Improving *k* Nearest Neighbor with Exemplar Generalization for Imbalanced Classification

Yuxuan Li and Xiuzhen Zhang

School of Computer Science and Information Technology,
RMIT University, Melbourne, Australia
`{li.yuxuan,xiuzhen.zhang}@rmit.edu.au`

Abstract. A k nearest neighbor (kNN) classifier classifies a query instance to the most frequent class of its k nearest neighbors in the training instance space. For imbalanced class distribution, a query instance is often overwhelmed by majority class instances in its neighborhood and likely to be classified to the majority class. We propose to identify exemplar minority class training instances and generalize them to Gaussian balls as concepts for the minority class. Our k Exemplar-based Nearest Neighbor (kENN) classifier is therefore more sensitive to the minority class. Extensive experiments show that kENN significantly improves the performance of kNN and also outperforms popular re-sampling and cost-sensitive learning strategies for imbalanced classification.

Keywords: Cost-sensitive learning, imbalanced learning, kNN, re-sampling.

1 Introduction

Skewed class distribution is common for many real-world classification problems, including for example, detection of software defects in software development projects [14], identification of oil spills in satellite radar images [11], and detection of fraudulent calls [9]. Typically the intention of classification learning is to achieve accurate classification for each class (especially the rare class) rather than an overall accuracy without distinguishing classes. In this paper our discussions will be focused on the two-class imbalanced classification problem, where the minority class is the positive and the majority class is the negative.

Imbalanced class distribution has been reported to impede the performance of many concept learning systems [20]. Many systems adopt the maximum generality bias [10] to induct a classification model, where the concept[1] of the least number of conditions (and therefore the most general concept) is chosen to describe a cluster of training instances. In the presence of class imbalance this induction bias tends to over-generalize concepts for the majority class and miss concepts for the minority class. In formulating decision trees [17] for example,

[1] The term concept is used in its general sense. Strictly in the context of classification learning it is a subconcept of the complete concept for some class.

induction may stop at a node where class for the node is decided by the majority of instances under the node and instances of the minority class are ignored.

In contrast to most concept learning systems, k nearest neighbor (kNN) classification [6,1,2] or instance-based learning, does not formulate a generalized conceptual model from the training instances at the training stage. Rather at the classification stage, a simple and intuitive rule is used to make decisions: instances close in the input space are likely to belong to the same class. Typically a kNN classifier classifies a query instance to the class that appears most frequently among its k nearest neighbors. k is a parameter for tuning the classification performance and is typically set to three to seven.

Although instance-based learning has been advocated for imbalanced learning [10,19,3], to the best of our knowledge, a large-scale study of applying kNN classification to imbalanced learning has not been reported in literature. Most research efforts in this area have been on trying to improve its classification efficiency [1,2,21]. Various strategies have been proposed to avoid an exhaustive search for all training instances and to achieve accurate classification.

In the presence of class imbalance kNN classification also faces challenges to correctly detect the positive instances. For a query instance, if its neighborhood is overwhelmed by negative instances, positive instances are still likely to be ignored in the decision process. Our main idea to mitigate the decision errors is to introduce a training stage to generalize the positive instances from a point to a gaussian ball in the instance space. Rather than generalizing every positive instances which may introduce false positives, we propose an algorithm to identify the exemplar positive instances called *pivot positive instances* (c.f. Section 3), and use them to reliably derive the positive class boundary.

Experiments on 12 real-world imbalanced datasets show that our classifier, k Exemplar-based Nearest Neighbor (kENN), is effective and significantly improves the performance of kNN for imbalanced learning. kENN also outperforms the current re-sampling and cost-sensitive learning strategies, namely SMOTE [5] and MetaCost [7], for imbalanced classification.

1.1 Related Work

kNN has been advocated for learning with minority instances [10,19,3] because of its high specificity bias of keeping all minority instances. In [10], the problem of small disjuncts (a small cluster of training instances) was first studied and the maximum specificity bias was shown to reduce errors for small disjuncts. In [19] kNN was used to improve learning from small disjuncts encountered in the C4.5 decision tree system [17]. In [3] kNN was employed to learn from small disjuncts directly, however learning results were not provided to demonstrate its performance. In contrast to these previous work, we propose to generalize exemplar positive instances to form concepts for the minority class, and then apply kNN directly for imbalanced classification.

