C(u_i)}$, and $|Unique(f_j)|$ returns the category number or discretization interval number in feature j.

The weight α_j indicates the distribution matching degree of the values of a feature to the class labels. For example, if a training dataset

has 6 instances with class labels of $\{C_1, C_1, C_2, C_2, C_2, C_1\}$ respectively, and feature f_1 has values of $\{A, A, B, B, B, A\}$ while feature f_2 has values of $\{M, M, M, N, N, N\}$, then the value distribution of feature f_1 is more consistent with the distribution of classes than f_2 does. The more consistent in distribution for the feature values to the class labels, the more important the feature is. If all the values in a feature are same, that is, $|Unique(f_j)| = 1$, then this feature cannot be used in the classification task and we set the weight to zero. We also regard this feature weight as *the coupling relationship between features and classes*. For example, in Table 5.1, we will have the normalized feature weights: $\alpha_1 = 0.2586, \alpha_2 = 0.2069, \alpha_3 = 0.2414$, and $\alpha_4 = 0.2931$.

C) Feature-Pair Weight

Definition 12 *The **Feature-Pair Weight** describes the importance of each feature-pair (f_j, f_k) $(j \neq k)$according to their value-pairs' distribution consistency with the distribution of classes. Formally, we have:*

$$\gamma_{jk} = \begin{cases} \sum_{i=1}^{m} \sqrt{(\frac{Fre(x_i^{jk}, R^{C(u_i)})}{m \cdot |R^{C(u_i)}|})} & if \quad |Unique(f_{jk})| > 1 \\ 0 & if \quad |Unique(f_{jk})| = 1 \end{cases} \tag{5.6}$$

where m is the total number of instances in the dataset, x_i^{jk} is the value pair of the jth feature and the kth feature for instance u_i, $R^{C(u_i)}$ consists of all the instances which share the same class as instance u_i, and the according instance number is $|R^{C(u_i)}|$, while $Fre(x_i^{jk}, R^{C(u_i)})$ defines as a frequency count function that count the occurrences of value pair x_i^{jk} in feature j and feature k of set $R^{C(u_i)}$, and $|Unique(f_{jk})|$ returns the unique value-pair amount in feature pair (j, k).

The Feature-Pair Weight γ_{jk} indicates the distribution matching degree of the value pairs of two features to the class labels. Same as feature weight α_j, this weight measures the importance of different feature-pairs, and its range

is confined to inter-similarity. γ_{jk} is different from α_j, for two high feature weights, such as a high α_j and a high α_k, will not guarantee a high γ_{jk}. We also regard this weight as *the coupling relationship between feature-pairs and classes.*

5.3.3 Similarity Calculation

In this part, the similarity between instances is defined for the class-imbalanced mixed type data. Wang et al. (Wang et al. 2011) introduce a coupled nominal similarity (CNS) for categorical data, but it is only defined for categorical clustering tasks. Here, we adapt the CNS in our classification algorithm and extend it to mixed type data which contains both categorical features and numerical features. We use the Euclidean distance in our intra-similarity calculation on numerical features, and if the inter-similarity calculation relates to numerical features, we apply a same strategy on its discretization result as we do on categorical features.

A) Intra-Feature Coupling Similarity

Definition 13 *Given a training dataset D, a pair of values $v_j^x, v_j^y (v_j^x \neq v_j^y)$ of feature a_j. The **Intra Coupling Similarity** between categorical feature values v_j^x and v_j^y of feature a_j is formalized as:*

$$\delta^{Intra}(v_j^x, v_j^y) = \frac{RF(v_j^x) \cdot RF(v_j^y)}{RF(v_j^x) + RF(v_j^y) + RF(v_j^x) \cdot RF(v_j^y)}, \tag{5.7}$$

where $RF(v_j^x)$ and $RF(v_j^y)$ are the relative occurrence frequency of values v_j^x and v_j^y in feature a_j, respectively.

The Intra Coupled Similarity just reflects the direct interaction of two values in the same feature. The higher these frequencies are, the closer such two values are. Thus, Equation (5.7) is designed to capture the value similarity in terms of occurrence times by taking into account the frequencies of categories. Besides, since $1 \leq RF(v_j^x), RF(v_j^y) \leq m$, then $\delta^{Intra} \in [1/3, m/(m+2)]$. For example, in Table 5.1, values "usual" and "great-pret" of feature "parents"

are observed four and two times, so $\delta^{Intra}((\text{usual}), (\text{great-pret})) = (4*2)/(4+2+4*2) = 4/7$.

For numerical features, we use $1/(1 + Euclidean)$ as the feature values' Intra-similarity δ^{Intra}.

B) Inter-Features Coupling Similarity

In contrast, the Inter Coupling Similarity defined below is to capture the interaction of two values (or the group in the discretization result) from one feature while based on the co-occurrence of other features.

Definition 14 *Given a training dataset D and two different features a_i and a_j $(i \neq j)$, two feature values $v_i^x, v_i^y (v_i^x \neq v_i^y)$ from feature a_i and a feature value v_j^z from feature a_j. v_i^x and v_i^y are inter-related if there exists at least one pair value (v_p^{xz}) or (v_p^{yz}) that co-occurs in features a_i and a_j of instance U_p. The **Inter-Coupling Similarity** between feature values v_i^x and v_i^y of features a_i according to feature value v_j^z of a_j is formalized as:*

$$\delta_{i|j}^{Inter}(v_i^x, v_i^y | v_j^z) = 2 \cdot \frac{\min\left(F(v_p^{xz}), F(v_p^{yz})\right)}{RF(v_i^x) + RF(v_i^y)}, \tag{5.8}$$

where $F(v_p^{xz})$ and $F(v_p^{yz})$ are the co-occurrence frequency count function for value pair v_p^{xz} or v_p^{yz}, and $RF(v_i^x)$ and $RF(v_i^y)$ is the occurrence frequency in feature i.

Accordingly, we have $\delta_{i|j}^{Inter} \in [0, 1]$. The Inter-Coupled Similarity reflects the interaction or relationship of two categorical values from one feature but based on the connection to other feature. In Table 5.1, for example, as $\delta_{1|2}^{Inter}((\text{usual}), (\text{pretentious})|(\text{proper})) = 2 \cdot \min(1, 2)/(4 + 4) = 1/4$ and $\delta_{1|2}^{Inter}((\text{usual}), (\text{great-pret})|(\text{proper})) = 2 \cdot \min(1, 1)/(4 + 2) = 1/3$, while $\delta_{1|2}^{Inter}((\text{usual}), (\text{pretentious})|(\text{less-proper})) = 2 \cdot \min(1, 2)/(4 + 4) = 1/4$, and $\delta_{1|2}^{Inter}((\text{usual}), (\text{great-pret})|(\text{less-proper})) = 0$, so in feature 1 ("parents"), the Inter-Coupling Similarity of value pair [(usual),(pretentious)] is much higher than the value pair [(usual),(great-pret)] based on feature 2 ("has-nurs") as $(1/4 + 1/4) > 1/3$. In other words, that means if we consider feature 2, the

categorical value "usual" is more close to the value "pretentious" than to the value "great-pret".

C) Adapted Coupling Object Similarity

By taking into account the feature importance, the *Adapted Coupling Object Similarity* between instances u_{i_1} and u_{i_2} is formalized as:

$$\begin{aligned} & AS(u_{i_1}, u_{i_2}) \\ & = \sum_{j=1}^{n} [\alpha_j \delta_j^{Intra} \cdot \sum_{k=1}^{n} \gamma_{jk} \delta_{j|k}^{Inter}] \\ & = \sum_{j=1}^{n} [\alpha_j \delta_j^{Intra}(v_j^{i_1}, v_j^{i_2}) \cdot \sum_{k=1}^{n} \gamma_{jk} \delta_{j|k}^{Inter}(v_j^{i_1}, v_j^{i_2} | v_k)], \qquad (k \neq j) \end{aligned} \tag{5.9}$$

where $v_j^{i_1}$ and $v_j^{i_2}$ are the values of feature j for instances u_{i_1} and u_{i_2}, respectively. δ_j^{Intra} and $\delta_{j|k}^{Inter}$ are the intra-feature coupling similarity (defined in Equation 5.7) and inter-features coupling similarity (defined in Equation 5.8), respectively. The α_j is the feature weight defined in Equation (5.5) and the γ_{jk} is the feature-pair weight defined in Equation (5.6).

5.3.4 Integration

Finally, we aggregate all the previous factors into a new *Integrated Similarity*.

The **Integrated Similarity** is represented by the adapted coupling similarity measure which taking into account the feature weight, the feature-pair weight, the intra-coupling and inter-coupling as well as the class size information. Formally, we have

$$IS(u_t, u_i) = \theta(C(u_i)) \cdot AS(u_t, u_i), \qquad (t \neq i) \tag{5.10}$$

where u_t and u_i are two different instances, respectively; $C(u_i)$ denotes the class of u_i; $\theta(\cdot)$ is the sized membership of class defined in Equation (5.4); and $AS(\cdot)$ is the adapted coupling object similarity defined in Equation (5.9).

As indicated by Equation (5.10), on one hand, the $\theta(\cdot)$ can capture the class size information, which is helpful to handle the class imbalance issues

and suitable for multi-class classification tasks; On the other hand, the adapted similarity $AS(\cdot)$ includes not only the feature-class coupling information (feature weight), but it also capture the feature values' intra-feature coupling relationship and inter-feature coupling relationship based on different features. These coupling relationship reflect the inner relationship of real world data. By doing data discretization, we break out the limit which coupled relationship can only be applied in categorical dataset, and extend such strategy to mixed data type. Therefore, the similarity in our algorithm is more reasonable than that in the existing similarity calculation related algorithms for the imbalanced real world mixed type data. Therefore, the similarity in our algorithm is more reasonable than that in the existing similarity calculation related algorithms for the imbalanced categorical data. Algorithm 5.1 illustrates the main idea of our algorithm.

In the experiments, we illustrate our method by kNN algorithm. After obtaining the similarity between the instances u_t and $\{u_1, u_2, \cdots, u_m\}$, we choose the k nearest neighbors which have highest similarity values. The most frequently occurred class c_f in the k neighbors is the target class for u_t. For example, in Table 5.1, we have $IS(u_0, u_1) = 3.9785$, $IS(u_0, u_2) = 3.8054$ and $IS(u_0, u_5) = 3.8332$ to be the top three nearest neighbors to u_0, so u_0 should be labeled as its real class, class A (with $k = 3$).

5.4 Experiments and Evaluation

5.4.1 Experiments Setting

As the publicly available data sets were often not designed for the non-IIDness test as in this work, we choose the commonly used UCI (Bache & Lichman 2013), KEEL (Alcalá et al. 2010) data and some real world data, which are all contain both numerical and categorical features. If our algorithm can show improvement on such data compared to the baselines, then it has potential to differentiate itself from others in more complex data with strong coupling relationships. In total, 10 data sets are taken from the UCI

Data Repository, KEEL data set repository, and an Australian university's students performance database. A short description of these data sets is provided in Table 5.3. The imbalanced rate (minority/total) is shown as *IRate*. These data sets have been selected as they typically have an imbalance class distribution (the highest IR is 101.04) and all contain both categorical and numerical features(as shown in Table 5.3, the "N" in "#(N+C) Features" denotes the numerical feature numbers, and the "C" in "#(N+C) Features" denotes the categorical feature numbers). The data sets such as D9 and D10 which have a more balanced class distribution are selected to evaluate our method's expansion capability.

We conducted 10-fold cross validation three times (Raeder, Hoens & Chawla 2010) to evaluate the performance in all experiments. We not only select several variants of kNN, such as the classic kNN (Wu & Kumar 2009), kENN(Yuxuan & Zhang 2011), CCW-kNN(Liu & Chawla 2011) and SMOTE based kNN to compare with, but also the very popular classifiers C4.5 and NaiveBayes. To make the strategy more acceptable, we further incorporate our coupling method into some kNN-based algorithms (the new ones are named with a prefix of $HC+$) to compare the performance after and before combined with coupling. In all our experiments, we set $k = 5$ to all those kNN-based classifiers, and the confidence levels for kENN is set to 0.1.

To evaluate the performance od algorithms on class-imbalanced data classification, we use Receiver Operating Characteristic (ROC) curve and the Area Under the ROC Curve (AUC)(Fawcett 2006) to evaluate the results. As the AUC indicates the average classification performance, so the better performance of an algorithm is, the higher AUC it achieves.

5.4.2 Results and Analysis

Table 5.4 shows the AUC results for our HC-kNN compared with the state of the art algorithms. The top two results are highlighted in bold. From the table we can see that compared with other approaches, our HC-kNN has the highest AUC result and outperforms others in most of the data

sets, especially in data sets with high imbalance rate. Also, our proposed HC-kNN always outperforms the classic kNN on all data sets. This proves that considering the coupling relationships between features and classes can capture the intrinsic data characteristics. Note that the SMOTE-based kNN does not always demonstrate significant improvement compared with kNN, sometimes even worse (such as in dataset D2 and D10). It means that only using SMOTE on imbalanced data may not bring much improvement, but even some noise.

