Boosting SVM Classifiers with Logistic Regression

Yuan-chin Ivan Chang YCCHANG@SINICA.EDU.TW
Institute of Statistical Science
Academia Sinica
Taipei, Taiwan 11529, ROC

Editor:

Abstract

The support vector machine classifier is a linear maximum margin classifier. It performs very well in many classification applications. Although, it could be extended to nonlinear cases by exploiting the idea of kernel, it might still suffer from the heterogeneity in the training examples. Since there are very few theories in the literature to guide us on how to choose kernel functions, the selection of kernel is usually based on a try-and-error manner. When the training set are imbalanced, the data set might not be linear separable in the feature space defined by the chosen kernel. In this paper, we propose a hybrid method by integrating "small" support vector machine classifiers by logistic regression models. By appropriately partitioning the training set, this ensemble classifier can improve the performance of the SVM classifier trained with whole training examples at a time. With this method, we can not only avoid the difficulty of the heterogeneity, but also have probability outputs for all examples. Moreover, it is less ambiguous than the classifiers combined with voting schemes. From our simulation studies and some empirical results, we find that these kinds of hybrid SVM classifiers are robust in the following sense: (1) It improves the performance (prediction accuracy) of the SVM classifier trained with whole training set, when there are some kind of heterogeneity in the training examples; and (2) it is at least as good as the original SVM classifier, when there is actually no heterogeneity presented in the training examples. We also apply this hybrid method to multi-class problems by replacing binary logistic regression models with polychotomous logistic regression models. Moreover, the polychotomous regression model can be constructed from individual binary logistic regression models.

Keywords: Support vector machine, imbalanced training sets, logistic regression, polychotomous logistic regression.

1. Introduction

The support vector machine classifier is a linear maximum margin classifier. It can be extended to nonlinear cases by exploiting the idea of kernel, and transforming the data from its original input space to the feature space without explicitly specifying the transformation (see Vapnik (1995)). Then, to construct a linear maximum margin classifier in the feature space. Typically, it is nonlinear in the input space. But it is still linear in a higher dimensional feature space implicitly defined by the chosen kernel function.

The SVM classifiers performs very well in many classification applications such as pattern recognition, text classification, bio-informatics, etc. A good geometric interpretation and the efficiency of generalization are its advantages. But if the given training data are

heterogeneous or imbalanced, it might have less satisfying performance. It is worse, especially, when the given training data have some implicit geometric structure and is not linear separable in the feature space defined by the chosen kernel function. Usually, this problem can be solved by choosing another appropriate kernel function. But in the most cases, the kernel functions was chosen empirically or by a try-and-error manner. So far, we have found no satisfying theoretical results on kernel selection in the literature, yet.

For a classification problem with imbalanced training set, if the training set can be appropriately divided into some smaller sub-sets, then each sub-training set is more likely to be linear separable in the (implicitly chosen) feature space. Then, we can construct many small "localized" SVM classifiers for each sub-training set, which will not be affected by the heterogeneity of the whole training set. Thus, they will have better local prediction accuracies than the SVM classifier trained with whole training examples. If we can effectively combine these classifiers into a global one, then it is very likely that it will outperform the classifier trained with whole training examples. So, the remaining problem is how to combine these local classifiers into one global classifier, and still to gain their local advantages, simultaneously?

In this paper, we propose a hybrid classification method by integrating the "local SVM classifiers" with logistic regression techniques; i.e. by a divide-and-conquer method. The computational bottle neck of the hybrid method proposed here is still in the SVM. The logistic regression will not introduce any new computational difficulties. By Begg and Gray (1984), we can build the polychotomous logistic regression model from some individualized binary logistic regression models, which will make the computation of multi-class problem more efficient. Therefore, it can be useful when the size of training set is large. In Zhu and Hastie, they propose a "kernelized logistic regression" method, which can also provide probability outputs for examples. But the computational complexity of their method is $O(n^3)$ where n denotes the size of training examples. Hence, it is very difficult to apply to the problems with large training sets.

In machine learning, the voting (or weighting) is one of the popular methods to put many classifiers together becoming one classifier (see Tumer and Ghosh (1996), Hastie and Tibshirani (1998), Bauer and Kohavi (1999) and Optiz and Maclin (1999)). There are also many studies about how to train a classifier with imbalanced training examples and on different ensemble methods. For example, see Provost and Fawcett (1998), Japkowicz (2000a,b), Weiss and Provost (2001) and Kubat et al. (1997). But there are always some ambiguities on how to decide the weights for the classifiers with different classification accuracy. Using logistic regression models to integrate these local classifiers allows us to take advantages of the statistical modelling theory to find the optimal weights for each local classifiers. The results can be easily extended to multi-class classification problems and the multiple labelling problems by using polychotomous logistic regression models instead.

We have applied this method to both simulated data as well as some bench mark data from UCI repository of machine learning (Blake and Merz (1998)). These empirical results show that the proposed hybrid method is very promising in dealing with classification problem with non-homogeneous training data.

In the following of this paper, we will first explain the effects of the non-homogeneity in the training sets using simulated data sets. We state the hybrid procedures for both binary and multi-class classification problems in Section 3. Section 4 contains some sim-

ulation/empirical results. In Section 5, we will discuss the possible future works. A brief introduction of (polychotomous) logistic regression will also be given in Appendix.

