Table of Contents

Summary of the Data Mining Contest for the IEEE International Conference on Data Mining, Pisa, Italy 2008 by T.J. Stocki, N. Japkowicz, G. Li, R.K. Ungar, I. Hoffman, J Yi .................................................................1

Summary of ICDM’08 Data Mining Contest by W.Fan, E. Zhong, S. Xie, Y. Huang, K. Zhang, J.Peng, J. Ren.................................................................7

Classification of Imbalanced Data with Random Sets and LogitBoost Algorithm by V. Nikulin...........................................................................................................9

An Instance-based Nearest-neighbor Approach to Classifying Nuclear Explosion Data by D. Oviatt, D. Frandsen, K. Clement, and C. Giraud-Carrier...12


Report for ICDM 2008 Data Mining Contest by Z. Zhang.................................................20
Summary of the Data Mining Contest for the IEEE International Conference on Data Mining, Pisa, Italy 2008.

T.J. Stocki¹, N. Japkowicz², G.Li², R.K. Ungar¹, I. Hoffman¹, J. Yi¹
¹Radiation Protection Bureau, Health Canada
²School of Information Technology, University of Ottawa

Abstract

The IEEE ICDM 2008 Data Mining Contest is, simply put, about keeping the world safe using data mining. This contest is about developing and testing data mining techniques to verify worldwide compliance of the global ban on nuclear tests. Such tests can be detected by measuring the amount of special xenon isotopes. Obviously, it's not just that simple; these isotopes are also emitted during various legal activities.

1. Introduction

All Compliance verification of the Comprehensive Nuclear-Test-Ban Treaty (CTBT), when the treaty enters into force, will employ four remote sensing technologies to detect nuclear explosions [1]. Only radionuclide detection can unequivocally establish that an explosion was due to a nuclear detonation. Radioactive noble gas (the following isotopes: Xe-131m, Xe-133m, Xe-133, and Xe-135) are sampled and measured in a procedure called radionuclide monitoring[2,3]. Different relative combinations of these isotopes correspond to different signatures that can be mapped to distinct sources (such as nuclear power plants, medical isotope production facilities, or various types of weapons).

In the first few weeks after an explosion the relative concentrations of the four isotopes are expected to be released in “fingerprint” relative concentrations quite distinct from other background sources. The problem of attributing a specific observation of airborne concentrations of radioxenon to an explosion is twofold. Firstly, since the CTBT stations are not located at the source of the explosion, the radioxenon is detected at a location which can be well over a thousand kilometres away. This atmospheric transport process can take weeks, which can increase the complexity of this signature. Secondly, one can never observe radioxenons emitted purely from an explosion source but admixtures of this gas with the radioxenons released from all background sources. These 2 points above constitute an interesting data mining problem for the Preparatory Commission for the Comprehensive Nuclear-Test-Ban Treaty Organization (CTBTO).

2. Description of the dataset used.

Radioxenon measurements from four five CTBTO monitoring sites were provided. These were comprised of a few hundred sets of observations of the four species for each site. A synthesized a set of explosion observations at these same sites were added to actual radioxenon concentrations caused by background sources. The data sets are composed of two classes, Background (B), and Background plus Explosion (B+E). Each type has a set of quadruplets representing the four activity concentrations of Xe-131m, Xe-133m, Xe-133, and Xe-135 for a given air sample.

3. Description of the computational tasks

Two versions of data sets were provided. The first had each datum described according to station of origin, a unique randomly assigned tracking number allowing the contest evaluators trace the datum back to the original scenario of explosion release, whether it is Background or whether it is Background plus Explosion (labeled). The second version had each datum described by station of origin using the same stations as the first data set and a unique randomly assigned tracking number allowing the contest evaluators to trace the datum back to the original scenario of explosion release. The second set of data contained cases of B or B+E but this were unknown to the contestants (unlabelled) The first version of the data was employed in Tasks 1 and 2. The final version of the data will be employed in Task 3.
3.1. Task 1

The first task was to classify, as accurately as possible, the results as Background or Explosion over the entire set of stations (V, W, X, Y and Z) provided with one classifier. Contestants were allowed to combine data as they saw fit. They were allowed to separately tune classifier parameters for each station but they were not allowed to not have separate classifier parameter types for each station nor separate classifiers. Contestants were allowed to report on more than one classifier for this task.