From Table 5.4 we can see that when the imbalance rates are higher than 15.13, our method can achieve a much better improvement (the least one is 3.36% and the highest one is 12.09%) on these data which does not incorporate much non-IIDness characteristics. That confirms again that our coupling strategy is very effective for imbalanced mixed type data classification tasks.

The Experiment 2 aims to show the changes of incorporating the coupling similarity into other classification algorithms before and after. For doing this, we create three comparison sets by integrating the proposed coupling mechanism into kENN to form a CF+kENN, CCWkNN to form a CF+CCWkNN, and SMOTE based kNN to form a CF+SMOTE based kNN, and compared their performance. All comparable algorithms are with the same parameter settings.

Table 5.5 shows the performance results of these comparable algorithms with vs. without the coupling mechanism. The result shows that incorporating our new similarity metrics will bring more or less improvement for these classic distance or similarity-based algorithms. This further shows that our proposed idea can capture the intrinsic characteristics, and is better than existing methods, especially for class-imbalanced mixed type data.

The Experiment 3 evaluated our coupling similarity on different imbalance rate. We do SMOTE on student data and create 50 new data sets, in which the minority class varies from 1% to 50% of the total instances. Fig. 5.1, Fig. 5.2 and Fig. 5.3 shows the improvement of the basic algorithms which combined with our Coupling Similarity on different imbalance rate. As

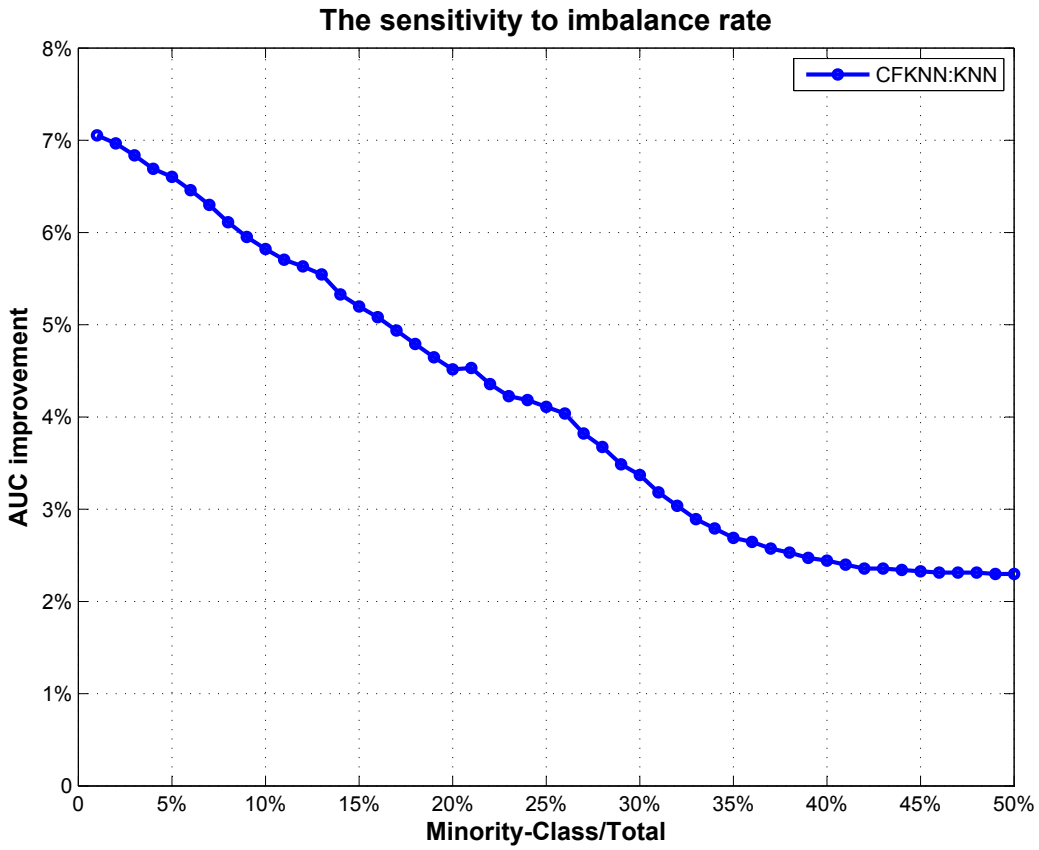

Figure 5.1: Sensitivity of IR (CF-kNN:kNN)

it shows in these figures, when minority class takes up less than 10% of the total instances, both kNN and kENN (combined with Coupling) can have an improvement of over 5.821%, and even for CCWkNN, the improvement can be over 5.372%. But with the imbalance rate declining, this improvement falls simultaneously. When minority class comes to 35% of the total records (which can be defined as "balanced" data in some cases) or over, the improvement will not be so outstanding while stay stable at about 2.2%. This experiment demonstrates that our strategy is sensitive to the imbalance rate, and it is more suitable for being used in the scenario with high imbalance rate, that is, imbalanced mixed type Non-IID data.

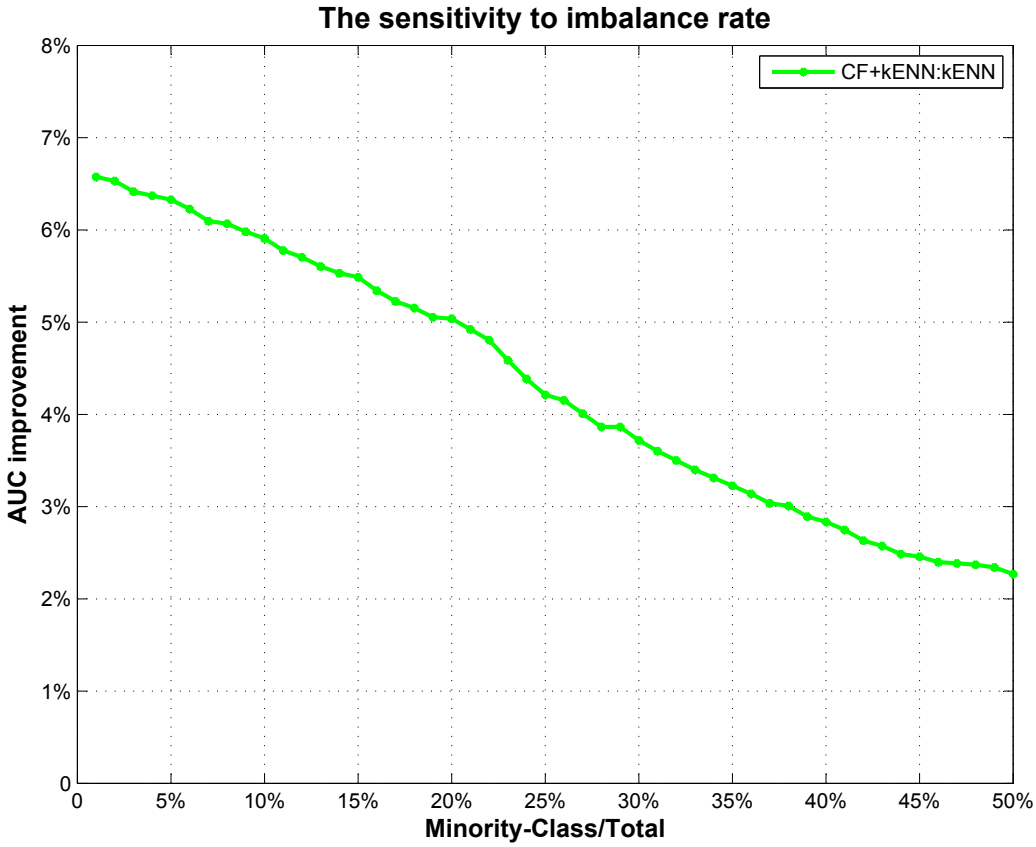

Figure 5.2: Sensitivity of IR (CF+kENN:kENN)

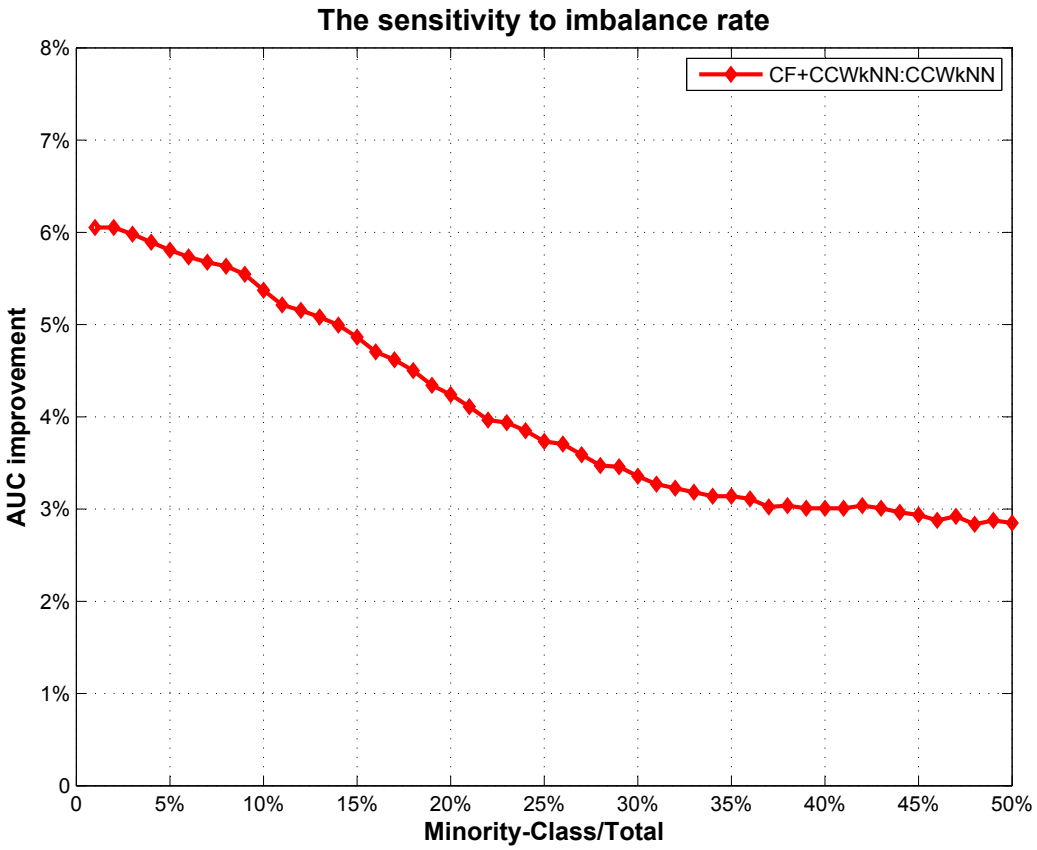

Figure 5.3: Sensitivity of IR (CF+CCWkNN:CCWkNN)

5.5 Summary

Traditional classifiers mainly focus on dealing with balanced dataset and overlook the coupling relationships between data. Classifying imbalanced data which have both numerical and categorical features is very challenging. We proposed a hybrid coupling kNN to classify imbalanced mixed type data with strong relationships between features and classes. It incorporates the class size information, the feature and feature-pair importance, the intra-feature and the inter-features interaction together to handle the tasks. The experiment results show that our HC-kNN has a more stable and higher average performance than some other algorithms, such as the kNN, kENN, CCWkNN, SMOTE-based kNN, Decision Tree and NaiveBayes, when applied for class-imbalanced data which have both numerical and categorical features.