2. Heterogeneity

It is well known that the performance of the case-based learning algorithm depends heavily on the given training examples. It hard to have a good prediction accuracy, if there is some heterogeneity in the training set. Unfortunately, due to modern data collection techniques, it is very common to have large and imbalanced training sets in many classification applications. For example, in data mining applications of biological sciences, text classification, etc, the size of the negative examples often heavily outnumbers the size of the positive examples (Kubat et al. (1997)).

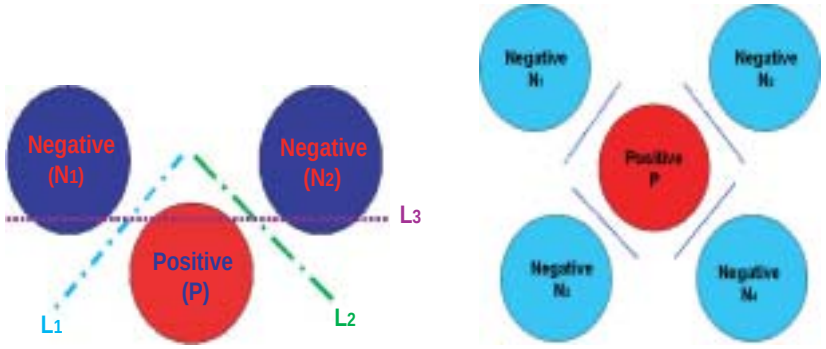
In many practical applications, we might just have very little knowledge about what the "positive" cases should be; or, there are just few positive cases in hand. (That's why we need to train a classifier. Otherwise, we already know how to distinguish them from others, and no classifier is needed.) But it might be convenient for us to point out what kind of cases should not be "classified" as positive cases. Thus, it is convenient for us to include a lot of "negative" examples into training sets.

Usually, people will collect their negative examples from the "complement of the positive examples". Since we have very little idea about what the range of positive examples should be. It is very likely that the collected negative examples are actually from many different sub-populations (see also Weiss and Provost (2001)). They are called negative by different aspects from a view point of the given positive examples, which we are really interested in. In other words, they might be called "negative" for different reasons. Geometrically, these negative sub-sets might be located in different directions/distances from the location of positive examples. When it happens, the "geometry-based" classification algorithms, such as the SVM, will suffer from this kind of the hidden "geometric-heterogeneity" (such as trying different kernel functions). Because the SVM depends heavily on the implicit geometric property of the data in the feature space, and we are not able to foresee the possible geometric structure in the feature space, it is hard to choose a suitable kernel function for each classification problem. We will illustrate effects of the "geometric-heterogeneity" by the pictures below. They are very common in the practical problems. Especially, when the sizes of the positive and the negative examples in the training set are imbalanced.

2.1 Geometric Heterogeneity

Now, assume a kernel is chosen. Figure 1 and 2 are pictures of the of the training data in the feature space projected by a transformation implicitly defined by the chosen kernel. Figure 1 shows that the negative and positive parts of the data set might not be separable by a simple linear hyper-plain. Nevertheless, the positive set can be easily separated from each negative sub-sets by a different linear hyper-plain. Figure 2 shows that some the negative sets are redundant. (In the rest of this section, we will assume that the kernel function has been chosen by users, and all the pictures of data are in the feature space.)

Let C_1 and C_2 be two "local" classifiers constructed by SVM using P against N_1, and P against N_2 in Figures 1(a), respectively. Let L_1 and L_2 be two corresponding separating hyper-plains for these two sub-classifiers C_1 and C_2. Suppose C_3 is a SVM classifier trained

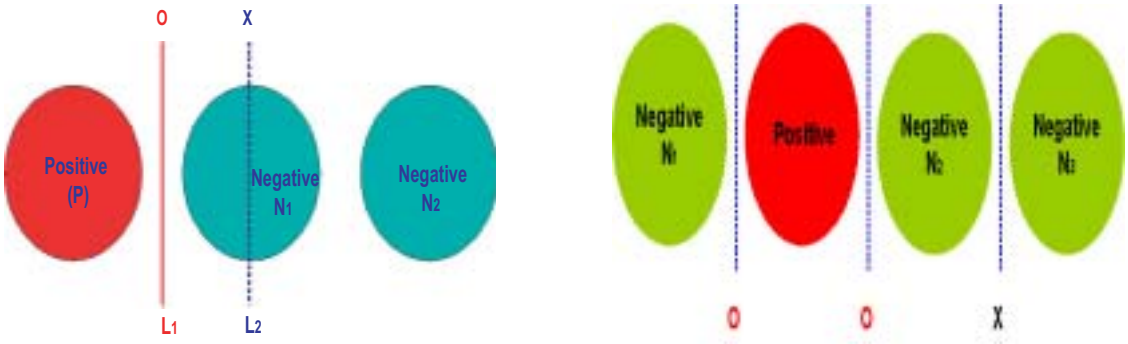


Mickey Mouse (a) Olympic Circles (b)

Figure 1: Effect of Heterogeneity I

with whole training examples (i.e. to use N_1 and N_2 together versus P for training), and L_3 is the separating hyper-plain of C_3. It is clear to see from Figure 1 (a) that it is nonlinear separable in the feature space defined by the chosen kernel, and Classifier C_3 will have small generalization errors. On the other hand, the local classifier C_1 and C_2 might be much better than C_3, locally. But, each of them alone is certainly not a good global classifier. It suggests that it is possible to construct a better classifier than C_3 by combining C_1 and C_2, effectively. Voting might be a convenient choice, but it usually will introduce a lot of ambiguity and noise (see Bauer and Kohavi (1999)). A better integration method is needed.