With the assumption that learning algorithms perform best when classes are evenly distributed, re-sampling training data for even class distribution has been proposed to tackle imbalanced learning. Kubat and Matwin [12] tried to

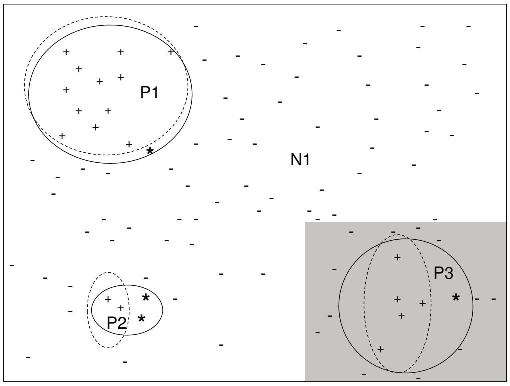


Fig. 1. An artificial imbalance classification problem

under-sample the majority class, while Ling and Li [13] combined over-sampling of the minority class with under-sampling of the majority class. Especially Chawla and Bowyer [5] proposed Synthetic Minority Over-sampling TEchnique (SMOTE) to over-sample the minority class by creating synthetic samples. It was shown that SMOTE over-sampling of the minority class in combination with under-sampling the majority class often could achieve effective imbalanced learning.

Another popular strategy tackling the imbalanced distribution problem is cost-sensitive learning [8]. Domingos [7] proposed a re-costing method called MetaCost, which can be applied to general classifiers. The approach made error-based classifiers cost-sensitive. His experimental results showed that MetaCost reduced costs compared to cost-blind classifier using C4.5Rules as baseline.

Our experiments (c.f. Section 5) show that SMOTE in combination with under-sampling of majority class as well as MetaCost significantly improves the performance of C4.5 for imbalanced learning. However these strategies somehow do not statistically significantly improve the performance of kNN for class imbalance. This may be partly explained by that kNN makes classification decision by examining the local neighborhood of query instances where the global re-sampling and cost-adjustment strategies may not have pronounced effect.

2 Main Ideas

Fig. 1 shows an artificial two-class imbalance problem, where positive instances are denoted as "+" and negative instances are denoted as "-". True class boundaries are represented as solid lines while the decision boundaries by some classification model are represented as dashed lines. Four query instances that indeed belong to the positive class are represented as stars (*). Three subconcepts associated with the positive class are the three regions formed by the solid lines, denoted as P1, P2 and P3 respectively. Subconcept P1 covers a large portion of instances in the positive instance space whereas P2 and P3 correspond to small

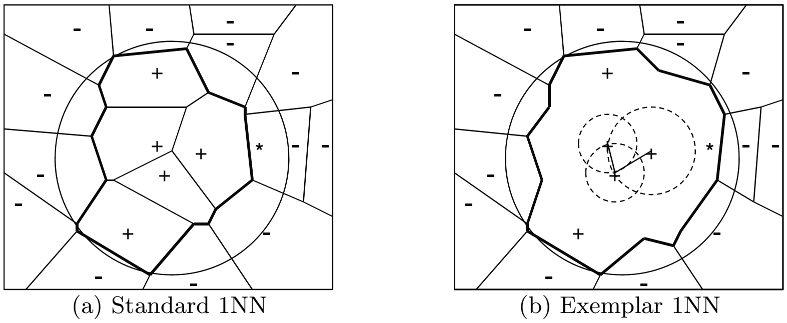


Fig. 2. The Voronoi diagram for the subspace of subconcept P3 of Fig. 1

disjuncts of positive instances. Note that the lack of data for the subconcepts P2 and P3 cause the classification model to learn inappropriate decision boundaries for P2 and P3. As a result, two query instances (denoted by *) that are indeed positive defined by P2 fall outside the positive decision boundary of the classifier and similarly for another query instance defined as positive by $P3$.