[***Note***] A conference version (Liu, Cao & Yu 2014*b*) of this chapter has been accepted already and be published by IJCNN2014 as below:

- **Chunming Liu**, Longbing Cao, Philip S Yu (2014), A hybrid coupled k-nearest neighbor algorithm on imbalance data. *in* 'Proceedings of the Neural Networks (IJCNN), 2014 International Joint Conference on (**WCCI14**)', IEEE, pp. 2011-2018. (**ERA ranking: A**)

Table 5.1: An Fragment from The UCI Dataset: Nursery Data

ID	**parents**	**has-nurs**	**form**	**social**	**Class**	$Sim_{Overlap}$	Sim_{Cosine}
u_0	usual	improper	foster	nonprob	A		
u_1	usual (4)	proper (4)	incomplete (4)	slightly-prob (5)	A	0.25	0.8484
u_2	pretentious (4)	less-proper (3)	completed (2)	nonprob (2)	A	0.25	0.9278
u_3	usual (4)	less-proper (3)	incomplete (4)	slightly-prob (5)	B	0.25	0.8660
u_4	usual (4)	improper (1)	incomplete (4)	nonprob (2)	B	0.75	0.8762
u_5	usual (4)	critical (1)	completed (2)	problematic (3)	B	0.25	0.9731
u_6	pretentious (4)	proper (4)	complete (3)	problematic (3)	B	0	0.8744
u_7	pretentious (4)	proper (4)	incomplete (4)	slightly-prob (5)	B	0	0.8484
u_8	pretentious (4)	less-proper (3)	foster (1)	slightly-prob (5)	B	0.25	0.8956
u_9	great-pret (2)	proper (4)	complete (3)	slightly-prob (5)	B	0	0.7253
u_{10}	great-pret (2)	very-crit (1)	complete (3)	problematic (3)	B	0	0.8002

Table 5.2: The Frequency of Values Co-occurrence

	morning	***afternoon***	***evening***	**Total**
Sunny	44	47	9	100
Cloudy	48	45	7	100
Rainy	8	8	84	100
Total	100	100	100	

Algorithm 5.1 : Hybrid Coupled kNN Algorithm

Require: $U = \{u_1, \cdots, u_m\}$ is a set of m instances; $F = \{a_1, \cdots, a_n\}$ is a set of n categorical and numreical features; $C = \{c_1, \cdots, c_l\}$ is a set of l classes, in which each class has dramatically different numbers of instances; u_t is an instance without label

Ensure: The class label of u_t

1: **for** i=1 **to** l
2: **if** f_i is numerical, then **Do** discretization on f_i
3: **end**
4: **for** i=1 **to** l
5: $\theta(c_i) \leftarrow$ class size weight
6: **end**
7: **for** i=1 **to** n
8: $\alpha_i \leftarrow$ feature weight
9: **end**
10: **for** i=1 **to** m
11: $\delta_i^{Ia} \leftarrow$ Intra-feature Coupling
12: **for** j=1 **to** n
13: $\delta_{i|j}^{Ie} \leftarrow$ Inter-feature Coupling
14: **end**
15: $AS \leftarrow$ Adapted Coupled Object Similarity
16: $IS \leftarrow$ Integerated Similarity
17: **end**
18: Select k nearest neighbors to u_t
19: **Return** the most frequent class label in those k neighbors

Table 5.3: The Data Sets with Mixed Type Features

Index	**Dataset**	**Source**	**#Instances**	**#(N+C)**	**#Class**	**Minority**	**IRate**
D1	Student	REAL	50000	(24+8)	2	Positive	101.0
D2	Abalone	UCI	4177	(7+1)	29	Class15	39.5
D3	Annealing	UCI	798	(6+32)	5	U	22.5
D4	Dermatology	UCI	366	(1+33)	6	P.R.P.	17.3
D5	Census-Income	UCI	299285	(12+28)	2	5000+	15.1
D6	Zoo	UCI	101	(1+16)	7	Set6	11.6
D7	Contraceptive	UCI	1473	(2+7)	3	Long-term	3.4
D8	Adult	UCI	45222	(6+8)	2	>50K	3.2
D9	German Credit	KEEL	1000	(7+13)	2	bad	2.3
D10	Credit Approval	UCI	690	(6+9)	2	positive	1.2

Table 5.4: The AUC Results Comparison for HC-kNN and Other Algorithms

Dataset	Minority(%)	**HC-kNN**	kNN	kENN	CCWkNN	SMOTE	C4.5	Naive	**improvement**
D1	0.98%	**0.909**	0.845	0.849	0.854	**0.866**	0.857	0.857	4.97%-7.59%
D2	2.47%	**0.718**	0.672	0.680	**0.692**	0.688	0.683	0.682	3.75%-6.89%
D3	4.26%	**0.768**	0.714	0.735	**0.743**	0.732	0.737	0.729	3.36%-7.49%
D4	5.46%	**0.76**	0.715	0.720	**0.729**	0.678	0.716	0.724	4.28%-12.09%
D5	6.20%	**0.815**	0.782	**0.803**	0.798	0.788	**0.803**	0.791	1.49%-4.28%
D6	7.92%	**0.887**	0.842	**0.869**	**0.869**	0.854	0.857	0.859	2.08%-5.30%
D7	22.61%	**0.755**	0.718	0.729	0.725	**0.743**	0.726	0.736	1.64%-5.12%
D8	23.93%	**0.938**	0.904	0.915	0.910	0.910	**0.920**	0.919	1.95%-3.79%
D9	29.72%	**0.769**	0.738	**0.757**	0.744	0.755	0.752	0.756	1.53%-4.24%
D10	44.50%	**0.916**	0.893	**0.913**	0.910	0.887	0.907	0.912	0.33%-3.27%

Table 5.5: Comparison for Algorithms With and Without Coupling

Dataset	Minority(%)	kENN	CF+kENN	CCWkNN	CF+CCWkNN	SMOTE	CF+SMOTE
D1	0.98%	0.849	**0.905**	0.854	**0.906**	0.866	**0.922**
D2	2.47%	0.680	**0.724**	0.692	**0.733**	0.688	**0.735**
D3	4.26%	0.735	**0.783**	0.743	**0.788**	0.732	**0.778**
D4	5.46%	0.720	**0.766**	0.729	**0.771**	0.678	**0.718**
D5	6.20%	0.803	**0.912**	0.798	**0.873**	0.788	**0.836**
D6	7.92%	0.869	**0.922**	0.869	**0.918**	0.854	**0.908**
D7	22.61%	0.729	**0.764**	0.725	**0.725**	0.743	**0.776**
D8	23.93%	0.915	**0.957**	0.910	**0.946**	0.910	**0.951**
D9	30.00%	0.757	**0.780**	0.744	**0.785**	0.755	**0.800**
D10	44.50%	0.913	**0.936**	0.910	**0.932**	0.887	**0.907**

Chapter 6

Coupling Analysis in Multi-label Classification

6.1 Overview

6.1.1 Background

In traditional single-label classification task, given a set of m possible distinct classes, each instance is associated with one and only one class. Let X be the input space and L be the label space, the task of single-label classification is to find such a function $h : X \rightarrow L$ from the training examples. A wide variety of machine learning algorithms, such as the k-nearest neighbors, support vector machine, and logistic regression methods, have been proposed to resolve such single-label classification tasks. Although these traditional approaches have been proved to be successful in handling some real world problems, for the problems which the objects do not fit the single-label rule, they may not work well, e.g., a news article may associates with Sports, Olympics and Doping topics.

Such tasks are usually denoted as multi-label classification problems. In fact, a conventional single-label classification problem can simply be taken as a special case of the multi-label classification problem where there has only

one label in the class label space. Multi-label classification problems exist in many domains (Tsoumakas & Katakis 2007), for example, in automatic text categorization, a document can associate with several topics, such as arts, history and Archeology; and in gene functional analysis of bio-informatics, a gene can belong to both metabolism and transcription classes; and in music categorization, a song may labeled as Mozart and sad; and in image classification tasks, an image may contain several concepts simultaneously, such as beach, mountain and sunset.

In dealing with multi-label classification problems, many solutions have been proposed. In general, all these methods can be broadly divided into two approaches: problem transformation methods and algorithm adaptation methods.

Problem Transformation

The problem transformation methods firstly transforms multi-label learning tasks into multiple single-label learning tasks, and these new tasks are then handled by the standard single-label learning algorithms. There exist two straightforward problem transformation methods (Boutell et al. 2004) which force the multi-label problem into a traditional single-label classification. The first one randomly or subjectively selects one of the multiple labels of each multi-label instance as its label and discards all the others, while the second approach simply discards all the instances that have multi labels. These two problem transformation methods discard lots of useful information from the original data and therefore are never considered in the real world application.

The third problem transformation method considers each distinct label set that exist in the multi-label data set as a new single label (Diplaris, Tsoumakas, Mitkas & Vlahavas 2005), and then transfer the multi-label problem into a traditional multi-class problem. The main problem of this method is that it may transfer the dataset into a dataset with a large number of classes while few instances per class, i.e., a sparse dataset.

The most commonly used problem transformation method is named as

Table 6.1: An Example of Multi-label Data

Instances	Label1	Label2	Label3	Label4
u_1	l_1			l_4
u_2			l_3	l_4
u_3	l_1		l_3	
u_4		l_2	l_3	
u_5		l_2	l_3	l_4

"binary relevance", which learns $|L|$ separate binary classifiers, one for each different label l ($l \in L$) (Boutell et al. 2004, Gonçalves & Quaresma 2003, Lauser & Hotho 2003). By labeling the instances with one specific label as positive ones while labeling the others as negative ones, it duplicates the original dataset into $|L|$ separate data sets. Every single dataset contains all the instances of the original dataset. Then the traditional binary classifier can be trained for each unique label one by one. For example, for the dataset shown in Table 6.1, four new data sets will be generated and each corresponding to a particular class label (Table 6.2). For each dataset in Table 6.2, any instance with the associated label is marked as positive (+) or negative (-) according to its original label value. Then the traditional single-label binary classification algorithms can then be applied to each dataset and train out four separate classifiers. If combined with kNN, then it comes to be the BR-kNN.

Algorithm Adaptation

The algorithm adaptation methods are those methods that extend specific learning algorithms in order to handle multi-label data. As mentioned previously, ML-kNN (Zhang & Zhou 2007), IBLR (Cheng & Hüllermeier 2009), AdaBoosting.MH (Schapire & Singer 2000), BSVM (Boutell et al. 2004), BP-MLL (Zhang & Zhou 2006) and Decision Tree (Clare & King 2001) are all state-of-the-art algorithm adaptation methods.

Table 6.2: Transformed Data Sets using Binary Relevance

Instances	Data1 (Label1)	Data2 (Label2)	Data3 (Label3)	Data4 (Label4)
u_1	+	-	-	+
u_2	-	-	+	+
u_3	+	-	+	-
u_4	-	+	+	-
u_5	-	+	+	+

The k-nearest neighbor (kNN) algorithm (Cover & Hart 1967) has a long history in the data mining area for single-label classification. The kNN algorithm is an intuitive yet effective machine learning method for solving conventional classification problems. It is generally regarded as an instance-based learning or lazy learning method because hypotheses are constructed locally and the computation is deferred until the test dataset is acquired. Simply to say, in kNN, a new instance is classified to the most common class by a majority vote of its k nearest neighbor instances. If integrated with a problem transformation method, it is easy to adapt the kNN algorithm for multi-label classification.

A number of multi-label learning methods are adapted from kNN (Brinker & Hüllermeier 2007, Spyromitros et al. 2008, Wieczorkowska et al. 2006, Zhang & Zhou 2007). ML-kNN, the first multi-label lazy learning approach, is based on the traditional kNN algorithm and the maximum a posterior (MAP) principle (Zhang & Zhou 2007). ML-kNN first finds out the nearest neighbors of the new instance, and then based on statistical information from the neighboring instance, maximum a posterior principle is applied to determine the label set for the new instance. The rationale for the approach is that an instance's labels depend on the number of neighbors that possess identical labels. Given an instance x with an unknown label set $L(x) \subseteq L$, ML-kNN first identifies the k nearest neighbors in the training data and counts the number of neighbors belonging to each class (i.e. a variable z

from 0 to k). Then the maximum a posterior principle is used to determine the label set for the test instance. The posterior probability of $l_i \in L$ is given by

$$P(l_i \in L(x)|z) = \frac{P(z|l_i \in L(x)) \cdot P(l_i \in L(x))}{P(z)} \tag{6.1}$$

Then, for each label $l_i \in L$, the algorithm builds a classifier h_i using the rule

$$h_i(x) = \begin{cases} 1 & P(l_i \in L(x)|z) > P(l_i \notin L(x)|z) \\ \\ 0 & \text{otherwise} \end{cases} \tag{6.2}$$

$h_i(x) = 1$ means label l_i is in x's real label set, while 0 means it does not. z is the number of neighbors belonging to a specific class. The prior and likelihood probabilities in Eq. 6.1 are estimated from the training dataset in advance.

ML-kNN has two inheriting merits from both lazy learning and MAP principle: one is the decision boundary can be adaptively adjusted due to the varying neighbors identified for each new instance, and another one is that the class-imbalance issue can be largely mitigated due to the prior probabilities estimated for each class label.

Subsequently, some researchers developed the Instance-Based Logistic Regression (IBLR) algorithm (Cheng & Hüllermeier 2009), which is a combination of instance-based learning and logistic regression techniques. It includes the statistics of the k nearest neighbors as features in logistic regression. Both IBLR and ML-kNN are widely used in multi-label classification algorithms that exploit instance-based learning, and can sometimes outperform state-of-the-art model-based approaches (Cheng & Hüllermeier 2009, Zhang & Zhou 2007).

6.1.2 Challenges and Solutions

Although simple and powerful, there are some shortcomings in its processing strategy. ML-kNN uses the popular binary relevance (BR) strategy (Vembu & Gärtner 2011), which may transfer the problem into many class-imbalance

tasks, and then tend to degrade the performance of the classifiers. Another problem of it is the estimation of the posteriori may be affected by the facts that the instances with and without a particular label are typically highly imbalanced. Furthermore, its ignorance of the inter relationship between labels is another issue which limits its usage. Such relationship is described as a Coupled behavior in some previous research (Cao et al. 2012, Cao 2014*a*). In (Wang et al. 2011, Liu et al. 2014*a*), Can and Liu etc. analysis the coupling relationship on categorical data. These works all proved the effectiveness of considering the dependency between different attributes.

This chapter is organized as follows. Section 6.1 briefly reviews the background and challenges of the topic, and a brief solution is suggested. Preliminary definitions are specified in Section 6.2 and also give a detailed description of the new algorithm we proposed. The experimental results are discussed in Section 6.3. Finally, the conclusion is discussed in Section 6.4.