Figure 1(b) is a more extreme example. When the negative examples in the feature space sit on all directions as shown in Figure 1 (b), it is definite not linear separable. (Certainly, we can apply another kernel function to the feature space. But as mentioned before, there is no clear guideline of kernel selection, yet.)



Pipe (a) Bi-Direction (b)

Figure 2: Effect of Heterogeneity II

Figure 2 (a) shows that the separating plain L_1 is good enough to separate the positive examples P from N_1 and N_2. The hyper-plain L_2, constructed with P and N_2, is redundant;

that is the negative examples N_2 provides very little information for constructing the final classifier. In this case, it is very difficult to combine these two into one classifier with high specificity and high sensitivity by voting (or weighting) scheme. Figure 2 (b) shows us a more complicated situation than Figure 2(a).

The common of data sets in Figure 1 and 2 is that the negative examples in the training sets can be further divided into several sub-groups (N_i's), and from the view point of the positive examples(P), these sub-groups (N_i's) actually sit on different directions (aspects) (1) or on different "distances" away from the positive group (2). Thus, it is possible to improve the performance of the original SVM classifier by combining these local classifiers, effectively.

3. Hybrid Method

Suppose the size of negative examples is much larger than the size of the positive examples, and these negative examples can be divided into some smaller subsets by a clustering algorithm. It follows from the discussion in the previous section, we can construct many local SVM classifiers using the whole positive examples against each negative subset obtained from the clustering algorithm. Then, we can integrate these local SVM classifiers with logistic regression models. The statistical model selection techniques can assist us to take the advantages of each local classifier such that the final global classifier could have a better performance than original SVM classifiers has. When the training examples are homogeneous, the performance of this hybrid method will still be compatible to that of the SVM classifier trained with whole data.

3.1 Hybrid Procedure

For a two-class classification problem with ℓ training examples, a (2-norm) soft margin support vector machine is to solve the following equation:

$$\begin{aligned} &\text{minimize}_{\xi,\mathbf{w},b} \langle \mathbf{w} \cdot \mathbf{w} \rangle + C \sum_{i=1}^{\ell} \xi^2 \\ &\text{subject to } y_i \left(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \right) \geq 1 - \xi_i, \; i = 1, \cdots, \ell, \\ &\qquad \xi_i \geq 0, \; i = 1, \cdots, \ell, \end{aligned} \tag{1}$$

where $y_i \in \{-1, 1\}$ denotes the labels of examples and x_i denotes the feature vector. Let $H = H(\mathbf{x}) = \langle \mathbf{w} \cdot \mathbf{x} \rangle + b = 0$ be the separating hyper-plane obtained from (1). Then the "signed-distance" from example x_i to the hyper-plane H is $d_{H,i} = y_i \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b$.

3.1.1 Binomial/Multinomial Case

Let P and N denote the sets of positive and the negative examples, respectively. Suppose we have an imbalanced training data that the size of negative examples outnumbers of the size of positive examples; i.e. the ratio of $||P||/||N||$ is relatively small. (Notation $||S||$ denotes the number of elements in a set S.) Assume further that we can divide the set N into k subsets, say $N_1, \cdots, N_k$. Then for each i, $i = 1, \cdots, k$, we can construct an SVM

classifier Δ_i by using $P \bigcup N_i$ as a training set. (Thus, there are k classifiers in total.) For $i = 1, \cdots, k$, let H_i be the separating hyper-plain of Classifier Δ_i in feature space, and $d_{s,i}$ be the "signed-distance" of the example s to the hyper-plain H_i. Thus, every example s in the data set can be represented by a k-dimension vector $d = (d_{s,1}, \cdots, d_{s,k})$.

Procedure I (Nominal Case):

Step 0. Divide the negative examples into k partitions by a clustering algorithm.

Step 1. Using C_0 as positive examples (baseline) and one of the $\{C_i, i = 1, \cdots, k\}$ at a time as negative examples to produce k SVM linear classifiers (Δ_1, $\cdots$, Δ_k).

Step 2. Compute the "signed distances" for each example to the separating hyper-planes of all classifiers obtained in **Step** 1. That is, every example s in the data set can be represented by $d_s = (d_{s,1}, \cdots, d_{s,k})$. (That is the vector d_s will be used as a new feature vector for example s.)

Step 3. Perform logistic regressions using d as vectors of covariates/regressors. (Model selection techniques can be used for selecting the best subset regressors.)

Step 4. Use cross-validation to find a probability threshold T for dividing the positive and negative examples.

Step 5. Repeat **Step** 2 for all testing data, then calculating the prediction probability p_0 for all testing examples using logistic regression model obtained from **Step** 3.

Step 6. Label the testing example as positive, when p_0 is larger than a threshold T, otherwise label it as negative.