3.2. Task 2

In the second task, conversely, the contestants were requested to identify an optimal algorithm for each station given (V, W, X, Y and Z).

3.3. Task 3

In the third task, the contestants applied the classifiers developed in Tasks 1 and 2 using the second data set and reporting their results for evaluation. The primary goal of this contest is to produce methods that are broadly applicable over different station background measurement distributions and explosion source hypotheses. The best methods will also have a very efficient learning curve. Furthermore, methods more proficient in properly categorizing data arising from specific classes of explosion release hypotheses or station background types, because these methods add a forensic or diagnostic dimension. In more detail, Task 3 was subdivided into three tests whose description now follows.

3.3.1. Task 3 Test 1

In Task 3, test 1, the contestants classified an unlabelled data set comprised of similar data as used for the development of their classifiers. In this test, the contestants used the same classifier they developed in Task 1 on the unlabelled data set. This test ensures the performance of the methods for similar concentrations of radioxenon sampled from the same distribution as the training set but not necessarily the same proportions of B and B+E cases. The values of AUC obtained on this data set are reported in Section 6.

3.3.2. Task 3 Test 2

For the actual data used to develop their classifiers, the contestants were requested to classify the unlabelled data set with their classifiers for tuning with 20% of their employed training data set (the labelled data set): then similarly for 40%, 60%, 80%, and 100%. This test examines the efficiency in use of data required to tune the classifiers. Of course, recognition will be given to contestants who employ relative relatively small subsets of the training data provided in the first instance to develop their classifiers.

3.3.2. Task 3 Test 3

For the classifier developed under Task 1 only, the contestants were asked to provide results for a factorial analysis of the sensitivity of the effectiveness of their method to small changes in their parameter values. Hence, the contestants were requested to provide results for 2n trials where “n” is the number of numerical parameters used by their methods. Parameter jumps on the order of 10% are requested where, as appropriate and as judged by the contestant, it is:

- * 10% of the full range of the parameter possible values or
- * 10% of the value parameter value itself or
- * for integer values, the same considerations as above rounded to the nearest integer.

For example, if the classifier has 3 numerical parameters, $2^3$ or 8 trials are needed for all possible combinations of high and low parameters. This test is done on the labelled data set.

4. Awards

The contestants will not be judged on each task, rather there will be three types of prizes: Contest Crown, Most Muscular, and Kangaroo Prizes. The Contest Crown prize will be awarded to the classifier judged to have best over all performance (best balance). The Most Muscular prize is for the classifiers with highest AUC scores for the full data set and by station. Kanagaroo Prizes will be given to classifiers that are talented in unusual or unexpected respects. This evaluation is underway at the time of writing this paper.

As a reminder, the ROC (Reciever Operating Characteristic) is an important measure for visualization of the performance of classifiers. It is drawn by the true positive rate on the y-axis versus false positive rate on the x-axis. The ROC has been shown to be more effective than other performance

---

1 The contestants did not have access to the labels of this data set. It was a blind classification exercise.
measures in some practical cases, where the ROC provides a way to select optimal models and to discard suboptimal ones. An example of a ROC curve is shown in Figure 1. As a statistical summary of ROC, the Area Under the ROC Curve (AUC) is often used for comparison of performance for discrimination among learned classifiers.

![Figure 1: The ROC curve used to determine the AUC.](image)

5. Summary of Submissions

The contest was attempted by ten different teams who solved the problem using a variety of methods that differed from one another with respect to the base classifiers, the classifier combination and the pre-processing techniques they used. Only one of these submissions [Hovepian] could not be processed because it was entered on the uncorrected data. [Hovepian]’s results are, thus, not comparable to the other teams.

A summary of all the submissions is given here. For more detail about each approach, please refer to the individual summaries included following this overview, in alphabetical order.

Most teams employed a number of pre-processing methods for dealing with the strong imbalance displayed by the data set as well as for transforming the data set. The various pre-processing methods attempted were the following, divided along the lines of the problems they seek to address:

Data Sampling to address the class imbalance problem:

- Random Over-Sampling [Oviatt et al.]
- Random Under-sampling [Fan et al.]
- Ensembles for Random Under-sampling [Zhang], [Nikulin]
- Informed Under-Sampling [Wu et al.]