Given the problem in Fig. 1, we illustrate the challenge faced by a standard kNN classifier using the subspace of instances at the lower right corner. Figure 2(a) shows the Voronoi diagram for subconcept P3 in the subspace, where the positive class boundary decided by standard 1NN is represented as the polygon in bold line. The 1NN induction strategy where the class of an instance is decided by the class of its nearest neighbor results in a class boundary much smaller than the true class boundary (circle). As a result the query instance (denoted by *), which indeed is a positive instance inside the true positive boundary, is predicted as negative by standard 1NN. Obviously to achieve more accurate prediction, the decision boundary for the positive class should be expanded so that it is closer to the true class boundary.

A naive approach to expanding the decision boundary for the positive class is to generalize every positive instance in the training instance space from a point to a Gaussian ball. However this aggressive approach to expanding the positive boundary can most definitely introduce false positives. We need a strategy to selectively expand some positive points in the training instance space so that the decision boundary closely approximates the real class boundary while not introducing too many false positives.

Our main idea of expanding the decision boundary for the positive class while minimizing false positives is based on exemplar positive instances. Exemplar positive instances should be the positive instances that can be generalized to reliably classify more positive instances in independent tests. Intuitively these instances should include the strong positive instances at or close to the center of a disjunct of positive instances in the training instance space. Weak positive instances close to the class boundaries should be excluded.

Fig. 2(b) shows the Voronoi diagram after the three positive instances at the center of the disjunct of positive instances have been used to expand the boundary for the positive class. Obviously the decision boundary after adjustment is much closer to the real class boundary. As a result, the query instance (represented by *) is now enclosed by the boundary decided by the classifier and is correctly predicted as positive.

3 Pivot Positive Instances

Ideally exemplar positive instances can be reliably generalized to form the subconcept for a disjunct of positive instances with low false positive errors in the space of training instances. We call these exemplar instances *pivot positive instances* (PPIs) and define them using their neighborhood.

Definition 1. *The Gaussian ball $B(x, r)$ centered at an instance x in the training instance space $\mathbb{R}^n$ (n is the number of features defining the space) is the set of instances within distance r of x: $\{y \in \mathbb{R}^n |\ distance(x, y) \leq r\}$.*

Each Gaussian ball defines a positive subconcept and only those positive instances that can form sufficiently accurate positive subconcepts are pivot positive instances, as defined below.

Definition 2. *Given a training instance space $\mathbb{R}^n$, and a positive instance $x \in \mathbb{R}^n$, let the distance between x and its nearest positive neighbor be e. For a false positive error rate (FP rate) threshold δ, x is a pivot positive instance (PPI) if the subconcept for Gaussian ball $B(x, e)$ has an FP rate $\leq \delta$.*

For simplicity the FP rate for the Gaussian ball centered at a positive instance is called the FP rate for the positive instance. To explain the concept of PPI, let us for now assume that the false positive rate for a positive instance is its observed false positive ratio in the training instance space.

Example 1. Consider the positive instances in the subspace highlighted in Fig. 1 Given a false positive rate threshold of 30%, the three positive instances at the center have zero false positives in its Gaussian ball (shown in enlarged form in Fig. 2(b)) and therefore are PPIs. The other two positive instances however are not PPIs, as they have observed false positive ratio of respectively 50% (2 out of 4) and 33.3% (1 out of 3).

The observed false positive ratio for a positive instance in the training instance space is not accurate description of its performance in the presence of independently chosen test instances. We estimate the false positive rate by re-adjusting the observed false positive ratio using pessimistic estimate. A similar approach has been used to estimate the error rate for decision tree nodes in C4.5 [17,22]. Assume that the number of false positives in a Gaussian ball of N instances follows the binomial distribution $B(N, p)$, where p is the real probability of false positives in the Gaussian ball. For a given confidence level c where its corresponding z can be computed, p can be estimated from N and the observed false positive ratio f as follows [22]:

$$\frac{f + z^2/2N + z\sqrt{f(1-f)/N + z^2/4N^2}}{1 + z^2/N} \tag{1}$$

Algorithm 1. Compute the set of positive pivot points

```
Input:   a) Training set $T$ ($|T|$ is number of instances in $T$); b) confidence level $c$.
Output:  The set of pivot positive instances $P$ (with radius $r$ for each Gaussian ball)
 1: $\delta \leftarrow$ FP rate threshold by Equation (1) from $c$, $|T|$, and prior negative frequency
 2: $P \leftarrow \phi$
 3: for each positive instance $x \in T$ do
 4:    $G \leftarrow$ neighbors of $x$ in increasing order of distance to $x$
 5:    for $k = 1$ to $|G|$ do
 6:       if $G[k]$ is a positive instance then
 7:          break          {;; $G[k]$ is the nearest positive neighbor of $x$}
 8:       end if
 9:    end for
10:    $r \leftarrow$ distance($x$, $G[k]$)
11:    $f \leftarrow \frac{k-1}{k+1}$     {;; Gaussian ball B(x, r) has $k+1$ instances and $(k+1-2)$ FPs}
12:    $p \leftarrow$ the FP rate by Equation (1) from $c$, $k$ and $f$
13:    if $p \leq \delta$ then
14:       $P \leftarrow P \cup \{x\}$       {;; $x$ is a pivot positive instance, and $P$ is the output}
15:    end if
16: end for
```

The Gaussian ball for a positive instance always has two positive instances— the reference positive instance and its nearest positive neighbor. The confidence level is a parameter tuning the performance of PPIs. A high confidence level means the estimated false positive error rate is close to the observed false negative ratio, and thus few false positives are tolerated in identifying PPIs. On very imbalanced data we need to tolerate a large number of false positives to aggressively identify PPIs to achieve high sensitivity for the positives. Our experiments (Section 5.3) confirm this hypothesis. The default confidence level is set to 10%.

We set the FP rate threshold for identifying PPIs based on the imbalance level of training data. The threshold for PPIs are dynamically determined by the prior negative class frequency. If the false positive rate for a positive instance estimated using Equation (1) is not greater than the threshold estimated from the prior negative class frequency, the positive instance is a PPI. Under this setting, a relatively larger number of FP errors are allowed in Gaussian balls for imbalanced data while less errors are allowed for balanced data. Especially on very balanced data the PPI mechanism will be turned off and kENN reverts to standard kNN. For example on a balanced dataset of 50 positive instances and 50 negative instances, at a confidence level of 10%, the FP rate threshold for PPIs is 56.8% (estimated from the 50% negative class frequency using Equation (1)). A Gaussian ball without any observed FP errors (and containing 2 positive instances only) has an estimated FP rate of 68.4%[2]. As a result no PPIs are identified at the training stage and standard kNN classification will be applied.

[2] Following standard statistics, when there are not any observed errors, for N instances at confidence level c the estimated error rate is $1 - \sqrt[N]{c}$.

4 k Exemplar-Based Nearest Neighbor Classification

We now describe the algorithm to identify pivot positive instances at the training stage, and how to make use of the pivot positive instances for classification for nearest neighbor classification.

The complete process of computing pivot positive instances from a given set of training instances is illustrated in Algorithm 1. Input to the algorithm are the training instances and a confidence level c. Output of the algorithm are the pivot positive instances with their radius distance for generalization to Gaussian balls. In the algorithm first FP rate threshold δ is computed using Equation (1) from confidence level c, number of training instances $|T|$ and the prior negative class frequency (line 1). The neighbors of each positive instance x are sorted in increasing order of their distance to x (line 4). The loop of lines 3 to 16 computes the PPIs and accumulating them in P to be output (line 14). Inside the loop, the FP rate p for each positive instance x is computed using Equation (1) from the observed FP ratio f (line 12) and if $p \leq \delta$, x is identified as a PPI and kept in P (line 14). The main computation of Algorithm 1 lies in the process computing up to $n-1$ nearest neighbors for each positive instance x and sorting according to their distance to x, where n is the size of the training set (line 4). Algorithm 1 thus has a complexity of $O(p{*}n \log n)$, where p and n are respectively the number of positive and all instances in the training set. Note that $p << n$ for imbalanced datasets, and so the algorithm has reasonable time efficiency.