Remark 1 *The number of partitions, k, may depend on the ratio of $||P||/||N||$. Our simulation results show that the performances are very stable for different k's. We will suggest to choose a k such that $||P||/||N_i||$ is close to 1, for each $i = 1, \cdots, k$. Here k is chosen by user, and it can be much smaller than the number of the original feature vector. Thus, Step 2 can be used as a "dimension-reduction" scheme.*

Suppose we have a multi-class classification problems with $k + 1$ classes. Let C_i, $i = 0, \cdots, k$ denote these $k+1$ sub-classes of training examples. If there is no ordinal relationship among these classes, and the sizes of classes are comparable, then we can apply *Procedure I* to the multi-class problem by replacing *Step 6* by *Step 6'* below:

Step 6' Assigning a test example will be assigned to Class i, if $p_i = max\{p_0, \cdots, p_k\}$. (Randomization might be needed if there is a tie in $\{p_0, \cdots, p_k\}$.)

3.1.2 Multi-class Problems With Ordinal Relationship

Following the notation above. Suppose that the sub-classes C_i's have a monotonic relationship; such that $C_0 \subset C_1 \subset \cdots \subset C_k$. In order to incorporate this information into

classification, a polychotomous logistic regression model will be used. Below is a procedure for the ordinal classes case.

Procedure II (Ordinal Case):

Step 1. Use SVM to produce k classifiers ($\Delta_1, \cdots, \Delta_k$) based "one-against-preceding-classes"; i.e. for each $j = 1, \cdots, k$, to train Classifier Δ_j using C_j as the positive set and $\bigcup_{i=0,\cdots,j-1} C_i$ as the negative set.

Step 2. Compute the distance from each subject to the separating hyperplane of each classifiers; i.e. for each subject there is a vector $d = (d_1, d_2, \cdots, d_k)$, where d_i denote the distance of the subject to the separating hyperplane of the i-th classifiers.

Step 3. Perform logistic regressions using d as vectors of covariates/regressors. (Again, the model selection methods can assist us to select the best subset of covariates/regressors.)

Step 4. Repeat *Step 2*, and calculating the prediction probability p_i, for $i = 0, \cdots, k$ for all testing examples using the final model obtained in *Step 3*.

Step 5. Assign a testing example to Class i, if $p_i = \max\{p_0, \cdots, p_k\}$. (Again, randomization might be needed if there is a tie in p's.)

4. Empirical Result

We apply the proposed method to some simulated data sets as well as some real data sets from UCI machine learning repository. The results are summarized in below. It shows that the hybrid method is very promising in both binary and multiple classification problems.

4.1 Simulation

In our simulation studies, we suppose that a kernel is chosen, and the simulated data are already in the feature space defined by the chosen kernel. Note that the SVM classifier is just a linear classifier in this feature space. So, we will compare our results with the SVM linear classifiers in this feature space. All the simulation results are done with a statistical software R. The SVM code of R is in its package $e1071$, which is implemented by David Meyer based on C/C++ code by Chih-Chung Chang and Chih-Jen Lin (see URL http://www.r-project.org).

Study 1
The data are generated from bivariate normal distributions with different mean (location) vectors and the same covariance matrix I_2 as below:

$$\mathbf{x}_{ij} \sim N(\mu_i, I_2), \tag{2}$$

where $\mu = \mathbf{v} * d$, for $\mathbf{v}^t = (0,0), (1,0), (0,1), (-1,0), (0,-1)$ and $d = 0.5, 1, 1.5, 2, 3, 4, 4.5$, and I_2 denote the 2×2 identity matrix. For each Class i, $i = 0, 1, \cdots, 5$, there are 400 examples ($j = 1, \cdots, 200$) generated from (2). For each class, we choose 200 for training

and the rest 200 will be used as testing examples. Hence, the size of training set and testing set are both 1000 examples, and the ratio $||P||/||N||$ is 1/4. Figure 3 are typical data maps in the feature space for different d's. The results are summarized in Table 1.

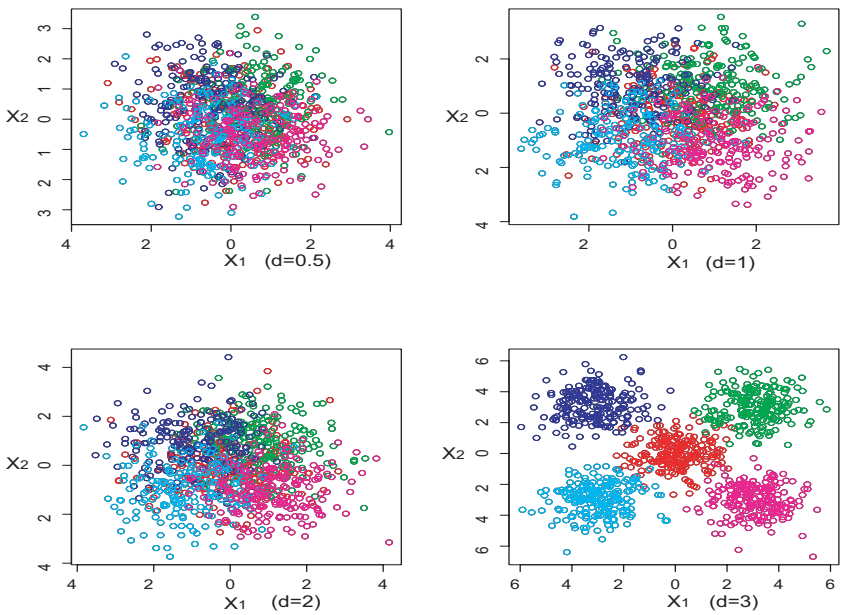


Figure 3: Cloverleaf ($d = 0.5, 1, 2, 3$)

Suppose we can foresee that data map of the training set in the feature space is as showed in Figure 3. Then, we can simply apply another radial based kernel function to the feature space to get a very good SVM classifier, and it is denoted by **SVM R** in Table 1. In Table 1, the **SVM L** is SVM linear classifier in this feature space, and **Hybrid** denotes the proposed method that integrates the local SVM classifiers with a logistic model.