6.2 Methodology

In this chapter, a kNN algorithm based on non-iidness (Cao 2014*b*) (*CML-kNN* for short) is proposed to handle the multi-label data classification problems.

6.2.1 Problem Statement

We formally define the multi-label classification problem as this: Let X denotes the space of instances and $Y = \{l_1, \ldots, l_n\}$ denotes the whole label set where $|Y| = n$. $T = \{(x_1, L(x_1)), \ldots, (x_m, L(x_m))\}$ ($|T| = m$) is the multi-label training data set, whose instances are drawn identically and independently from an unknown distribution D. Each instance $x \in X$ is associated with a label set $L(x) \in Y$. The goal of our multi-label classification is to get a classifier $h : X \to Y$ that maps a feature vector to a set of labels, while optimizing some specific evaluation metrics.

6.2.2 Coupled Label Similarity

In this part, the similarity between labels is defined for the multi-label data. It is much more doable for numerical data to calculate the distance or similarity, since the existing metrics such as Manhattan, Euclidean distance and coefficient were mainly built for numeric variables. But the labels are categorical data. How to denote the similarity between them is a big issue. As we all know, matching and frequency (Boriah et al. 2008) are the most common ways to measure the similarity of categorical data. Accordingly, two similarity measures are defined: the Overlap Similarity between two categorical value v_i and v_j is defined as

$$\text{OverlapSimilarity}(v_i, v_j) = \begin{cases} 1, & \text{if } v_i = v_j \\ 0, & \text{if } v_i \neq v_j, \end{cases} \tag{6.3}$$

and the Frequency Based Cosine Similarity between two vectors V_i and V_j is defined as

$$\text{CosineSimilarity}(V_i, V_j) = \frac{V_i \cdot V_j}{||V_i||\,||V_j||}. \tag{6.4}$$

The overlap similarity between two categorical values is to assign 1 if they are identical otherwise 0 if different. Further, for two multivariate categorical data points, the similarity between them will be proportional to the number of features in which they match. This will cause problems in some situations. For example, considering a categorical dataset D_c, which has only two features: color and size. Color takes three possible values: red, green, blue, and size takes three values: small, medium and large. Table 6.3 shows the frequency of co-occurrences of the two categorical values.

Based on the feature values given by dataset D_c, the overlap similarity between the two instances (green, small) and (green, medium) is $\frac{1}{2}$, and the overlap similarity between (blue, small) and (blue, medium) is also $\frac{1}{2}$. But the frequency distribution in Table 6.3 shows that (green, small) and (green, medium) are frequent co-occurrences, while (blue, small) and (blue, medium) are very rare co-occurrences. Hence, the overlap measure is too

Table 6.3: Frequency of Value Pairs

	small	***medium***	***large***	**Total**
red	44	47	9	100
green	48	45	7	100
blue	8	8	84	100
Total	100	100	100	

simplistic by just giving the equal importance to matches and mismatches, and the co-occurrence information in categorical data reflects the interaction between features and can be useful to define what makes two categorical values more or less similar. However, such co-occurrence information hasn't been incorporated into the existing similarity metrics.

To capture the inner relationship between categorical labels, we introduce an *Intra-Coupling Label Similarity (IaCLS)* and an *Inter-Coupling Label Similarity (IeCLS)* below to capture the interaction of two label values from two different labels.

Definition 15 *Given a training multi-label data set D and two different labels l_i and l_j $(i \neq j)$, the label value is v_i^x, v_j^y respectively. The* ***Intra-Coupling Label Similarity*** *(IaCLS) between label values v_i^x and v_j^y of label l_i and l_j is formalized as:*

$$\delta^{Ia}(v_i^x, v_j^y) = \frac{RF(v_i^x) \cdot RF(v_j^y)}{RF(v_i^x) + RF(v_j^y) + RF(v_i^x) \cdot RF(v_j^y)}, \tag{6.5}$$

where $RF(v_i^x)$ and $RF(v_j^y)$ are the occurrence frequency of label value v_i^x and v_j^y in label l_i and l_j, respectively.

The Intra-coupling Label Similarity reflects the interaction of two different label values in the label space. The higher these similarities are, the closer such two values are. Thus, Equation (6.5) is designed to capture the label value similarity in terms of occurrence times by taking into account the frequencies of categories. Besides, since $1 \leq RF(v_i^x), RF(v_j^y) \leq m$, then $\delta^{Ia} \in [1/3, m/(m+2)]$.

In contrast to the Intra-Coupling, we also define an *Inter-Coupling Label Similarity* below to capture the interaction of two different label values according to the co-occurrence of some value (or discretized value group) from feature spaces.

Definition 16 *Given a training multi-label data set D and two different labels l_i and l_j $(i \neq j)$, the label value is v_i^x, v_j^y respectively. v_i^x and v_j^y are defined to be Inter-Coupling related if there exists at least one pair value (v_p^{zx}) or (v_p^{zy}) that occurs in feature a_z and labels of instance U_p. The* ***Inter-Coupling Label Similarity*** *(IeCLS) between label values v_i^x and v_i^y according to feature value v_p^z of feature a_z is formalized as:*

$$\delta^{Ie}(v_i^x, v_j^y | v_p^z) = \frac{\min\left(F(v_p^{zx}), F(v_p^{zy})\right)}{\max(RF(v_i^x), RF(v_j^y))}, \tag{6.6}$$

where $F(v_p^{zx})$ and $F(v_p^{zy})$ are the co-occurrence frequency count function for value pair v_p^{zx} or v_p^{zy}, and $RF(v_i^x)$ and $RF(v_j^y)$ is the occurrence frequency of related class label. v_p^z is the value in categorical feature a_z or the discretized value group in numerical feature a_z.

Accordingly, we have $\delta^{Ie} \in [0, 1]$. The Inter-Coupling Label Similarity reflects the interaction or relationship of two label values from label space but based on the connection to some features.

Definition 17 *By taking into account both the Intra-Coupling and the Inter-Coupling, the* ***Coupled Label Similarity*** *(CLS) between two label values v_i^x and v_j^y is formalized as:*

$$CLS(v_i^x, v_j^y) = \delta^{Ia}(v_i^x, v_j^y) \cdot \sum_{k=1}^{n} \delta^{Ie}(v_i^x, v_j^y | v_k), \tag{6.7}$$

where v_i^x and v_j^y are the label values of label l_i and l_j, respectively. δ^{Ia} and δ^{Ie} are the intra-coupling label similarity (defined in Eq. 6.5) and inter-coupling label similarity (defined in Eq. 6.6), respectively. n is the number of attributes and v_k denotes the values in the k*th feature a_k.*

Table 6.4: CLS Array

	Label1	**Label2**	**Label3**	**Label4**
Label1	1.0	0	0.25	0.33
Label2	0	1.0	0.50	0.33
Label3	0.25	0.50	1.0	0.50
Label4	0.33	0.33	0.50	1.0

The *Coupled Label Similarity* defined in Eq. 6.7 reflects the interaction or similarity of two different labels. The higher the *CLS*, the more similar two labels be. In Table 6.1, for example, $CLS(l_1, l_4) = 0.33$, $CLS(l_1, l_3) = 0.25$, so in the data set, an instance with label l_4 is more similar or close to instances with label l_1 than those instances with label l_3 do. That is to say, label pair (l_1,l_4) is closer to each other than the label pair (l_1, l_3). For Table 6.1, we got the coupled label similarity array which showed in Table 6.4.

6.2.3 Extended Nearest Neighbors

Based on the Coupled Label Similarity, we introduce our extended nearest neighbors. Based on the similarity between labels, we can transfer a label set into a set with only a certain label, it also means a multi-label instance can be extended to a set of single-label. If we specify a basic label l_b, then any instance can be transformed into a set with only one label l_b. For example, in Table 6.1, instance u_5 has a label set of $\{l_2, l_3, l_4\}$, then according to the label similarity array Table 6.4, it can be transformed into $\{1 \cdot l_2, 0.5 \cdot l_2, 0.33 \cdot l_2\}$ if we choose label l_2 as the basic label. We can then call the original multi-label instance u_5 equals a single-label instance with a label of $\{1.83 \cdot l_2 | l_2\}$.

If u_5 is the neighbor of some instance, when we consider the label l_2, the instance u_5 can be presented as an instance which contains $1 + 0.5 + 0.33 = 1.83$ label l_2, and vice versa, instance u_5 also presents there are $(1-1) + (1-0.5) + (1-0.33) = 1.17$ instances which not contain the label l_2, and there will have $(1.83 + 1.17 = 3 = |L(u_5)|)$. This is the basic idea when we finding our extended nearest neighbors.

Table 6.5: Extended Nearest Neighbors

instance	Extended Neighbors	To Label
u_5	$0 \cdot l_1 + 0.25 \cdot l_1 + 0.33 \cdot l_1$	l_1
u_5	$1 \cdot l_2 + 0.5 \cdot l_2 + 0.33 \cdot l_2$	l_2
u_5	$0.5 \cdot l_3 + 1 \cdot l_3 + 0.5 \cdot l_3$	l_3
u_5	$0.33 \cdot l_4 + 0.5 \cdot l_4 + 1 \cdot l_4$	l_4

6.2.4 Coupled ML-kNN

For the unseen instance x, lets $N(x)$ represent the set of its k nearest neighbors identified in data set D. For the j-th class label, CML-kNN chooses to calculate the following statistics:

$$C_j = Round(\sum_{i=1}^{k} \delta_{L_i^*|j}) \tag{6.8}$$

Where L_i is the label set of the i-th neighbor and $L_i \in N(x)$, and $\delta_{L_i^*|j}$ denotes the sum of the CLS values of the i-th neighbor's label set to the j-th label l_j, and $Round()$ is the rounding function.

Namely, C_j is a rounding number which records all the CLS value of all x's neighbors to label l_j.

Let H_j be the event that x has label l_j , and $P(H_j|C_j)$ represents the posterior probability that H_j holds under the condition that x has exactly C_j neighbors with label l_j . Correspondingly, $P(\neg H_j|C_j)$ represents the posterior probability that H_j doesn't hold under the same condition. According to the MAP rule, the predicted label set is determined by deciding whether $P(H_j|C_j)$ is greater than $P(\neg H_j|C_j)$ or not:

$$Y = \{l_j | \frac{P(H_j|C_j)}{P(\neg H_j|C_j)} > 1, 1 \leq j \leq q\} \tag{6.9}$$

According to the Bayes Theory, we have:

$$\frac{P(H_j|C_j)}{P(\neg H_j|C_j)} = \frac{P(H_j) \cdot P(C_j|H_j)}{P(\neg H_j) \cdot P(C_j|\neg H_j)} \tag{6.10}$$

Here, $P(H_j)$ and $P(\neg H_j)$ represents the prior probability that H_j holds and doesn't hold. Furthermore, $P(C_j|H_j)$ represents the likelihood that x

has exactly C_j neighbors with label l_j when H_j holds, and $(P(Cj|\neg Hj))$ represents the likelihood that x has exactly C_j neighbors with label l_j when H_j doesn't hold.

When we counting the prior probabilities, we integrated our coupled label similarity into the process:

$$\begin{aligned} P(H_j) &= \frac{s + \sum_{i=1}^{m} \delta_{L_i^*|j}}{s \times 2 + m \times n}; \\ P(\neg H_j) &= 1 - P(H_j); \end{aligned} \tag{6.11}$$

where $(1 \leq j \leq n)$ and m is the records number in training set, and s is a smoothing parameter controlling the effect of uniform prior on the estimation which generally takes the value of 1 (resulting in Laplace smoothing).

Same as ML-kNN, for the j-th class label l_j, our CML-kNN maintains two frequency arrays α_j and β_j. As our method considers the other labels which has a similarity to a specific label, the frequency arrays will contain $k \times n + 1$ elements:

$$\begin{aligned} \alpha_j[r] &= \sum_{i=1}^{m} \delta_{L_i^*|j} | C_j(x_i) = r \qquad (\delta_{L_i^*|j} \geq 0.5) \\ \beta_j[r] &= \sum_{i=1}^{m} (n - \delta_{L_i^*|j}) | C_j(x_i) = r \quad (\delta_{L_i^*|j} < 0.5) \\ & (0 \leq r \leq k \times n) \end{aligned} \tag{6.12}$$

Here, we take an instance with $\delta_{L_i^*|j} \geq 0.5$ as an instance which does have label j and we take an instance with $\delta_{L_i^*|j} < 0.5$ as an instance which doesn't have label j. Therefore, $\alpha_j[r]$ counts the sum of CLS values to label j of training examples which have label l_j and have exactly r neighbors with label l_j, while $\beta_j[r]$ counts the CLS to label j of training examples which don't have label l_j and have exactly r neighbors with label l_j. Afterwards,

the likelihoods can be estimated based on elements in α_j and β_j:

$$\begin{aligned} P(C_j|H_j) &= \frac{s + \alpha_j[C_j]}{s \times (k \times n + 1) + \sum_{r=0}^{k \times n} \alpha_j[r]} \\ P(C_j|\neg H_j) &= \frac{s + \beta_j[C_j]}{s \times (k \times n + 1) + \sum_{r=0}^{k \times n} \beta_j[r]} \\ (1 \leq j \leq n, 0 \leq C_j \leq k \times n) & \end{aligned} \tag{6.13}$$

Thereafter, by combining the prior probabilities (Eq.6.11) and the likelihoods (Eq.6.13) into Eq.(6.10), we will get the predicted label set in Eq.(6.9).