At the classification stage, to implement the concept of Gaussian balls for pivot positive instances, the distance of a query instance to its k nearest neighbors is adjusted for all PPIs. Specifically for a query instance t and a training instances x , the adjusted distance between t and x is defined as:

$$adjusted\_distance(t,x) = \begin{cases} distance(t,x) - x.radius & \text{if } x \text{ is a PPI} \\ distance(t,x) & \text{otherwise} \end{cases} \quad (2)$$

where $distance(t,x)$ is the distance between t and x using some metric of standard kNN. With the above equation, the distance between a query instance and a PPI in the training instance space is reduced by the radius of the PPI. As a result the adjusted distance is conceptually equivalent to the distance of the query instance to the edge of the Gaussian ball centered at the PPI. The adjusted distance function as defined in Equation (2) can be used in kNN classification in the presence of class imbalance, and we call the classifier k Exemplar-based Nearest Neighbor (kENN).

5 Experiments

We conducted experiments to evaluate the performance of kENN. kENN was compared against kNN and the naive approach of generalizing positive subconcepts for kNN (Section 2). kENN was also compared against two popular imbalanced learning strategies SMOTE re-sampling and MetaCost cost-sensitive

Table 1. The experimental datasets, ordered in decreasing level of imbalance

Dataset	size	#attr (num, symb)	classes (pos, neg)	minority (%)
Oil	937	47(47, 0)	(true, false)	4.38%
Hypo-thyroid	3163	25 (7, 18)	(true, false)	4.77%
PC1	1109	21 (21,0)	(true, false)	6.94%
Glass	214	9 (9,0)	(3, other)	7.94%
Satimage	6435	36 (36,0)	(4, other)	9.73%
CM1	498	21 (21,0)	(true, false)	9.84%
New-thyroid	215	5 (5,0)	(3, other)	13.95%
KC1	2109	21 (21,0)	(true, false)	15.46%
SPECT_F	267	44 (44,0)	(0, 1)	20.60%
Hepatitis	155	19 (6,13)	(1, 2)	20.65%
Vehicle	846	18 (18,0)	(van, other)	23.52%
German	1000	20 (7,13)	(2, 1)	30.00%

learning, using kNN (IBk in WEKA) and C4.5 (J48 in WEKA) as the base classifiers. All classifiers were developed based on the WEKA data mining toolkit [22], and are available at `http://www.cs.rmit.edu.au/~zhang/ENN`. For both kNN and kENN k was set to 3 by default, and the confidence level of kENN was set to 10%. To increase the sensitivity of C4.5 to the minority class, C4.5 was set with the -M1 option that minimum one instance was allowed for a leaf node and without pruning. SMOTE oversampling combined with undersampling was applied to 3NN and C4.5, denoted as 3NNSmt+ and C4.5Smt+ respectively. SpreadSubsample was used to undersample the majority class for uniform distribution (M=1.0), and then SMOTE was applied to generate additional 3 times more instances for the minority class. MetaCost was used for cost-sensitive learning with 3NN and C4.5 (denoted as 3NNMeta and C4.5Meta) and the cost of each class was set to the inverse of class ratio.

Table 1 summarizes the 12 real-world imbalanced datasets from various domains used in our experiments, from highly imbalanced (the minority 4.35%) to moderately imbalanced (the minority 30.00%). The Oil dataset was provided by Robert Holte [11], and the task is to detect the oil spill (4.3%) from satellite images. The CM1, KC1 and PC1 datasets were obtained from the NASA IV&V Facility Metrics Data Program (MDP) repository (`http://mdp.ivv.nasa.gov/index.html`). The task is to predict software defects (around 10% on average) in software modules. The remaining datasets were compiled from the UCI Machine Learning Repository (`http://archive.ics.uci.edu/ml`). In addition to the natural 2-class domains, like thyroid diseases diagnoses and Hepatitis, we also constructed four imbalanced datasets by choosing one class as the positive and the remaining classes combined as the negative.