For $d = 0.5$, **SVM L** might have the highest 80% accuracy, but it has 100% false negative rate, and then is certainly not a good classifier we want. Although, the Hybrid method has the lowest accuracy, but it has the most balanced false positive and false negative rates. From the false negative rates, it is clear that the SVM L is not a good classifier in this case. In the rest of Table 1, we only compare the results of **Hybrid** and **SVM R**, and we found that the results of **Hybrid** are very comparable to those of **SVM R**. Moreover, we found that **Hybrid** have more balanced false positive and negative rates, and the ROC curve below (Figure 4) will reflect this advantage of our hybrid method.

Now, suppose that we do not know the negative examples are actually from 4 different distributions. We first apply a K-mean clustering algorithm to it, then again apply the Procedure I to get a hybrid classifier. (In our simulations, we use the K-mean function in Matlab with its default parameters.) Figure 5 shows us an typical example ($d = 1.5$) of clusters obtained from the k-mean algorithm for different k's. Table 2 summarizes the results for different clusters numbers ($k = 3, 4, 5$) and different d's.

d	method	True Neg.	False Pos.	False Neg.	True Pos.	Accuracy
0.5	Hybrid	**659**	**141**	**150**	**50**	**70.9%**
	SVM R	765	35	195	5	77.0 %
	SVM L	**800**	0	**200**	0	**80.0%**
1	Hybrid	710	90	147	53	**76.3%**
	SVM R	779	21	185	15	79.4%
1.5	Hybrid	710	90	100	100	**81.0%**
	SVM R	769	31	147	26	79.5%
2	Hybrid	741	59	50	150	**89.1%**
	SVM R	750	50	68	132	88.2%
3	Hybrid	788	12	13	187	**97.5%**
	SVM R	794	6	18	182	97.6%
4	Hybrid	798	2	2	198	**99.6%**
	SVM R	799	1	2	198	99.7%
4.5	Hybrid	800	0	1	199	**99.9%**
	SVM R	800	0	2	198	99.8%

Table 1: Hybrid = SVM Linear + Logistic regression; SVM L = SVM classifier with linear kernel; SVM R = SVM classifier with radial base kernel function.

We found that the results in Table 2 is very similar to that in Table 1. In order to study the false positive and false negative rates, for $d = 4$, and for each $k = 3, 4, 5$, we repeat the simulations 10 times in order to compute the average ROC curve. In Figure 4 are ROC curves for different k's. In the first row of Figure 4 are the individual ROC curves for each run of simulation with different cluster number k's. The second row are their corresponding average ROC curves. It shows that the hybrid methods can have very good sensitivity and specificity. These results suggest that if we cannot foresee the geometric structure, and the negative examples outnumbers the positive examples, then we can divide the negative examples into smaller sub-sets and apply Procedure I to construct a hybrid classifier, which could have a more stable prediction accuracy.

Study 2:
In Study 2, the size of training and testing sets are the same as in Study 1. They are also generated from bivariate normal, but with mean vector $\mu = (d \times i, o)^t$, for $i = 0, \cdots, 4$, and $d = 0.5, 1, 1.5, 2, 3, 4$. The picture of data sets with different d's are shown in Figure 6. That is there are five groups with the same distribution, but different location parameter d's.

For each i, $i = 0, \cdots, 4$, we generate a group a data. Similar to that in Study 1, there are 200 examples for each class, and 1000 example in total in the training set. Testing data are generated in the same manner. But we now are treat them as multi-class problems. That is our purpose is to classify testing examples to one of these 5 groups.

Using Procedure II — that is to integrate SVM local classifiers with a ordinal logistic regression model. The results for $d = 0.5, 2$ and 4 are summarized in Table 3. The column 3 to 5 are Accuracy, Precision and Recall for each groups (Group 0 to 4) and different d's,

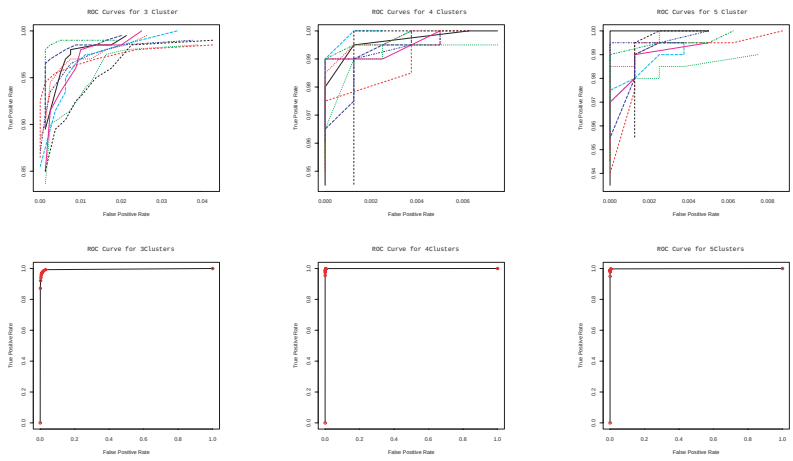


Figure 4: ROC curves for $k = 3, 4, 5$ and $d = 4$; First row are curves for 10 individual simulations of each k, and the second row are the corresponding mean ROC curves of the first row.