6.2.5 Algorithm

Given an unknown test instance x_t, the algorithm determines the final label set of the instance. Algorithm 6.1 illustrates the main idea of our process. Our proposed CML-kNN contains of six main parts.

a) Maintain the label similarity array;

b) Finding the nearest neighbors for every instance in training set;

c) Getting the prior probabilities and frequency arrays;

d) Finding the nearest neighbors for the target instance;

e) Calculate the statistics value;

f) Calculate the result.

Firstly, we calculate the label similarity according to their inter-relationships and maintain the Coupled Label Similarity Array $A(L)$ from the training data set. Secondly, for every training instance, we identify its traditional k nearest neighbors. After that, for every different label, we calculate its prior probability which combined with CLS. Simultaneously, we expand the neighbors set for every instance to a new label-coupled neighbors set using the CLS, and calculate the frequency array for every label. After these works done, we identify the k neighbors of the test instance x_t. After applying CLS on

this neighbor set and calculate the label statistics, we can finally get the predicted label set.

It is worth noting that our key idea is the label similarity, which tries to learn the label distance and then transfer any label into a specific label. Below, we will specify all the building blocks one by one.

Algorithm 6.1 : Coupled ML-kNN Algorithm

Require: An unlabeled instance x_t and a labeled dataset $T\{(x_1, L(x_1)), \ldots, (x_m, L(x_m))\}$, where $|T| = m$ and $|L| = n$

Ensure: The label set $L(x_t)$ of instance x_t

1: Calculate the CLS array $A(L)$ according to Eq.(6.7)
2: **for** $i = 1$ **to** m **do**
3: Identify the k nearest neighbors $N(x_i)$ for x_i;
4: **end for**
5: **for** $j = 1$ **to** n **do**
6: Calculate $P(H_j)$ and $P(\neg H_j)$ according to Eq.(6.11);
7: Maintain the label-coupled frequency arrays α_j, β_j using Eq.(6.12);
8: **end for**
9: Identify the k nearest neighbors $N(x_t)$ for x_t;
10: **for** $j = 1$ **to** n **do**
11: Calculate the statistic C_j according to Eq.(6.8);
12: **end for**
13: **Return** the label set $L(x_t)$ of instance x_t according to Eq.(6.9) ;

6.3 Experiments and Evaluation

6.3.1 Experiment Data

A total of eight commonly-used multi-label data sets are tested for experiments in this study, and the statistics of the data sets are shown in Table 6.6. Given a multi-label data set $M = \{(x_i, L_i)|1 \leq i \leq q\}$, we use $|M|$, $f(M)$, $La(M)$, $F(M)$ to represent the number of instances, number of features,

number of total labels, and feature type respectively. In addition, several multi-label statistics (Read et al. 2011) are also shown in the Table:

a) Label cardinality: $LC(M) = \frac{1}{q}\sum_{i=1}^{q}|L_i|$, which measures the average number of labels per example;

b) Label density: $LD(M) = \frac{LC(M)}{La(M)}$, which normalizes $LC(M)$ by the number of possible labels;

c) Distinct label sets: $DL(M) = |\{L|\exists x : (x, L) \in M\}|$, which counts the number of distinct label combinations appeared in the data set;

d) Proportion of distinct label sets: $PDL(M) = \frac{DL(M)}{|M|}$, which normalizes $DL(M)$ by the number of instances.

As shown in Table 6.6, eight data sets are included and are ordered by Label density: $LD(M)$.

6.3.2 Experiment Setup

In our experiments, we compare the performance of our proposed CML-kNN with that some state-of-the-art multi-label classification algorithms: ML-kNN, IBLR and BSVM. All nearest neighbor based algorithms are parameterized by the size of the neighborhood k. We set the value of $k = 9$ (odd number for voting), and use the Euclidean metric as the distance function when computing the nearest neighbors, and for BSVM, models are learned via the cross-training strategy(Boutell et al. 2004). We also choose the BR-kNN as the basic algorithm to compare with. We perform 10-fold cross-validation three times on all the above data sets.

6.3.3 Evaluation Criteria

Multi-label classification requires different metrics than those used in traditional single-label classification. A lot of criteria have been proposed for evaluating the performance of multi-label classification algorithms (Tsoumakas

& Katakis 2007). Given a multi-label data set $M = \{(x_i, L_i)|1 \leq i \leq q\}$. Let h be a multi-label classifier and $Y_i = h(x_i)$ be the set of labels predicted by h for instance x_i. We use three evaluation criteria for multi-label classification: the Hamming loss, one error and average precision (Schapire & Singer 2000). The definitions of the criteria are described as follows:

a) The Hamming Loss :

$$HammingLoss = \frac{1}{q}\sum_{i=1}^{q}\frac{|h(x_i)\Delta Y_i|}{|L|}, \tag{6.14}$$

where Δ stands for the symmetric difference of two sets and corresponds to the XOR operation in Boolean logic.

b) One error: the metric computes how many times the top-ranked label is not relevant.

$$OneError = \frac{1}{q}\sum_{i=1}^{q} I(argmax_{\lambda \in L} f(x_i, \lambda) \notin Y_i), \tag{6.15}$$

where $I(\pi)$ equals 1 if π holds and 0 otherwise.

c) Average precision: this is the average of the per-instance average precision over all test instances:

$$AP = \frac{1}{q}\sum_{i=1}^{q}\frac{1}{|Y_i|}\cdot\frac{|\{y'|R_f(x_i, y') \leq R_f(x_i, y), y' \in Y_i\}|}{R_f(x_i, y)}, \tag{6.16}$$

6.3.4 Experiment Results

The experiment results are shown in Table 6.7 - Table 6.9. For each evaluation criterion, "↓" indicates "the smaller the better", while "↑" indicates "the bigger the better". And the numbers in parentheses denote the rank of the algorithms among the five compared algorithms.

The result tables indicate that CML-kNN and BSVM outperforms other algorithms significantly, which implies that exploiting the frequency of neighbors' label is effective, and especially for our CML-kNN, the improvement is significant compared to BR-kNN, that means incorporating the label

relationship will greatly improve the BR strategy. Meanwhile, ML-kNN, I-BLR and BR-kNN do not perform as well compared to the other algorithms. This implies that only exploiting the exact neighbor information is not sufficient, and the similar neighbor (correlations between labels) should also be considered.

Overall, our proposed CML-kNN outperform all the compared methods on all three measures. The average ranking of our method on these data sets using three different metrics is (1.50, 1.50, 1.50), the first, while the second best algorithm, BSVM, only achieves (2.50, 2.38, 2.25). The BR-kNN performs the worst, which only achieves (4.13,4.25,4.75).

It is worth noting that although our proposed method runs the best on average, it does not mean that it is suitable for all kinds of data. For example, when used on data set "enron" and "genbase", the result is not as good as on other data sets. Sometimes it even got a worse result than BR-kNN. For example, when used on "enron" and evaluated by the Hamming Loss, our supposed CML-kNN only achieved a *4th* rank(0.061), while BR-kNN can get a second well result(0.052). The reason may be because the weak or loose connection between different labels in those data set, and our extended neighbors may introduce more noisy information than useful information. But in terms of average performance, our method performs the best (the first rank).

6.4 Summary

ML-kNN learns a single classifier h_i for each label l_i independently, so it is actually a binary relevance classifier. In other words, it does not consider the correlations between different labels. The algorithm is often criticized for this drawback. In this paper, we introduced a coupled label similarity, which explores the inner-relationship between different labels in multi-label classification according to their natural co-occupance. This similarity reflects the distance of the different labels. Furthermore, by integrating this similarity

into the multi-label kNN algorithm, we overcome the ML-kNN's shortcoming and improved the performance. Evaluated over three commonly-used multi-label data sets and in terms of Hamming Loss, One Error and Average Precision, the proposed method outperforms ML-KNN, BR-kNN, IBLR and even BSVM. This result shows that our supposed coupled label similarity is appropriate for multi-label learning problems and can work more effectively than other methods.

[***Note***] A conference version of this chapter has been accepted already and will be published by **PAKDD2015** as below:

- **Chunming Liu**, Longbing Cao (2015) A Coupled k-Nearest Neighbor Algorithm for Multi-label Classification. The 19th Pacific-Asia Conference on Knowledge Discovery and Data Mining (**PAKDD15**), full paper accepted. (**ERA ranking: A**)

Table 6.6: Experiment Data Sets for Multi-Label Classification

Data Set	|M|	f(M)	La(M)	LC(M)	LD(M)	DL(M)	PDL(M)	F(M)
emotions	593	72	6	1.869	0.311	27	0.046	numerical
yeast	2417	103	14	4.237	0.303	198	0.082	numerical
image	2000	294	5	1.236	0.247	20	0.010	numerical
scene	2407	294	6	1.074	0.179	15	0.006	numerical
enron	1702	1001	53	3.378	0.064	753	0.442	categorical
genbase	662	1185	27	1.252	0.046	32	0.048	categorical
medical	978	1449	45	1.245	0.028	94	0.096	categorical
bibtex	7395	1836	159	2.402	0.015	2856	0.386	categorical

Table 6.7: Experiment Result1 - Hamming Loss↓

	CML-kNN	**BR-kNN**	**ML-kNN**	**IBLR**	**BSVM**
emotions	0.189(1)	0.219(5)	0.194(2)	0.201(4)	0.199(3)
yeast	0.194(1)	0.205(5)	0.195(2)	0.198(3)	0.199(4)
image	0.157(1)	0.189(5)	0.172(2)	0.182(4)	0.176(3)
scene	0.078(1)	0.152(5)	0.084(2)	0.089(3)	0.104(4)
enron	0.061(4)	0.052(2)	0.052(2)	0.064(5)	0.047(1)
genbase	0.003(2)	0.004(3)	0.005(4)	0.005(4)	0.001(1)
medical	0.013(1)	0.019(4)	0.016(3)	0.026(5)	0.013(1)
bibtex	0.013(1)	0.016(4)	0.014(2)	0.016(4)	0.015(3)
AvgRank	**(1.50)**	4.13	2.38	4.00	2.50

Table 6.8: Experiment Result2 - One Error↓

	CML-kNN	**BR-kNN**	**ML-kNN**	**IBLR**	**BSVM**
emotions	0.244(1)	0.318(5)	0.263(3)	0.279(4)	0.253(2)
yeast	0.222(1)	0.235(4)	0.228(2)	0.237(5)	0.232(3)
image	0.267(1)	0.601(5)	0.319(3)	0.432(4)	0.314(2)
scene	0.197(1)	0.821(5)	0.219(2)	0.235(3)	0.251(4)
enron	0.308(3)	0.237(1)	0.313(4)	0.469(5)	0.245(2)
genbase	0.008(2)	0.012(5)	0.009(3)	0.011(4)	0.002(1)
medical	0.158(2)	0.327(4)	0.252(3)	0.414(5)	0.151(1)
bibtex	0.376(1)	0.631(5)	0.589(3)	0.576(2)	0.599(4)
AvgRank	**(1.50)**	4.25	2.88	4.00	2.38

Table 6.9: Experiment Result3 - Average Precision↑

	CML-kNN	**BR-kNN**	**ML-kNN**	**IBLR**	**BSVM**
emotions	0.819(1)	0.595(5)	0.799(3)	0.798(4)	0.807(2)
yeast	0.769(1)	0.596(5)	0.765(2)	0.759(3)	0.749(4)
image	0.824(1)	0.601(5)	0.792(3)	0.761(4)	0.796(2)
scene	0.885(1)	0.651(5)	0.869(2)	0.862(3)	0.849(4)
enron	0.591(3)	0.435(5)	0.626(2)	0.564(4)	0.702(1)
genbase	0.994(3)	0.992(4)	0.989(5)	0.994(2)	0.998(1)
medical	0.876(1)	0.782(4)	0.806(3)	0.686(5)	0.871(2)
bibtex	0.567(1)	0.329(5)	0.351(4)	0.476(3)	0.531(2)
AvgRank	**(1.50)**	4.75	3.00	3.50	2.25

Chapter 7

Conclusions and Future Work

7.1 Conclusions

In this thesis, we first introduce the research background of our topic, and then in Chapter 2, we reviewed the related research of the research peers have done and the problems and challenges in current strategy. In order to get a good view of our topic, we elaborate the coupling method from four aspects in the following Chapters: coupling in categorical data, coupling in numerical data, coupling in mixed type data and coupling in multi-label data. For every aspect, we first analysis the existing problems and then explained in detail of our solution to the issue. Supported by massive experiments on benchmark data sets, we proved the effectiveness of our coupling strategy in all these four aspects of Supervised Learning.