Feature Set Transformations:

- Feature Set Expansion:
  - Attempt to include background domain information, though none was added in the end. [Oviatt et al.]
  - Addition of variables for trend identification [Hariharn et al.]
  - Addition of feature ratios [Lath]
  - Feature expansion for unconvoluting the information contained in single features and increasing domain dimensionality [Fan et al.]

Dealing with zero-values:

- Heuristic feature transformation [Jarozewicz]
- Ave-Normalization [Wu et al.]

Feature Transformation and Selection:

- Principal Component Analysis [Lath]
- Manual variable transformations and interaction effects detection [Hariharn]
- Nothing [Zhang, 2008], [Nikulin, 2008]

---

2 If the summary for a particular approach is not provided, this means that it was not available in publishable form, at the time this report was compiled.
6. Results

We will now present the results obtained by the various teams. This section is divided into two parts. In the first part, we summarize the results obtained on the labeled data sets. In the second part, we return the results obtained on the blind data set. Please, note that the number of results returned varies from one task to the next. This is because each team was allowed to enter the parts of the competition they were most comfortable with and to ignore the others.

6.1 Results obtained on the labeled data sets

The results obtained on Task 1 are summarized in Table 1 below; those obtained on Task 2 are summarized in Table 2. In all cases, we only return the AUC, which is the most reliable performance metric in case of class imbalances. The results are ranked from best performance to worst.

<table>
<thead>
<tr>
<th>Team</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jarozewicz</td>
<td>0.8589</td>
</tr>
<tr>
<td>Fan et al.</td>
<td>0.750833</td>
</tr>
<tr>
<td>Zhang</td>
<td>0.6968</td>
</tr>
<tr>
<td>Hariharan et al. (Best)</td>
<td>0.6677</td>
</tr>
</tbody>
</table>

Table 1: Task 1 Results

We recall that the first task consisted of classifying the output of all the stations with a single classifier, which could, potentially, be tuned on different stations. In this task, [Jarozewicz]' system performs best by far, as it obtains an AUC of 0.86, which is over 0.10 AUC points ahead of the next best classifier. The second best result was obtained by [Fan et al.], with an AUC of 0.75. The next three results fall well below the best two results.

The results for Task 2 are reported in Table 2, below. We recall that Task 2 consisted of identifying an optimal classifier for each station. We rank the results according to their average AUC on all the stations, but we display the specific AUC obtained on each station for further analysis.

<table>
<thead>
<tr>
<th>Team</th>
<th>Avg</th>
<th>V</th>
<th>W</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fan et al.</td>
<td>0.83</td>
<td>0.84</td>
<td>0.94</td>
<td>0.84</td>
<td>0.85</td>
<td>0.66</td>
</tr>
<tr>
<td>Zhang</td>
<td>0.78</td>
<td>0.84</td>
<td>0.88</td>
<td>0.71</td>
<td>0.81</td>
<td>0.64</td>
</tr>
<tr>
<td>Jarozewicz</td>
<td>0.77</td>
<td>0.82</td>
<td>0.89</td>
<td>0.71</td>
<td>0.81</td>
<td>0.62</td>
</tr>
<tr>
<td>Wu et al.</td>
<td>0.75</td>
<td>0.87</td>
<td>0.97</td>
<td>0.65</td>
<td>0.75</td>
<td>0.51</td>
</tr>
<tr>
<td>Hariharan et al. (Best)</td>
<td>-</td>
<td>0.82</td>
<td>0.85</td>
<td>0.71</td>
<td>0.75</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 2: Task 2 Results

6.2 Results obtained on the blind data set

We now report on the results obtained on the blind test. This time, the contestants were asked to apply the classifier they obtained in Task 1 to an unlabeled data set and return their labelling. We computed the AUC obtained on this data set by matching their response to the expected ones. We generally give more credence to these results than the earlier ones since the contestants did not get a chance to tune their systems to this data the way they might have with the labelled testing set. The results each team obtained are reported in Table 3. In the case where the mention “(Best)” appears near the name of the team, this means that the team submitted several versions of their work and that we only retained the one which obtained the best results.