d	Clusters K	True Neg.	False Neg.	False Pos.	True Pos.	Accuracy(%)
0.5	3	708	168	92	32	74.0
	4	711	164	89	36	74.7
	5	691	159	109	41	73.2
1.0	3	682	137	118	63	74.5
	4	668	121	132	79	74.7
	5	658	136	142	64	72.2
1.5	3	704	96	96	104	80.8
	4	715	86	85	114	82.9
	5	708	92	92	108	81.6
2	3	739	65	61	135	87.4
	4	739	60	61	140	87.9
	5	736	56	64	144	88.0
3	3	777	23	23	177	95.4
	4	784	15	16	185	96.9
	5	785	14	15	186	97.1
4	3	794	5	6	195	98.9
	4	799	1	1	199	99.8
	5	799	0	1	200	99.9

Table 2: SVM + Logistic Regression With Clustered Negative Examples

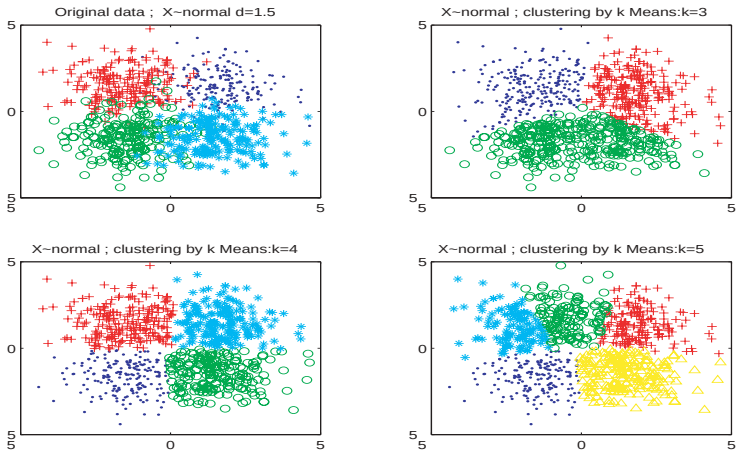


Figure 5: K-mean Clustering of Negative Examples With $d = 1.5$ (K=3,4,5); Only negative examples are shown in this picture.

respectively. The F in the last columns of Table 3 is a harmonic mean of precision and recall, which is a common measurement used in the information retrieval/text classification.

As expected, we found that the prediction accuracy of the middle classes are lower than those of the classes in the two ends. But all the prediction accuracy are increasing when d is getting larger.

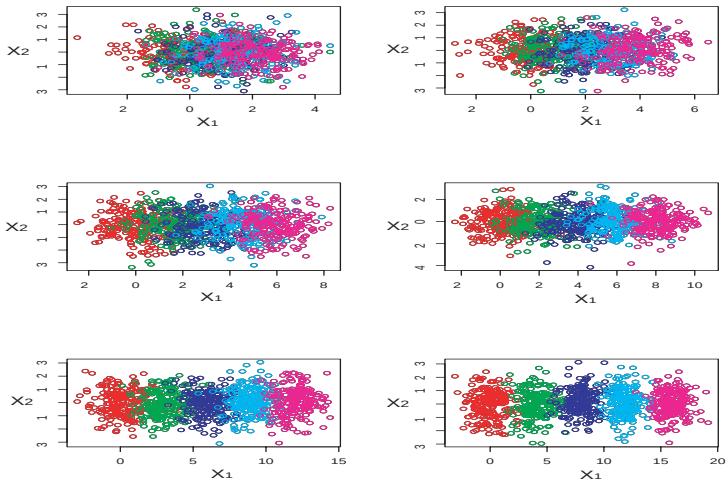


Figure 6: Ordinal Data: (d=0.5, 1, 1.5, 2, 3, 4)

d	Group	Accuracy	Precision (P)	Recall (R)	$F = \frac{2}{\frac{1}{P}+\frac{1}{R}}$
0.5	0	0.802	0.506	0.420	0.459
	1	0.513	0.268	0.830	0.405
	2	0.351	0.218	0.870	0.349
	3	0.719	0.245	0.195	0.217
	4	0.815	0.506	0.420	0.459
2	0	0.954	0.881	0.890	0.886
	1	0.877	0.671	0.755	0.711
	2	0.862	0.642	0.700	0.670
	3	0.873	0.657	0.765	0.707
	4	0.945	0.901	0.815	0.856
4	0	0.991	0.980	0.975	0.977
	1	0.978	0.932	0.960	0.946
	2	0.980	0.946	0.955	0.951
	3	0.984	0.947	0.975	0.961
	4	0.991	0.970	0.985	0.978

Table 3: Ordinal Data

4.2 UCI Data Set

We apply the hybrid method to two data sets obtained from UCI Machine learning archive — Wisconsin Diagnostic Breast Cancer (WDBC) and Pen Digit Recognition.