More specifically, in Chapter 3, we propose a new similarity measure to classify the class imbalanced categorical data which has strong relationships between objects, attributes and classes. By taking the class weight and attribute weight into account, this coupled similarity measure effectively extracts the inter-feature coupling and intra-feature coupling relationships from categorical attributes, as this can reveal the true interaction between attributes. So the distance between instances will be closer to the real situation compared to the old strategy, and hence will make the instances of

each class more compact in relation to each other. This will make it easier to distinguish and bring benefits in handling class-imbalance issues. The experiment results verified that our supposed method has a more stable and higher average performance than the classic algorithms, such as kNN, kENN, CCWkNN, SMOTE-based kNN, Decision Tree and NaiveBayes, especially when applied on imbalanced categorical data.

While not like categorical data, for numerical data, the popular Euclidean distance or Minkowski distance assumes that the attributes in the data set are independent of each other and have no outer connections. From the analysis in this thesis, we know that this will cause some problems in expressing the real distance between instances and may affect the accuracy of the classifiers. To overcome this shortcoming, in Chapter 4, we introduce a coupled relationship (intra-feature coupling relationship and inter-feature coupling relationship) for objects with numerical attributes, to consider the inner relationship in and between the continuous attributes. We present the coupling distance by calculating the relationship between the discrete groups of different features. Substantial experiments verify that our coupled distance on numerical data outperforms the Euclidean distance and Minkowski distance on several verification metrics. The experiment result and the statistical analysis demonstrate that our coupling strategy greatly improves upon the expression of the real distance of objects which have numerical features.

In order to make our coupling strategy more suitable for more complex situations, we propose a new method in Chapter 5 which applies our coupling concept to mixed type data, that is to say, data that contains both numerical and categorical features. In this supposed method, we do discretization first on numerical attributes to transfer such continuous values into separate groups, so as to adapt the coupling distance as we do on categorical features, then integrate this feature-coupling distance with the Euclidean distance, to overcome the shortcoming of the previous algorithms. Compared to the basic and variant nature of nearest neighbor algorithms, the experiment results show improvement and further verify the effectiveness of such a coupling

strategy.

We also extend our coupling concept to multi-label classification. As we already know, the traditional single-label classifiers are not suitable for multi-label tasks, due to the overlap of the class labels. The most used multi-label classifier, ML-kNN, as a binary relevance classifier, does not consider the relationship between different labels. To improve the performance of ML-kNN, we introduce a coupled label similarity in Chapter 6, which explores the inner relationship between different labels in multi-label classification according to their natural co-occurrence. Based on this coupling label similarity, we extend the concept of nearest neighbors to a new one. This extended coupling nearest neighbors overcomes one of the drawbacks of ML-kNN, which guarantees that there will be some similar instances in the neighborhood even in extreme situations. After applying this strategy into the multi-label kNN algorithm, we improve the performance significantly. As evaluated by three commonly-used multi-label classifier evaluation criteria, Hamming Loss, One Error and Average Precision, respectively, the experiment result indicates that our proposed method outperforms ML-KNN, BR-kNN and even IBLR. This proves that the coupling label similarity is appropriate for the multi-label learning problems.

Each chapter (i.e. from Chapter 3 to Chapter 6) of this thesis is supported by one published or submitted conference paper[1] listed in Appendix A (List of Publications). More encouragingly, several novel methods proposed in this thesis have been successfully applied in other topics and domains, such as document analysis, fraud detection and Big Data Analysis, with relevant papers recognized by research peers. Therefore, what we have done and proposed in this thesis is of great significance to the supervised learning related research, and the coupled similarity exposes the intrinsic structure and essential nature of problems and applications.

[1]The conference version papers of chapter 3, 5 and 6 are accepted or published, the paper of chapter 4 is still under review.

7.2 Future Work

From the previous discussion, we know that in real world classification tasks, assuming that the features are independent of each other then ignoring the interaction between them is wrong, and this applies no matter whether in balanced classification or in imbalanced classification, or even in multi-label tasks. As a solution to this problem, we propose a coupling similarity (or distance) to capture the inter-feature coupling relationship and the inner-feature relationship in or between attributes and labels. Experiment result and relative analysis proves the usefulness and improvement of our method in many tasks, such as for categorical data, numerical data, mixed type data, imbalanced data and multi-label data. However, as this thesis only focuses on part of the issue, there is still lots of work that needs to be done in the future.

For the coupling similarity we have supposed, three problems should be focused on. The first is that the coupling concept in our strategy only considers the inter-relationship between two features, so what if we were to take part in three, four or even more features when calculating the inter-feature coupling similarity? Will the new strategy bring more accuracy or just turn out to be a disaster.

Another area we need to look at is how to evaluate the weight of the inter-coupling relationship in the whole similarity calculation process. As in some real world data sets, such as some bank data and tax revenue data, we have found that the improvement is not as significant as it is in some other data sets. The reason may be that such data has a weak inter-relationship between features. If we assign a high weight to the inter-coupling, it will produce a large amount of noise to the real similarity or distance. Therefore how to evaluate the importance of the inter-coupling similarity is a problem and hence how to learn the weight of inter-coupling effectively while not just using the exhaustive method is another issue we should consider in future work.

The third area we need to do in the future is to extend the current feature

value based coupling similarity to the object based coupling similarity. Currently, we have only considered the relationship of the inner or inter attribute values. But in the real world, such as a social-networking, one person may have a special connection to another person, and that connection would not only display the same age, hobby, job, or religion, but maybe some hidden features which are hard to obtain. In such cases, only considering the similarity revealed by the features will not be enough for the learning. So will the concept of objects-coupling bring some improvement for these kinds of supervised learning? Further work needs to be done in the future.

For the class-imbalance questions, there is still some unfinished work. For example, in this thesis, we only combined our coupling similarity with the kNN algorithm. How will the performance be if we apply the coupling strategy to other algorithms, such as the SVM algorithm? Will the transfer in the SVM damage the original relationship between features or will the transfer bring new relationships? The time complexity is another issue we should take into account.

For the multi-label problems, we only consider the coupling relationship between different class labels. In future work, we should take into account the influence of different features as we have done in single-label supervised learning tasks. Furthermore, how to control time complexity is a big issue that we should think about.

Appendix A

Appendix: List of Publications

Papers Published

- **Chunming Liu**, Longbing Cao (2015), A Coupled k-Nearest Neighbor Algorithm for Multi-label Classification. The 19th Pacific-Asia Conference on Knowledge Discovery and Data Mining (**PAKDD15**), full paper accepted. (**ERA ranking: A**)

- **Chunming Liu**, Longbing Cao, Philip S Yu (2014), Coupled fuzzy k-nearest neighbors classification of imbalanced non-IID categorical data. *In* 'Proceedings of the Neural Networks (IJCNN), 2014 International Joint Conference on (**WCCI14**)', IEEE, pp. 1122-1129. (**ERA ranking: A**)

- **Chunming Liu**, Longbing Cao, Philip S Yu (2014), A hybrid coupled k-nearest neighbor algorithm on imbalance data. *In* 'Proceedings of the Neural Networks (IJCNN), 2014 International Joint Conference on (**WCCI14**)', IEEE, pp. 2011-2018. (**ERA ranking: A**)

- Jinjiu Li, Can Wang, Wei Wei, Mu Li, **Chunming Liu** (2013), Efficient Mining of Contrast Patterns on Large Scale Imbalanced Real-Life Data. *In* 'Proceedings of the 17th pacific-asia conference on knowledge discovery and data mining (**PAKDD13**)' , pp. 62-73. [**Best Student Paper Award**] (**ERA ranking: A**)

- Wei Wei, Jinyan Li, Longbing Cao, Jingguang Sun, **Chunming Liu**, Mu Li (2013), Optimal Allocation of High Dimensional Assets through Canonical Vines. *In* 'Proceedings of the 17th pacific-asia conference on knowledge discovery and data mining (**PAKDD13**)' , pp. 366-377. (**ERA ranking: A**)

- Zhigang Zheng, Wei Wei, **Chunming Liu**, Wei Cao, Longbing Cao, and Maninder Bhatia (2015) An Effective Contrast Sequential Pattern Mining Approach on Taxpayer Behavior Analysis, full paper accepted. *World Wide Web Journal*.

Papers Submitted and Under Review

- **Chunming Liu**, Mu Li, Longbing Cao (2015) Coupling based Classification for Numerical Data, submitted to and been reviewed by the IEEE International Conference on Data Mining series (**ICDM15**).

- **Chunming Liu**, Longbing Cao (2015) Coupled Similarity in Imbalanced Data Classification, submitted to and been reviewed by the IEEE International Conference on Data Mining series (**ICDM15**).

- Mu Li, **Chunming Liu**, Longbing Cao (2015) Learning Large-Scale Coupling Relationships in BigData, submitted to and been reviewed by the 2015 IEEE International Conference on Data Science and Advanced Analytics (**IEEE DSAA15**).

Research Reports of Industry Projects

- **Chunming Liu**, Zhigang Zheng, Wei Cao, Maninder Bhatia. Action response model of IT – data mining modeling and evaluation report, Debt Collection and Optimisation Project, Australian Taxation Office(ATO), Oct 2014.

Appendix B

Appendix: List of Symbols

The following list is neither exhaustive nor exclusive, but may be helpful.

D	The data set, $D = \{x_1, x_2, \cdots, x_m\}$
U	The data set, $U = \{u_1, u_2, \cdots, u_m\}$
A	A set of features, $A = \{a_1, a_2, \cdots, a_n\}$
C	A set of class labels, $C = \{c_1, c_2, \cdots, c_l\}$
Y	A set of class labels
$S(\cdot)$	The similarity between two objects
E^i or E^j	An instance of D
k	The number of nearest neighbors
TS	The training set
T	Number of iterations
I	Weak learner
H_j	Denotes that x has label y_j

$P(H_j\|C_j)$	The posterior probability that H_j holds under the condition that x has exactly C_j neighbors with label y_j
s	A smoothing parameter controlling the effect of uniform prior on the estimation
k_j and k'_j	Frequency arrays
V_i and V_j	The instances' corresponding frequency vectors
u_i	An instance from D
$\theta(c_l)$	The Class Size Weight of c_l
α_j	The feature weight of feature j
$RF(v_j^x)$	The occurrence frequency of values v_j^x
$\delta^{Ie}(x_1, x_2)$	The inter-coupling similarity between attribute x_1 and x_2
$\delta^{Ia}(x_1, x_2)$	The intra-coupling similarity between attribute x_1 and x_2
$AS(\cdot)$	The Adapted Coupled Object Similarity
$IS(\cdot)$	The Integrated Similarity
β	The parameter that decides the weight of intra-coupled similarity
$v_j^{i_1}$ and $v_j^{i_2}$	The values of feature j for instances u_{i_1} and u_{i_2}
γ_{jk}	The Feature-Pair Weight
IR	The imbalance rate
$d(\cdot)$	The distance between two points
$R(a_s, a_t)$	The Pearson's product-moment correlation coefficient between attribute a_s and a_t

θ_s and θ_t	The mean of the sth and tth attribute
x_{is} and x_{it}	The sth and tth components of instance x_i
$\mu_{pq}(j)$	The Pearson's correlation coefficient between β_j^p and β_j^q
$R^{Intra}(a_j)$	The Intra-coupled similarity within numerical attribute a_j
$R^{Inter}(a_j\|\{a_k\})$	The Inter-coupled similarity between attribute a_j and other attribute $a_k(k \neq j)$
S'	The coupled information table
ξ	A $3 \times 3(n-1)$ constant matrix and all elements are equal 1
$Fre(x_{ij}, R^{C(u_i)})$	A frequency count function that count the occurrences of x_{ij} in feature j of set $R^{C(u_i)}$
T	The multi-label training data set, $T = \{(x_1, L(x_1)), \ldots, (x_m, L(x_m))\}$ $(\|T\| = m)$
$Round()$	The rounding function
$\delta_{L_i^*\|j}$	The sum of the CLS values of the i-th neighbor's label set to the j-th label l_j
P or Q	A point in Euclidean Space, $P, Q \in \mathbb{R}$
V^1 or V^2	A vector in Euclidean Space
Δ	The symmetric difference of two sets and corresponds to the XOR operation in Boolean logic

Bibliography

Abe, N., Zadrozny, B. & Langford, J. (2004), An iterative method for multi-class cost-sensitive learning, *in* 'Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 3–11.

Ahmad, A. & Dey, L. (2007*a*), 'A k-mean clustering algorithm for mixed numeric and categorical data', *Data and Knowledge Engineering* **63**(2), 503–527.

Ahmad, A. & Dey, L. (2007*b*), 'A method to compute distance between two categorical values of same attribute in unsupervised learning for categorical data set', *Pattern Recognition Letters* **28**(1), 110–118.

Alcalá, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L. & Herrera, F. (2010), 'Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework', *Journal of Multiple-Valued Logic and Soft Computing* **17**, 255–287.

Bache, K. & Lichman, M. (2013), 'UCI machine learning repository'.

Batista, G. E., Prati, R. C. & Monard, M. C. (2004), 'A study of the behavior of several methods for balancing machine learning training data', *ACM Sigkdd Explorations Newsletter* **6**(1), 20–29.

Bermejo, P., Gámez, J. A. & Puerta, J. M. (2011), 'Improving the performance of naive bayes multinomial in e-mail foldering by introducing

distribution-based balance of datasets', *Expert Systems with Applications* **38**(3), 2072–2080.