The Receiver Operating Characteristic (ROC) curve [18] is becoming widely used to evaluate imbalanced classification. Given a confusion matrix of four types of decisions True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN), ROC curves depict tradeoffs between $TP\ rate = \frac{TP}{TP+FN}$ and $FP\ rate = \frac{FP}{FP+TN}$. Good classifiers can achieve a high TP rate at a low

Table 2. The AUC for *k*ENN, in comparison with other systems. The best result for each dataset is in bold. AUCs with difference <0.005 are considered equivalent.

Dataset	3ENN	Naive	3NN	3NNSmt+	3NNMeta	C4.5	C4.5Smt+	C4.5Meta
Oil	**0.811**	0.788	0.796	0.797	0.772	0.685	0.771	0.764
Hypo-thyroid	0.846	0.831	0.849	0.901	0.846	0.924	**0.948**	0.937
PC1	**0.806**	0.786	0.756	0.755	0.796	0.789	0.728	0.76
Glass	0.749	0.623	0.645	0.707	0.659	0.696	0.69	**0.754**
Satimage	**0.925**	0.839	0.918	0.902	**0.928**	0.767	0.796	0.765
CM1	**0.681**	0.606	0.637	0.666	0.625	0.607	0.666	0.668
New-thyroid	**0.99**	0.945	0.939	0.972	0.962	0.927	0.935	0.931
KC1	**0.794**	0.732	0.759	0.756	0.779	0.64	0.709	0.695
SPECT_F	**0.767**	0.728	0.72	0.725	0.735	0.626	0.724	0.643
Hepatitis	**0.783**	0.71	0.758	0.772	0.744	0.753	0.713	0.745
Vehicle	0.952	0.945	**0.969**	0.942	0.956	0.921	0.926	0.929
German	**0.714**	0.677	0.69	0.686	0.705	0.608	0.649	0.606
Average	**0.818**	0.768	0.786	0.798	0.792	0.745	0.771	0.766

FP rate. Area Under the ROC Curve (AUC) measures the overall classification performance [4], and a perfect classifier has an AUC of 1.0. All results reported next were obtained from 10-fold cross validation and two-tailed paired t-tests at 95% confidence level were used to test statistical significance.

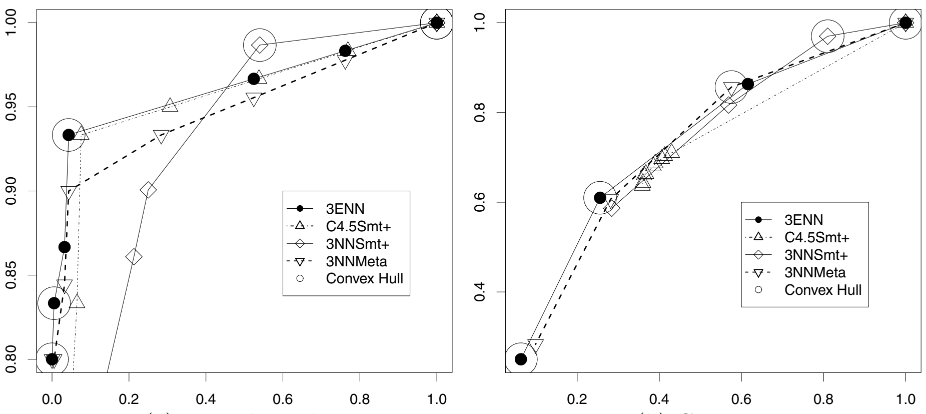
The ROC convex hull method provides visual performance analysis of classification algorithms at different levels of sensitivity [15,16]. In the ROC space, each point of the ROC curve for a classification algorithm corresponds to a classifier. If a point falls on the convex hull of all ROC curves the corresponding classifier is potentially an optimal classifier; otherwise the classifier is not optimal. Given a classification algorithm, the higher fraction of its ROC curve points lie on the convex hull the more chance the algorithm produce optimal classifiers.