<table>
<thead>
<tr>
<th>Team</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zhang</td>
<td>0.7642</td>
</tr>
<tr>
<td>Nikulin</td>
<td>0.7453</td>
</tr>
<tr>
<td>Wu et al. (Best)</td>
<td>0.7316</td>
</tr>
<tr>
<td>Fan et al.</td>
<td>0.7292</td>
</tr>
<tr>
<td>Jarozewicz</td>
<td>0.7222</td>
</tr>
<tr>
<td>Hariharan et al. (Best)</td>
<td>0.7169</td>
</tr>
<tr>
<td>Oviatt et al.</td>
<td>0.5031</td>
</tr>
</tbody>
</table>

Table 3: Results on Task 3—Test 1

The best results obtained on Task 2 were obtained by the team of [Fan et al.]. The next three best results are similar in average, although, [Wu et al.]-’s results are much less stable than the other two, when considering the stations separately. The last entry is incomplete and falls much below the results of the others.

The results we obtained places [Zhang] at the head. Aside from [Oviatt et al.]-’s results which are very low, all the results appear close to one another with a difference of slightly over 0.05 accuracy points recorded between the best and the worst showings.
One issue we would like to point out here is the discrepancy between the ranking of teams according to the results obtained on the labeled data set and that obtained on the blind data set. The 0.10 lead in AUC performance reported by [Jarożewicz] over the next best classifier, on the labeled data, did not translate on the blind data set as his system placed 5th, over 0.04 AUC points behind the best result. [Fan et al.]’s system which reported the second best AUC on the labeled data set, similarly ended up in 4th position on the blind data set, although the disparity in the two results is not as great as in the case of [Jarożewicz]. It is [Zhang] and [Nikulin], two teams that obtained modest results on the labeled data sets, which ended up doing best on the blind data set.

This observation presents an interesting lesson regarding the results reported in the machine literature that boast large improvements over other published or re-computed results. It is very possible that such results may not be as impressive as they may appear in the papers, when deployed in real-life settings such as those simulated by our blind test.

Only four teams entered Task 3—Test 2, which studied the efficiency of the systems in terms of data use. The results are displayed in Table 4.

<table>
<thead>
<tr>
<th>Teams</th>
<th>20%</th>
<th>40%</th>
<th>60%</th>
<th>80%</th>
<th>100%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fan et al. (Best)</td>
<td>0.73</td>
<td>0.73</td>
<td>0.73</td>
<td>0.73</td>
<td>0.73</td>
</tr>
<tr>
<td>Hariharan et al.(Best)</td>
<td>0.66</td>
<td>0.69</td>
<td>--</td>
<td>0.71</td>
<td>0.71</td>
</tr>
<tr>
<td>Wu et al. (Best)</td>
<td>0.63</td>
<td>0.72</td>
<td>0.73</td>
<td>0.71</td>
<td>0.73</td>
</tr>
<tr>
<td>Oviatt et al.</td>
<td>0.52</td>
<td>0.50</td>
<td>0.52</td>
<td>0.50</td>
<td>0.50</td>
</tr>
</tbody>
</table>

Table 4: Results on Task 3—Test 2

In all cases, the results show that the system came close to their full potential with 40% of the available data. [Fan et al.]’s system even reaches its full potential with 20% of the available data.

Two teams reported results on Task 3—Test 3. [Wu et al.] whose system displayed a difference range of 0.05 AUC points from the worst to the best parameter setting and [Fan et al.] who point out that their system is parameter free.

7. Linking Systems’ performance to their Characteristics

We now turn to the description of the characteristics displayed by the two systems that performed best on Task 3—Test 1.3

The two systems that performed best on the blind data set both use Adaboost. [Zhang]’s system uses Adaboost to combine Decision stumps, while [Nikulin]’s uses Adaboost to combine Logistic regression. They are the only two systems that combine the results of ensembles obtained by Random Under-sampling. And neither team performed any kind of feature selection.

This is a very interesting result, if it is not a coincidence, as it clearly suggests optimal approaches for dealing with the problem of Radioxenon in the atmosphere.

The next four teams in the blind test part of the competition all did quite well as well, and also present interesting algorithms re-sampling and feature transformation ideas.