Bezdek, J. C. (1981), *Pattern recognition with fuzzy objective function algorithms*, Kluwer Academic Publishers.

Bezdek, J. C. & Pal, S. K. (1992), *Fuzzy models for pattern recognition*, Vol. 56, IEEE Press, New York.

Błaszczyński, J., Deckert, M., Stefanowski, J. & Wilk, S. (2010), Integrating selective pre-processing of imbalanced data with ivotes ensemble, *in* 'Rough Sets and Current Trends in Computing', Springer, pp. 148–157.

Boriah, S., Chandola, V. & Kumar, V. (2008), 'Similarity measures for categorical data: A comparative evaluation', *red* **30**(2), 3.

Boutell, M. R., Luo, J., Shen, X. & Brown, C. M. (2004), 'Learning multi-label scene classification', *Pattern recognition* **37**(9), 1757–1771.

Bradford, J. P., Kunz, C., Kohavi, R., Brunk, C. & Brodley, C. E. (1998), Pruning decision trees with misclassification costs, *in* 'Machine Learning: ECML-98', Springer, pp. 131–136.

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

Breiman, L. (1996), 'Bagging predictors', *Machine learning* **24**(2), 123–140.

Breiman, L. (1999), 'Pasting small votes for classification in large databases and on-line', *Machine Learning* **36**(1-2), 85–103.

Brinker, K. & Hüllermeier, E. (2007), Case-based multilabel ranking., *in* 'IJCAI', pp. 702–707.

Brow, G., Wyatt, J., Harris, R. & Yao, X. (2005), 'Diversity creation methods: A survey and categorization', *Journal of Information Fusion* **1**, 1–28.

Burnaby, T. (1970), 'On a method for character weighting a similarity coefficient, employing the concept of information', *Journal of the International Association for Mathematical Geology* **2**(1), 25–38.

Calders, T., Goethals, B. & Jaroszewicz, S. (2006), Mining rank-correlated sets of numerical attributes, *in* 'Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 96–105.

Cao, L. (2014*a*), 'Coupling learning of complex interactions', *Information Processing & Management* .

Cao, L. (2014*b*), 'Non-iidness learning in behavioral and social data', *The Computer Journal* **57**(9), 1358–1370.

Cao, L., Ou, Y. & Yu, P. S. (2012), 'Coupled behavior analysis with applications', *Knowledge and Data Engineering, IEEE Transactions on* **24**(8), 1378–1392.

Celebi, M. E., Kingravi, H. A., Uddin, B., Iyatomi, H., Aslandogan, Y. A., Stoecker, W. V. & Moss, R. H. (2007), 'A methodological approach to the classification of dermoscopy images', *Computerized Medical Imaging and Graphics* **31**(6), 362–373.

Chawla, N., Bowyer, K., Hall, L. & Kegelmeyer, W. (2002), 'SMOTE: synthetic minority over-sampling technique', *Journal of Artificial Intelligence Research* **16**(1), 321–357.

Chawla, N. V., Cieslak, D. A., Hall, L. O. & Joshi, A. (2008), 'Automatically countering imbalance and its empirical relationship to cost', *Data Mining and Knowledge Discovery* **17**(2), 225–252.

Chawla, N. V., Japkowicz, N. & Kotcz, A. (2004), 'Editorial: special issue on learning from imbalanced data sets', *ACM Sigkdd Explorations Newsletter* **6**(1), 1–6.

Chawla, N. V., Lazarevic, A., Hall, L. O. & Bowyer, K. W. (2003), Smoteboost: Improving prediction of the minority class in boosting, *in* 'Knowledge Discovery in Databases: PKDD 2003', Springer, pp. 107–119.

Cheng, V., Li, C.-H., Kwok, J. T. & Li, C.-K. (2004), 'Dissimilarity learning for nominal data', *Pattern Recognition* **37**(7), 1471–1477.

Cheng, W. & Hüllermeier, E. (2009), 'Combining instance-based learning and logistic regression for multilabel classification', *Machine Learning* **76**(2-3), 211–225.

Cieslak, D. A., Chawla, N. V. & Striegel, A. (2006), Combating imbalance in network intrusion datasets., *in* 'GrC', pp. 732–737.

Clare, A. & King, R. D. (2001), Knowledge discovery in multi-label phenotype data, *in* 'Principles of data mining and knowledge discovery', Springer, pp. 42–53.

Cost, S. & Salzberg, S. (1993), 'A weighted nearest neighbor algorithm for learning with symbolic features', *Machine learning* **10**(1), 57–78.

Cover, T. & Hart, P. (1967), 'Nearest neighbor pattern classification', *Information Theory, IEEE Transactions on* **13**(1), 21–27.

Cover, T. M. (1991), 'Ja thomas elements of information theory'.

Das, G. & Mannila, H. (2000), Context-based similarity measures for categorical databases, *in* 'Principles of Data Mining and Knowledge Discovery', Springer, pp. 201–210.

Diplaris, S., Tsoumakas, G., Mitkas, P. A. & Vlahavas, I. (2005), Protein classification with multiple algorithms, *in* 'Advances in Informatics', Springer, pp. 448–456.

Domingos, P. (1999), Metacost: A general method for making classifiers cost-sensitive, *in* 'Proceedings of the fifth ACM SIGKDD international

conference on Knowledge discovery and data mining', ACM, pp. 155–164.

Elisseeff, A. & Weston, J. (2001), A kernel method for multi-labelled classification., *in* 'NIPS', Vol. 14, pp. 681–687.

Estabrooks, A., Jo, T. & Japkowicz, N. (2004), 'A multiple resampling method for learning from imbalanced data sets', *Computational Intelligence* **20**(1), 18–36.

Fawcett, T. (2006), 'An introduction to ROC analysis', *Pattern recognition letters* **27**(8), 861–874.

Fernández, A., García, S., del Jesus, M. J. & Herrera, F. (2008), 'A study of the behaviour of linguistic fuzzy rule based classification systems in the framework of imbalanced data-sets', *Fuzzy Sets and Systems* **159**(18), 2378–2398.

Ferri, C., Flach, P. & Hernández-Orallo, J. (2002), Learning decision trees using the area under the roc curve, *in* 'ICML', Vol. 2, pp. 139–146.

Figueiredo, F., Rocha, L., Couto, T., Salles, T., Gonçalves, M. A. & Meira Jr, W. (2011), 'Word co-occurrence features for text classification', *Information Systems* **36**(5), 843–858.

Fix, E. & Hodges Jr, J. L. (1951), Discriminatory analysis-nonparametric discrimination: consistency properties, Technical report, DTIC Document.

Freitas, A., Costa-Pereira, A. & Brazdil, P. (2007), Cost-sensitive decision trees applied to medical data, *in* 'Data Warehousing and Knowledge Discovery', Springer, pp. 303–312.

Freund, Y. & Schapire, R. E. (1995), A desicion-theoretic generalization of on-line learning and an application to boosting, *in* 'Computational learning theory', Springer, pp. 23–37.

Friedman, J., Hastie, T., Tibshirani, R. et al. (2000), 'Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors)', *The annals of statistics* **28**(2), 337–407.

Fürnkranz, J., Hüllermeier, E., Mencía, E. L. & Brinker, K. (2008), 'Multilabel classification via calibrated label ranking', *Machine Learning* **73**(2), 133–153.

Gambaryan, P. (1964), 'A mathematical model of taxonomy', *Izvest. Akad. Nauk Armen. SSR* **17**(12), 47–53.

Gan, G., Ma, C. & Wu, J. (2007), *Data clustering: theory, algorithms, and applications*, Vol. 20, Siam.

Gonçalves, T. & Quaresma, P. (2003), A preliminary approach to the multilabel classification problem of portuguese juridical documents, *in* 'Progress in Artificial Intelligence', Springer, pp. 435–444.

Goodall, D. W. (1966), 'A new similarity index based on probability', *Biometrics* pp. 882–907.

Gower, J. C. & Legendre, P. (1986), 'Metric and euclidean properties of dissimilarity coefficients', *Journal of classification* **3**(1), 5–48.

Green, D. M., Swets, J. A. et al. (1966), *Signal detection theory and psychophysics*, Vol. 1, Wiley New York.

Gunduz, A. & Principe, J. C. (2009), 'Correntropy as a novel measure for nonlinearity tests', *Signal Processing* **89**(1), 14–23.

Hassan, M. R., Hossain, M. M., Bailey, J. & Ramamohanarao, K. (2008), Improving k-nearest neighbour classification with distance functions based on receiver operating characteristics, *in* 'Machine Learning and Knowledge Discovery in Databases', Springer, pp. 489–504.

Hastie, T. & Tibshirani, R. (1996), 'Discriminant adaptive nearest neighbor classification', *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **18**(6), 607–616.

Hossain, M. M., Hassan, M. R. & Bailey, J. (2008), Roc-tree: A novel decision tree induction algorithm based on receiver operating characteristics to classify gene expression data., *in* 'SDM', SIAM, pp. 455–465.

Hu, S., Liang, Y., Ma, L. & He, Y. (2009), Msmote: improving classification performance when training data is imbalanced, *in* 'Computer Science and Engineering, 2009. WCSE'09. Second International Workshop on', Vol. 2, IEEE, pp. 13–17.

Hu, X. (2001), Using rough sets theory and database operations to construct a good ensemble of classifiers for data mining applications, *in* 'Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on', IEEE, pp. 233–240.

Huang, J. & Ling, C. X. (2005), 'Using auc and accuracy in evaluating learning algorithms', *Knowledge and Data Engineering, IEEE Transactions on* **17**(3), 299–310.

Huang, Y.-M., Hung, C.-M. & Jiau, H. C. (2006), 'Evaluation of neural networks and data mining methods on a credit assessment task for class imbalance problem', *Nonlinear Analysis: Real World Applications* **7**(4), 720–747.

Ienco, D., Pensa, R. G. & Meo, R. (2012), 'From context to distance: Learning dissimilarity for categorical data clustering', *ACM Transactions on Knowledge Discovery from Data (TKDD)* **6**(1), 1.

Jaccard, P. (1912), 'The distribution of the flora in the alpine zone. 1', *New phytologist* **11**(2), 37–50.

Jakulin, A. & Bratko, I. (2003), *Analyzing attribute dependencies*, Springer.

Japkowicz, N. (2001), 'Supervised versus unsupervised binary-learning by feedforward neural networks', *Machine Learning* **42**(1-2), 97–122.

Jones, K. S. (1972), 'A statistical interpretation of term specificity and its application in retrieval', *Journal of documentation* **28**(1), 11–21.

Kantardzic, M. (2011), *Data mining: concepts, models, methods, and algorithms*, John Wiley & Sons.

Keller, J. M., Gray, M. R. & Givens, J. A. (1985), 'A fuzzy k-nearest neighbor algorithm', *Systems, Man and Cybernetics, IEEE Transactions on* **15**(4), 580–585.

Khorshidpour, Z., Hashemi, S. & Hamzeh, A. (2010), Distance learning for categorical attribute based on context information, *in* 'Software Technology and Engineering (ICSTE), 2010 2nd International Conference on', Vol. 2, IEEE, pp. V2–296.

Khreich, W., Granger, E., Miri, A. & Sabourin, R. (2010), 'Iterative boolean combination of classifiers in the roc space: An application to anomaly detection with hmms', *Pattern Recognition* **43**(8), 2732–2752.

Kiliç, K., Uncu, Ö. & Türksen, I. B. (2007), 'Comparison of different strategies of utilizing fuzzy clustering in structure identification', *Information Sciences* **177**(23), 5153–5162.

Klir, G. & Yuan, B. (1995), *Fuzzy sets and fuzzy logic*, Vol. 4, Prentice Hall New Jersey.

Kotsiantis, S. & Pintelas, P. (2003), 'Mixture of expert agents for handling imbalanced data sets', *Annals of Mathematics, Computing & Teleinformatics* **1**(1), 46–55.

Krogh, A., Vedelsby, J. et al. (1995), 'Neural network ensembles, cross validation, and active learning', *Advances in neural information processing systems* pp. 231–238.

Kubat, M., Holte, R. C. & Matwin, S. (1998), 'Machine learning for the detection of oil spills in satellite radar images', *Machine learning* **30**(2-3), 195–215.

Kubat, M., Matwin, S. et al. (1997), Addressing the curse of imbalanced training sets: one-sided selection, *in* 'ICML', Vol. 97, Nashville, USA, pp. 179–186.

Kullback, S. (1959), 'Statistics and information theory'.

Kullback, S. & Leibler, R. A. (1951), 'On information and sufficiency', *The Annals of Mathematical Statistics* pp. 79–86.

Kuncheva, L. I. (2004), *Combining pattern classifiers: methods and algorithms*, John Wiley & Sons.

Kuncheva, L. I. (2005), 'Diversity in multiple classifier systems', *Information fusion* **6**(1), 3–4.

Kurgan, L. A. & Cios, K. J. (2004), 'Caim discretization algorithm', *Knowledge and Data Engineering, IEEE Transactions on* **16**(2), 145–153.

Lauser, B. & Hotho, A. (2003), Automatic multi-label subject indexing in a multilingual environment, *in* 'Research and Advanced Technology for Digital Libraries', Springer, pp. 140–151.