For all results reported next, data points for the ROC curves were generated using the ThresholdCurve module of WEKA, which correspond to the number of TPs and FPs that result from setting various thresholds on the probability of the positive class. The AUC value for ROC curves were obtained using the Mann Whitney statistic in WEKA. The convex hull for ROC curves were computed using the ROCCH package[3].

5.1 Performance Evaluation Using AUC

Table 2 shows the AUC results for all models. It can be seen that 3ENN is a very competitive model. Compared with the remaining models, 3ENN has the highest average AUC of 0.818 and wins on 9 datasets. In comparison the average AUC for the Naive method is just 0.768. 3ENN significantly outperforms all of 3NN (p = 0.005), 3NNSmt+ (p = 0.029), 3NNMeta (p = 0.008), C4.5Smt+ (p = 0.014) and C4.5Meta (p = 0.021). This result confirms that our exemplar-based positive concept generalization strategy is very effective for improving the

[3] Available at `http://home.comcast.net/~tom.fawcett/public_html/ROCCH/`



(a) New-thyroid (b) German

Fig. 3. ROC curves with convex hull on two datasets. The x-axis is the FP rate and the y-axis is the TP rate. Points on the convex hull are highlighted with a large circle.

performance of kNN for imbalanced classification, and furthermore the strategy is more effective than re-sampling and cost-sensitive learning strategies.

It should be noted that C4.5Smt+ and C4.5Meta both demonstrate improvement over C4.5. This shows that re-sampling and cost-sensitive learning strategies are effective for improving the performance of C4.5 for imbalanced classification, which is consistent with previous findings [5,20]. On the other hand however, 3NNSmt+ and 3NNMeta do not show significant improvement over 3NN. That these strategies are less effective on kNN for class imbalance may be attributed to that kNN adopts a maximal specificity induction strategy. For example, the re-sampling strategy ensures overall class balance however it does not necessarily mean that for the neighborhood of individual query instances the minority class is well represented. Not forming concepts from overall even sample distribution after re-sampling, kNN may still miss some positive query instances due to the under-representation of the positive class in their neighborhood.

5.2 The ROC Convex Hull Analysis

Table 2 has shown that C4.5Smt+ outperforms C4.5Meta, and so for readability we only compare the ROC curves of 3ENN against that of 3NNSmt+, 3NNMeta and C4.5Smt+. Fig. 3 shows the ROC curves of the four models on the New-thyroid and German datasets. From Table 2, 3ENN and 3NNSmt+ have the best AUC results of 0.99 and 0.972 on New-thyroid, which has a relatively high level of imbalance of 13.95%. But as shown in Fig. 3(a), the ROC curves of the four models show very different trends. Notably more points of 3ENN lie on the convex hull at low FP rates (<10%). Conversely more points of 3NNSmt+ lie on the convex hull at high FP rates (>50%). It is desirable in many applications to achieve accurate prediction at low false positive rate and so 3ENN is obviously a good choice for this purpose. German has a moderate imbalance level of 30%.

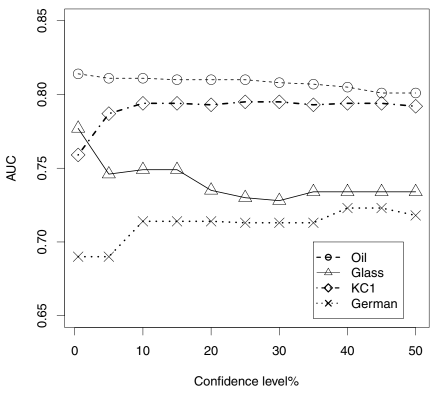


Fig. 4. The AUC of 3ENN with varying confidence level

ROC curves of the four models demonstrate similar trends on German, as shown in Fig. 3(b). Still at low FP rates, more points from 3ENN lie on the ROC convex hull, which again shows that 3ENN is a strong model.