The one system that did not perform well on the blind data set is the one by [Oviatt et al.]. This is not fully surprising as most research on the class imbalance problem suggests that oversampling the data causes overfitting and, consequently poor performance (see [4], for example).

8. Competition Results and Conclusion

As discussed in Section 4, three types of prizes were considered: Contest Crown, Most Muscular, and Kangaroo Prizes. This section will announce the Contest Crown and Most Muscular prizes. The Kangaroo prize will be announced after further analysis is performed.

The Contest Crown goes to [Fan et al.] who were consistently best in Task 2, and who ranked well enough in Tasks 1 and 3. Another added advantage of their submission, which contributed to our consideration for this best balanced prize, is that their system does not require tuning and performed well with 20% of the data.

3 Please, note that these systems will not necessarily win the competition, since, as discussed in Section 4, the awards are given according to more complex criteria, than simple high accuracy. The competition prizes are announced in Section 8.
The Most Muscular Prize goes to [Zhang] for his top results on Task 3—1 and his good results on Task 2.

These decisions were not easy to make, and we would like to acknowledge the very good performance also obtained on the blind data set by Nikulin—though, unfortunately, we did not have enough results to judge the overall performance of the system—, Wu et al.—who did quite well, overall, and was the only team to participate in all the tasks—, Jarozewicz—who also did well, except for the disparity in the results obtained on the labelled and unlabeled test sets—and Hariharan et al.—who also did acceptably well on the blind data set. Lastly, despite the fact that they did not obtain good results, we applaud the enthusiasm of the Oviatt et al. student team.

In conclusion, we thank all the teams for their participation which will help the world community progress towards the implementation of the Comprehensive Nuclear Test Ban Treaty, and we hope that this experience was as rewarding for them as it was for us!

9. References

1 Introduction

In order to verify the compliance of the Comprehensive Nuclear-Test-Ban Treaty (CTBT), remote detection and measurement of radioactive forms of noble gas called radioxenon is employed. These gas is emitted from nuclear sources including nuclear explosion. The principle of such detection is that certain combinations of the four radioxenon can be “fingerprints” of a nuclear explosion. However, the problem is complicated in two aspects. First, the detection station could be well over a thousand kilometres away from the explosion location, the gas emitted in an explosion could be remarkably degraded due to radioactive decay during weeks of atmospheric transport process, making the “fingerprints” less likely to be detected. Second, there could be other radioactive sources emitting radioxenon, such as nuclear power plants, medical isotope production facilities, or various types of weapons. The radioxenon given by sources other than nuclear explosion is treated as background. The general task of the contest is to devise methods to distinguish between those radioxenon measurements that are due purely to normal environmental emissions or background (B) from those measurements that contain the signature of an explosion combined background (B+E).

To participate in the contest, we have employed non-parametric methods based on ensemble of decision trees. These methods manifest their advantages when there is no prior knowledge on the true distribution of the target problem. Further, these methods are simple and efficient, and achieve overall reasonable performance for many problems. In the following sections, we give detailed description and present results for each task (if known at the time of this paper’s submission).

2 The chosen modeling technique: bRDT

The bRDT is the model averaging of RDT and BC44, in which, RDT is Random Decision Tree [Fan et al., 2003] and BC44 is Bagged C4.4 [Zhang et al., 2006]. Both are examples of decision tree ensembles, where the conditional probability of \( P(y|x) \) is computed by averaging the outputs of trees in the ensemble. Given feature set \( \{F_1, \ldots, F_k\} \), RDT chooses an un-used feature in the decision path randomly. Categorical features can be used only once in given any decision path, but continuous features can be chosen more than once and each time with a randomly chosen decision threshold. The statistics kept in each random tree are consistent with the training data. BC44 is a bagging method [Breiman, 1996] where each base classifier is built using C4.4 [Provost and Domingos, 2003].

3 Task 1 and 2

In task 1, contestants are required to train one classifier to classify the training data of each station (V,W,X,Y,Z). In task 2, optimal model can be picked for each station and different type of classifier can be applied to individual stations. In addition, contestants are allowed to “manipulate” over the original training data, such as sampling or combination to obtain each model’s optimal training data.

- In task 1, for the first 4 stations (V, W, X, Y), we use the original training data, while for station Z, we down-sample the negative examples, such that the size of negatives is roughly three times the size of positives. The sampling method is described in [Gao et al., 2007]. After constructing training sets for each stations, 5 bRDT models are built on these training sets.