Le, S. Q. & Ho, T. B. (2005), 'An association-based dissimilarity measure for categorical data', *Pattern Recognition Letters* **26**(16), 2549–2557.

Lin, D. (1998), An information-theoretic definition of similarity., *in* 'ICML', Vol. 98, pp. 296–304.

Lin, Y., Lee, Y. & Wahba, G. (2002), 'Support vector machines for classification in nonstandard situations', *Machine Learning* **46**(1-3), 191–202.

Ling, C. X., Yang, Q., Wang, J. & Zhang, S. (2004), Decision trees with minimal costs, *in* 'Proceedings of the twenty-first international conference on Machine learning', ACM, p. 69.

Liu, B., Ma, Y. & Wong, C. K. (2000), Improving an association rule based classifier, *in* 'Principles of Data Mining and Knowledge Discovery', Springer, pp. 504–509.

Liu, C., Cao, L. & Yu, P. S. (2014*a*), Coupled fuzzy k-nearest neighbors classification of imbalanced non-iid categorical data, *in* 'Neural Networks (IJCNN), 2014 International Joint Conference on', IEEE, pp. 1122–1129.

Liu, C., Cao, L. & Yu, P. S. (2014*b*), A hybrid coupled k-nearest neighbor algorithm on imbalance data, *in* 'Neural Networks (IJCNN), 2014 International Joint Conference on', IEEE, pp. 2011–2018.

Liu, W. & Chawla, S. (2011), 'Class confidence weighted knn algorithms for imbalanced data sets', *Advances in Knowledge Discovery and Data Mining* pp. 345–356.

Liu, W., Chawla, S., Cieslak, D. & Chawla, N. (2010), A robust decision tree algorithm for imbalanced data sets, *in* 'SDM 2010', pp. 766–777.

Liu, X.-Y., Wu, J. & Zhou, Z.-H. (2009), 'Exploratory undersampling for class-imbalance learning', *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* **39**(2), 539–550.

Liu, Y.-H. & Chen, Y.-T. (2005), Total margin based adaptive fuzzy support vector machines for multiview face recognition, *in* 'Systems, Man and Cybernetics, 2005 IEEE International Conference on', Vol. 2, IEEE, pp. 1704–1711.

López, V., Fernández, A., García, S., Palade, V. & Herrera, F. (2013), 'An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics', *Information Sciences* **250**, 113–141.

Lu, W.-Z. & Wang, D. (2008), 'Ground-level ozone prediction by support vector machine approach with a cost-sensitive classification scheme', *Science of the total environment* **395**(2), 109–116.

M. Kaytoue, S. K. & Napoli, A. (2011), Revisiting numerical pattern mining with formal concept analysis, *in* 'IJCAI', pp. 1342–1347.

Mamitsuka, H. (2006), 'Selecting features in microarray classification using roc curves', *Pattern Recognition* **39**(12), 2393–2404.

Manevitz, L. M. & Yousef, M. (2002), 'One-class svms for document classification', *the Journal of machine Learning research* **2**, 139–154.

Margineantu, D. D. (2002), Class probability estimation and cost-sensitive classification decisions, *in* 'Machine Learning: ECML 2002', Springer, pp. 270–281.

Mazurowski, M. A., Habas, P. A., Zurada, J. M., Lo, J. Y., Baker, J. A. & Tourassi, G. D. (2008), 'Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance', *Neural networks* **21**(2), 427–436.

McCallum, A. (1999), Multi-label text classification with a mixture model trained by em, *in* 'AAAI99 Workshop on Text Learning', pp. 1–7.

Michalski, R. S. (1980), 'Knowledge acquisition through conceptual clustering: A theoretical framework and an algorithm for partitioning data into conjunctive concepts', *Journal of Policy Analysis and Information Systems* **4**(3), 219–244.

Oza, N. C. & Tumer, K. (2008), 'Classifier ensembles: Select real-world applications', *Information Fusion* **9**(1), 4–20.

Pang-Ning, T., Steinbach, M., Kumar, V. et al. (2006), Introduction to data mining, *in* 'Library of Congress', p. 74.

Peng, X. & King, I. (2008), 'Robust bmpm training based on second-order cone programming and its application in medical diagnosis', *Neural Networks* **21**(2), 450–457.

Plant, C. (2012), Dependency clustering across measurement scales, *in* 'Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 361–369.

Polikar, R. (2006), 'Ensemble based systems in decision making', *Circuits and Systems Magazine, IEEE* **6**(3), 21–45.

Quinlan, J. R. (1991), 'Improved estimates for the accuracy of small disjuncts', *Machine Learning* **6**(1), 93–98.

Raeder, T., Hoens, T. R. & Chawla, N. V. (2010), Consequences of variability in classifier performance estimates, *in* 'Data Mining (ICDM), 2010 IEEE 10th International Conference on', IEEE, pp. 421–430.

Read, J., Pfahringer, B., Holmes, G. & Frank, E. (2011), 'Classifier chains for multi-label classification', *Machine learning* **85**(3), 333–359.

Riddle, P., Segal, R. & Etzioni, O. (1994), 'Representation design and brute-force induction in a boeing manufacturing domain', *Applied Artificial Intelligence an International Journal* **8**(1), 125–147.

Rokach, L. (2009), 'Taxonomy for characterizing ensemble methods in classification tasks: A review and annotated bibliography', *Computational Statistics & Data Analysis* **53**(12), 4046–4072.

Rokach, L. (2010), 'Ensemble-based classifiers', *Artificial Intelligence Review* **33**(1-2), 1–39.

Ross, T. J. (2009), *Fuzzy Logic with Engineering Applications*, John Wiley & Sons.

Rudin, C., Daubechies, I. & Schapire, R. E. (2004), 'The dynamics of adaboost: Cyclic behavior and convergence of margins', *The Journal of Machine Learning Research* **5**, 1557–1595.

Schapire, R. E. (1990), 'The strength of weak learnability', *Machine learning* **5**(2), 197–227.

Schapire, R. E. & Singer, Y. (2000), 'Boostexter: A boosting-based system for text categorization', *Machine learning* **39**(2-3), 135–168.

Seiffert, C., Khoshgoftaar, T. M., Van Hulse, J. & Napolitano, A. (2010), 'Rusboost: A hybrid approach to alleviating class imbalance', *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on* **40**(1), 185–197.

Shannon, C. E. (2001), 'A mathematical theory of communication', *ACM SIGMOBILE Mobile Computing and Communications Review* **5**(1), 3–55.

Shepard, R. N. (1987), 'Toward a universal law of generalization for psychological science', *Science* **237**(4820), 1317–1323.

Silva, C., Lotric, U., Ribeiro, B. & Dobnikar, A. (2010), 'Distributed text classification with an ensemble kernel-based learning approach', *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* **40**(3), 287–297.

Smirnov, E. (1968), 'On exact methods in systematics', *Systematic Biology* **17**(1), 1–13.

Spyromitros, E., Tsoumakas, G. & Vlahavas, I. (2008), An empirical study of lazy multilabel classification algorithms, *in* 'Artificial Intelligence: Theories, Models and Applications', Springer, pp. 401–406.

Stanfill, C. & Waltz, D. (1986), 'Toward memory-based reasoning', *Communications of the ACM* **29**(12), 1213–1228.

Tavallaee, M., Stakhanova, N. & Ghorbani, A. A. (2010), 'Toward credible evaluation of anomaly-based intrusion-detection methods', *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* **40**(5), 516–524.

Theilhaber, J., Connolly, T., Roman-Roman, S., Bushnell, S., Jackson, A., Call, K., Garcia, T. & Baron, R. (2002), 'Finding genes in the c2c12 osteogenic pathway by k-nearest-neighbor classification of expression data', *Genome research* **12**(1), 165–176.

Tsoumakas, G. & Katakis, I. (2007), 'Multi-label classification: An overview', *International Journal of Data Warehousing and Mining (IJDWM)* **3**(3), 1–13.

Tsoumakas, G., Katakis, I. & Vlahavas, L. (2011), 'Random k-labelsets for multilabel classification', *Knowledge and Data Engineering, IEEE Transactions on* **23**(7), 1079–1089.

Tumer, K. & Ghosh, J. (1996), 'Error correlation and error reduction in ensemble classifiers', *Connection science* **8**(3-4), 385–404.

Ueda, N. & Nakano, R. (1996), Generalization error of ensemble estimators, *in* 'Neural Networks, 1996., IEEE International Conference on', Vol. 1, IEEE, pp. 90–95.

Vembu, S. & Gärtner, T. (2011), Label ranking algorithms: A survey, *in* 'Preference learning', Springer, pp. 45–64.

Wang, C., Cao, L., Wang, M., Li, J., Wei, W. & Ou, Y. (2011), Coupled nominal similarity in unsupervised learning, *in* 'Proceedings of the 20th ACM international conference on Information and knowledge management', ACM, pp. 973–978.

Wang, C., She, Z. & Cao, L. (2013), Coupled attribute analysis on numerical data, *in* 'Proceedings of the Twenty-Third international joint conference on Artificial Intelligence', AAAI Press, pp. 1736–1742.

Wang, P. P. & Chang, S. K. (1980), *Fuzzy Sets: Theory of Applications to Policy Analysis and Information Systems*, Springer.

Weiss, G. M. & Provost, F. J. (2003), 'Learning when training data are costly: the effect of class distribution on tree induction', *J. Artif. Intell. Res.(JAIR)* **19**, 315–354.

Wieczorkowska, A., Synak, P. & Raś, Z. W. (2006), Multi-label classification of emotions in music, *in* 'Intelligent Information Processing and Web Mining', Springer, pp. 307–315.

Williams, D. P., Myers, V. & Silvious, M. S. (2009), 'Mine classification with imbalanced data', *Geoscience and Remote Sensing Letters, IEEE* **6**(3), 528–532.

Wilson, D. R. & Martinez, T. R. (1997), 'Improved heterogeneous distance functions', *arXiv preprint cs/9701101* .

Wu, G. & Chang, E. Y. (2003), Class-boundary alignment for imbalanced dataset learning, *in* 'ICML 2003 workshop on learning from imbalanced data sets II, Washington, DC', pp. 49–56.

Wu, G. & Chang, E. Y. (2005), 'Kba: Kernel boundary alignment considering imbalanced data distribution', *Knowledge and Data Engineering, IEEE Transactions on* **17**(6), 786–795.

Wu, W., Xing, E. P., Myers, C., Mian, I. S. & Bissell, M. J. (2005), 'Evaluation of normalization methods for cdna microarray data by k-nn classification', *BMC bioinformatics* **6**(1), 191.

Wu, X. & Kumar, V. (2009), *The top ten algorithms in data mining*, CRC Press.

Xie, J., Szymanski, B. K. & Zaki, M. J. (2010), Learning dissimilarities for categorical symbols., *in* 'FSDM', pp. 97–106.

Xu, Y., Cao, X. & Qiao, H. (2011), 'An efficient tree classifier ensemble-based approach for pedestrian detection', *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* **41**(1), 107–117.

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

Yang, T., Cao, L. & Zhang, C. (2010), A novel prototype reduction method for the k-nearest neighbor algorithm with $k \geq 1$, *in* 'Advances in Knowledge Discovery and Data Mining', Springer, pp. 89–100.

Yang, Y. & Chen, K. (2011), 'Time series clustering via rpcl network ensemble with different representations', *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* **41**(2), 190–199.

Yang, Z., Tang, W., Shintemirov, A. & Wu, Q. (2009), 'Association rule mining-based dissolved gas analysis for fault diagnosis of power transformers', *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* **39**(6), 597–610.

Yuxuan, L. & Zhang, X. (2011), Improving k nearest neighbor with exemplar generalization for imbalanced classification, *in* '15th Pacific-Asia Conference, PAKDD 2011', Springer, pp. 1–12.

Zadeh, L. A. (1965), 'Fuzzy sets', *Information and control* **8**(3), 338–353.

Zadrozny, B. & Elkan, C. (2001), Learning and making decisions when costs and probabilities are both unknown, *in* 'Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 204–213.

Zadrozny, B., Langford, J. & Abe, N. (2003), Cost-sensitive learning by cost-proportionate example weighting, *in* 'Data Mining, 2003. ICDM 2003. Third IEEE International Conference on', IEEE, pp. 435–442.

Zhang, M.-L. & Zhou, Z.-H. (2006), 'Multilabel neural networks with applications to functional genomics and text categorization', *Knowledge and Data Engineering, IEEE Transactions on* **18**(10), 1338–1351.

Zhang, M.-L. & Zhou, Z.-H. (2007), 'Ml-knn: A lazy learning approach to multi-label learning', *Pattern recognition* **40**(7), 2038–2048.

Zheng, Z., Wu, X. & Srihari, R. (2004), 'Feature selection for text categorization on imbalanced data', *ACM SIGKDD Explorations Newsletter* **6**(1), 80–89.

Zhu, Z.-B. & Song, Z.-H. (2010), 'Fault diagnosis based on imbalance modified kernel fisher discriminant analysis', *Chemical Engineering Research and Design* **88**(8), 936–951.