We apply Procedure I to the WDBC data set and apply Procedure II to the Pen Digit Recognition data set. Below are summarization of our results.

Wisconsin Diagnostic Breast Cancer:
This data set is known to be linear separable. The results shown below is obtained from 10-fold cross-validation for the hybrid method.

As the usual 10-fold cross-validation, we use one sub-group for testing, and the other 9 sub-groups for training. By using each sub-group in the training set, we can train an SVM classifier. The final ensemble classifier is constructed by integrating these 9 classifiers into one according to the steps of **Procedure I**. The results we have here is slightly better than the 97.5%, which is the accuracy given in the description of this data set (see Table 4).

Pen Digit Recognition:
In this data set, we have ten classes labelled 0 to 9. Let C_i denote the class of digit i, for $i = 0, \cdots, 9$. Then we construct 9 local classifiers using C_i versus $\bigcup_{j=0}^{i-1} C_j$ for $i = 1, \cdots, 9$. Here we use an ordinal logistic regression to integrate these local SVM classifiers according to **Procedure II**. We have 97.8% accuracy, the detailed is summarized in Table 5.

To apply the usual one-against-one voting scheme to an n classes multi-class classification problem, we have to construct $C_2^n = n(n-1)/2$ classifiers. That means there will need 45 local classifiers in this case. By using **Procedure II**, we only need 9 of them, so it is more efficient when the number of classes is large.

Group	True Positive	False Positive	False Negative	True Negative	Precision
1	35	0	1	21	0.983
2	35	1	1	20	0.965
3	36	1	0	20	0.983
4	36	1	0	20	0.983
5	36	0	0	21	1.000
6	36	0	0	21	1.000
7	33	2	2	19	0.927
8	35	0	0	21	1.000
9	35	0	0	22	1.000
10	36	1	0	21	0.983

Table 4: WDBC 10-Fold Cross-Validation; Average:0.982, Standard Deviation:0.023

(Ordinal)	Prediction									
True	0	1	2	3	4	5	6	7	8	9
0	349	0	8	0	0	0	0	0	6	0
1	0	353	9	0	1	0	0	1	0	0
2	0	2	362	0	0	0	0	0	0	0
3	0	3	0	331	0	0	0	0	0	2
4	0	3	0	0	361	0	0	0	0	0
5	0	1	0	5	0	328	0	0	0	1
6	1	0	0	0	0	0	335	0	0	0
7	0	18	0	0	2	0	2	342	0	0
8	1	0	0	0	0	2	0	0	333	0
9	0	4	0	0	0	0	0	2	1	329

Table 5: Pen Digit Recognition — Average Accuracy = 0.978

5. Summary and Discussion

In this paper, we propose a hybrid method that utilizes the local SVM classifiers as its foundations, and then integrates them using a logistic regression model into a more robust classifier. Using logistic regression techniques allows us to take advantage of the statistical theory of model selection methods, such as stepwise regression, added-variable plot etc., which have been well studied by statisticians. Based on the logistic model, we can have an appropriate weights to combine those local classifiers.

Moreover, with its probability outputs, this method can be apply to multi-class problems as well as multi-labelling problems. In order to have the best performance, we need a suitable clustering algorithm to divide the training set in the feature space. Hence, a kernelized clustering algorithm will be suitable for this purpose (see Bradley and Fayyad (1998) , and Zhang and Rudnicky).

Due to the geometric property of SVM, to integrate these local classifiers with the linear logistic model is very effective. Certainly, it is possible to replace the linear logistic regression models by other binary/categorical regression models in our procedures; such as

probit model or nonlinear logistic model (see **?**). For example, it might be appropriate to use nonlinear logistic models to integrate local classifiers obtained from nonlinear Fisher discriminant analysis. Moreover, It is possible to apply this idea to integrating classifiers based on other classification methods as long as we can "extract" some kinds of distances from those classifiers.

Appendix A. Logistic Regression

The dichotomous (polytomous) logistic regression models are common statistical models to modelling the binary (categorical) response variables. Suppose $Y \in \{0, 1, \cdots, k\}$, for some integer $k \geq 1$, be a categorical response variable, and $X \in R^p$ denotes the corresponding vector of covaraites/regressors. Assume that for $i = 1, \cdots, k$, (X, Y) satisfies that

$$\frac{P(Y = i|X)}{P(Y = 0|X)} = f_i(X, \beta) = \beta_{0,i} + X^t \beta_i. \tag{3}$$

Then for $k = 1$ and $k > 1$, the equation (3) called a dichotomous and a polytomous logistic regression models, respectively. Due to the nonlinearity, the estimate of coefficients must be calculated by iteration algorithms. When k is large, the calculating of regression coefficients of polychotomous regression model will take a long time. Begg and Gray (1984) have shown that estimation of the regression coefficients of a polytomous regression with $k + 1$ categories can be obtained by estimating the regression coefficients of k individualized binary logistic models, since these coefficients are algebraically equal. This result can shorten the computational time of calculating the coefficients of polytomous logistic regression;

If we believe that the response variable Y is ordinal, then instead of (3), we might use the following ordinal logistic regression model (see McCullagh (1989)): For $i = 1, \cdots, k$,

$$\frac{P(Y = i|X)}{P(Y < i|X)} = f_i(X, \beta) = \beta_{0,i} + X^t \beta_i. \tag{4}$$

In the ordinal case above, Begg and Gray(1984)'s method cannot be applied. With slight modification of their idea, we can still obtained the estimation of the coefficients of this ordinal logistic regression by the individualized binary logistic regression. The probabilities p_i's can be calculated by a backward manner.