- In task 2, the training sets are the same as in task 1. The optimal classifier for station V is BC44 instead of bRDT, but bRDT still has the best performance in the remaining 4 stations (W, X, Y, Z).

The results are summarized in Table 1. Since the classifier bRDT for station V in task 1 is not the optimal classifier for this given station in task 2, thus there are two rows 2
Table 1. Result of Task 1 and 2

<table>
<thead>
<tr>
<th>Station</th>
<th>AUC</th>
<th>Acc</th>
<th>TPR</th>
<th>FPR</th>
<th>Prec</th>
<th>Reca</th>
<th>F1m</th>
<th>MCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>V</td>
<td>0.8406</td>
<td>0.8204</td>
<td>0.5478</td>
<td>0.1598</td>
<td>0.1987</td>
<td>0.5478</td>
<td>0.2917</td>
<td>0.2501</td>
</tr>
<tr>
<td>W</td>
<td>0.8437</td>
<td>0.8163</td>
<td>0.6087</td>
<td>0.1687</td>
<td>0.2071</td>
<td>0.6087</td>
<td>0.3091</td>
<td>0.2768</td>
</tr>
<tr>
<td>X</td>
<td>0.9378</td>
<td>0.7748</td>
<td>1</td>
<td>0.2294</td>
<td>0.0741</td>
<td>1</td>
<td>0.1379</td>
<td>0.2389</td>
</tr>
<tr>
<td>Y</td>
<td>0.8622</td>
<td>0.2524</td>
<td>0.0766</td>
<td>0.2488</td>
<td>0.2524</td>
<td>0.2506</td>
<td>0.1747</td>
<td>0.7793</td>
</tr>
<tr>
<td>Z</td>
<td>0.8487</td>
<td>0.9661</td>
<td>0.025</td>
<td>0.0017</td>
<td>0.3333</td>
<td>0.025</td>
<td>0.0465</td>
<td>0.0837</td>
</tr>
</tbody>
</table>

Table 2. Result of test 3 in Task 3

<table>
<thead>
<tr>
<th>Station</th>
<th>AUC</th>
<th>Acc</th>
<th>TPR</th>
<th>FPR</th>
<th>Prec</th>
<th>Reca</th>
<th>F1m</th>
<th>MCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>V</td>
<td>0.84</td>
<td>0.82</td>
<td>0.55</td>
<td>0.16</td>
<td>0.2</td>
<td>0.55</td>
<td>0.29</td>
<td>0.25</td>
</tr>
<tr>
<td>W</td>
<td>0.94</td>
<td>0.77</td>
<td>1</td>
<td>0.23</td>
<td>0.07</td>
<td>1</td>
<td>0.14</td>
<td>0.24</td>
</tr>
<tr>
<td>X</td>
<td>0.84</td>
<td>0.82</td>
<td>0.55</td>
<td>0.16</td>
<td>0.2</td>
<td>0.55</td>
<td>0.29</td>
<td>0.25</td>
</tr>
<tr>
<td>Y</td>
<td>0.85</td>
<td>0.97</td>
<td>0.03</td>
<td>0</td>
<td>0.33</td>
<td>0.03</td>
<td>0.05</td>
<td>0.08</td>
</tr>
<tr>
<td>Z</td>
<td>0.66</td>
<td>0.87</td>
<td>0.14</td>
<td>0.06</td>
<td>0.2</td>
<td>0.14</td>
<td>0.16</td>
<td>0.1</td>
</tr>
</tbody>
</table>

and 3, corresponding to task 1 (bRDT) and task 2 (BC44) respectively. In addition, please note that in task 2, the optimal classifiers for station W, X, Y, Z are the same bRDT as those in task 1, thus there’s only one row for these 4 stations.

4 Task 3

In task 3, the contestants will apply their methods developed in Task 1 and 2 to predict over a second unlabeled data set. In test 1, the unlabeled data set with the same distribution as the training set is given, but it may not necessarily have the same proportions of B and B+E cases. This test set is classified by classifiers developed in Task 1 and 2. In test 2, contestants are requested to report performance using different percentage of employed training data set for learning curve study. In test 3, results of factorial analysis on the sensitivity of parameter values are required.