5.3 The Impact of Confidence Level on *k*ENN

As discussed in Section 3 confidence level affects the decision in *k*ENN of whether to generalize a positive instance to a Gaussian ball. We applied 3ENN to two highly imbalanced datasets and two moderately imbalanced datasets with confidence level from 1% to 50%. The AUC results are shown in Fig. 4. For the two datasets with high imbalance (Oil 4.38% and Glass 7.94%) AUC is negatively correlated with confidence level. For example on Oil when the confidence level increases from 1% to 50% the AUC decreases from 0.813 to 0.801. However for the two datasets with moderate imbalance (KC1 15.46% and German 30.00%) AUC is inversely correlated with confidence level. On German when confidence level increases from 1% to 50% AUC increases from 0.69 to 0.718. The opposite behavior of AUC in relation to confidence level may be explained by that on highly imbalanced data, to predict more positive instances, it is desirable to tolerate more false positives in forming Gaussian balls, which is achieved by setting a low confidence level. Such an aggressive strategy increases the sensitivity of *k*ENN to positive instances. On less imbalanced datasets where there are relatively sufficient positive instances, a high confidence level is desired to ensure a low level of false positives in positive Gaussian balls.

6 Conclusions

With *k*NN classification, the class of a query instance is decided by the majority class of its k nearest neighbors. In the presence of class imbalance, a query instance is often classified as belonging to the majority class and as a result many positive (minority class) instances are misclassified. In this paper, we have proposed a training stage where exemplar positive training instances are identified and generalized into Gaussian balls as concepts for the minority class. When classifying a query instance using its k nearest neighbors, the positive concepts formulated at the training stage ensure that classification is more sensitive to the minority class. Extensive experiments have shown that our strategy significantly improves the performance of kNN and also outperforms popular re-sampling and cost-sensitive learning strategies for imbalanced learning.

References

1. Aha, D.W. (ed.): Lazy learning. Kluwer Academic Publishers, Dordrecht (1997)
2. Aha, D.W., et al.: Instance-based learning algorithms. Machine Learning 6 (1991)
3. Bosch, A., et al.: When small disjuncts abound, try lazy learning: A case study. In: BDCML (1997)
4. Bradley, A.P.: The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recognition 30 (1997)
5. Chawla, N.V., et al.: SMOTE: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research 16 (2002)
6. Cover, T., Hart, P.: Nearest neighbor pattern classification. Institute of Electrical and Electronics Engineers Transactions on Information Theory 13 (1967)
7. Domingos, P.: Metacost: A general method for making classifiers cost-sensitive. In: KDD 1999 (1999)
8. Elkan, C.: The foundations of cost-sensitive learning. In: IJCAI (2001)
9. Fawcett, T., Provost, F.J.: Adaptive fraud detection. Data Mining and Knowledge Discovery 1(3) (1997)
10. Holte, R.C., et al.: Concept learning and the problem of small disjuncts. In: IJCAI 1989 (1989)
11. Kubat, M., et al.: Machine learning for the detection of oil spills in satellite radar images. Machine Learning 30(2-3) (1998)
12. Kubat, M., Matwin, S.: Addressing the curse of imbalanced training sets: One-sided selection. In: ICML 1997 (1997)
13. Ling, C., et al.: Data mining for direct marketing: Problems and solutions. In: KDD 1998 (1998)
14. Menzies, T., et al.: Data mining static code attributes to learn defect predictors. IEEE Transactions on Software Engineering 33 (2007)
15. Provost, F., et al.: The case against accuracy estimation for comparing induction algorithms. In: ICML 1998 (1998)
16. Provost, F.J., Fawcett, T.: Robust classification for imprecise environments. Machine Learning 42(3) (2001)
17. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco (1993)
18. Swets, J.: Measuring the accuracy of diagnostic systems. Science 240(4857) (1988)
19. Ting, K.: The problem of small disjuncts: its remedy in decision trees. In: Canadian Conference on Artificial Intelligence (1994)
20. Weiss, G.M.: Mining with rarity: a unifying framework. SIGKDD Explorations 6(1) (2004)
21. Wilson, D.R., Martinez, T.R.: Reduction techniques for instance-based learning algorithms. Machine Learning (2000)
22. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann, San Francisco (2005)