Theorem A.1 *The estimation of the regression coefficients of a ordinal polychotomous regression with $k + 1$ categories as in (4) can be obtained by estimating the regression coefficients of k individualized binary logistic models, since these coefficients are algebraically equal.*

Proof.

For $i = 0, \cdots, k$, let P_i denote probability $P(Y = i|X)$, and, for $i = 1, \cdots, k$, let,

$$P_i' = \frac{P_i}{P(Y \leq i|X)} = \frac{P_i}{\sum_{0 \leq j \leq i} P_j}.$$

Now, for each $i \geq 1$,

$$\log\left(\frac{P_i'}{1 - P_i'}\right) = \log\left(\frac{P_i}{P(Y \leq i|X)}\right) = f_i(X, \beta_i) = \beta_{0,i} + X^t \beta_i \tag{5}$$

is just a binary logistic regression. Hence, for each i, these β_i's can be estimated by using individualized binary regression technique, and so are P_i'. Note that $P_k = P_k'$, since $\sum_{0 \le j \le k} P_j = 1$. Hence P_k and its correspondent coefficients $\beta_{0,k}$ and β_k can be obtained from the k-th individualized binary logistic regression. The rest of P_i's $(P_0, \cdots, P_{k-1})$ can be calculated in a backward order as shown in below: For $i < k$,

$$P_i = P_i'(\sum_{j \le i} P_j) = P_i'(1 - \sum_{j > i} P_j) \tag{6}$$

■

Remark 2 *It is known that when the data are completely separated, the maximum likelihood estimate for the logistic regression models will not converge. In this case, we can apply the method of Firth (1993) to fit the logistic regression by maximum penalized likelihood with a Jeffreys invariant prior (see also Firth (1992), Heinze and Schemper (2002)) such that the iteration algorithm will always converge.*

Acknowledgments

I would like to acknowledge support for this project from the National Science Console (NSC grant 91-2118-M-001-007).

References

Eric Bauer and Ron Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36(1-2):105–139, 1999.

C.B. Begg and R. Gray. Calculation of polychotomous logistic regression parameters using individualized regression. *Biometrika*, 71:11–18, 1984.

C. L. Blake and C. J. Merz. UCI repository of machine learning databases, 1998. URL `http://www.ics.uci.edu/$\sim$mlearn/MLRepository.html`.

Paul S. Bradley and Usama M. Fayyad. Refining initial points for K-Means clustering. In *Proc. 15th International Conf. on Machine Learning*, pages 91–99. Morgan Kaufmann, San Francisco, CA, 1998.

D. Firth. Bias reduction, the jefferys prior and glim. In R. Gilchrist Fahemeir, L. Francis and G. Tutz, editors, *Advances in GLIM and Statistical Modelling*, pages 91–100. Springer-Verlag, New York, 1992.

D. Firth. Bias reduction of maximum likelihood estimates. *Biometrika*, 80:27–38, 1993.

Trevor Hastie and Robert Tibshirani. Classification by pairwise coupling. In Michael I. Jordan, Michael J. Kearns, and Sara A. Solla, editors, *Advances in Neural Information Processing Systems*, volume 10. The MIT Press, 1998.

Heinze and Schemper. A solution to the problem of separation in logistic regression. *Statist. Med.*, 21:2109–2149, 2002.

N. Japkowicz. The class imbalance problem: Significance and strategies. In *Proceedings of the 2000 International Conference on Artificial Intelligence (IC-AI'2000)*, volume 1, pages 111–117, 2000a.

N. Japkowicz. Learning from imbalanced data sets: a comparison of various strategies, 2000b. URL `citeseer.nj.nec.com/japkowicz00learning.html`.

Miroslav Kubat, Robert Holte, and Stan Matwin. Learning when negative examples abound. In *European Conference on Machine Learning*, pages 146–153, 1997.

P. McCullagh. *Generalized Linear Models, 2nd Edition*. Chapman and Hall, New York, 2nd edition, 1989.

D. Optiz and R. Maclin. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, 11:169–198, 1999.

Foster J. Provost and Tom Fawcett. Robust classification systems for imprecise environments. In *AAAI/IAAI*, pages 706–713, 1998.

K. Tumer and J. Ghosh. Error correlation and error reduction in ensemble classifiers. Technical report, Department of Electronical and Computer Engineering, University of Texas, Austin, 1996.

V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, NY, 1995.

G. Weiss and F. Provost. The effect of class distribution on classifier learning, 2001.

Rong Zhang and Alexander I. Rudnicky. A large scale clustering scheme for kernel k-means. Proc. of International Conference on Pattern Recognition 2002.

Ji Zhu and Trevor Hastie. Kernel logistic regression and the import vector machine. URL `citeseer.nj.nec.com/543493.html`.