Non-linear Feature Expansion There are a large number of training data, but for each example, only 4 features are provided. Under this situation, several factors can make these small number of features inadequate. Thus, we propose a “feature expansion" method using non-linear feature mapping to “zoom” the feature set. For each feature \( f \), we obtain two expanded features, \( f_2 \) and \( \ln (f + 1) \). For example, suppose we have three examples and each example has only one feature, A(0.9), B(1.0) and C(1.1). In addition, both A and B are positive, but C is negative. Clearly, the distance between A and C is the same as between B and C, e.g., “0.01 vs 0.01”. Therefore, their distances are not distinctive to build a simple model. However, if we expand the feature set using the above approach, we produce three new instances, A(0.9,0.81,0.64), B(1.0,1.0,0.69) and C(1.1,1.21,0.74). Then the distances of “A vs B” and “B vs C” are different, e.g., “0.049 vs 0.056”, and can now be used to construct predictive models. Based on the feature expansion approach, we obtain new training set with 12 features.

- For test 1, we use the whole dataset as the training data and construct bRDT.
- For test 2, we select training data from the new training set randomly according to the proportion from 20%, 40% to 100%. For each sample, we perform selection 10 times and average the estimated conditional probability.
- For test 3, since the chosen method bRDT is parameter-free, we just report one result of task 1.

The results for test 3 of Task 3 are shown in Table 2.

References


Abstract

Imbalanced data represent significant problem because the corresponding classifier has tendency to ignore patterns which have smaller representation in the training set. We propose to consider large number of balanced training subsets where representatives from the larger pattern are selected randomly. The final decision function represents an average of the component decision functions each of which is based on the particular random set.

1. Introduction

It is a well known fact that for various reasons it may not be possible to theoretically analyze a particular algorithm or to compute its performance in contrast to another. The results of the proper experimental evaluation are very important as these may provide the evidence that a method outperforms alternative approaches.

The method of random sets (RS) was introduced in [2] and may be used in conjunction with different methods. Based on the experimental results we decided to implement LogitBoost algorithm [1] as a component classifier.

2 Boosting Algorithms

Boosting works by sequentially applying a classification algorithm to re-weighted versions of the training data, and then taking a weighted majority vote of the sequence of classifiers thus produced. For many classification algorithms, this simple strategy results in dramatic improvements in performance.

2.1 An Exponential Criterion

The motivation in support of exponential target function is very simple and clear. Let us compare squared and exponential loss functions:

\[ (y_t - u_t)^2; \]  \hspace{1cm} (1a)
\[ \exp \{ -\rho \cdot y_t \cdot u_t \}, \quad \rho > 0. \] \hspace{1cm} (1b)

using two data instances \{1, -1\} and \{1, 4\} where first and second values correspond to the label and decision function. The first example represents a mis-classification, and exponential loss function (1b) detects this misclassification correctly in difference to the squared loss function (1a):

<table>
<thead>
<tr>
<th></th>
<th>{1, -1}</th>
<th>{1, 4}</th>
</tr>
</thead>
<tbody>
<tr>
<td>squared</td>
<td>4</td>
<td>9</td>
</tr>
<tr>
<td>exponential</td>
<td>(e^\rho)</td>
<td>(e^{-4\rho})</td>
</tr>
</tbody>
</table>

We can not optimize step-size in the case of gradient-based optimisation for the exponential target function. Respectively, we will need to maintain low value of the step-size in order to ensure stability of the algorithm. As a consequence, the whole optimization process may be very slow and time-consuming. The target of the following AdaBoost Algorithm is to facilitate optimization process.

2.2 AdaBoost Algorithm

Let \(X = (x_t, y_t), t = 1..m\), be a training sample of observations where \(x_t \in \mathbb{R}^\ell\) is \(\ell\)-dimensional vector of features, and \(y_t\) is binary label: \(y_t \in \{0, 1\}\). Boldface letters denote vector-columns, whose components are labeled using a normal typeface.

In practical situation the label \(y_t\) may be hidden, and the task is to estimate it using vector of features. Let us consider the most simple linear decision function

\[ u_t = u(x_t) = \sum_{j=0}^\ell w_j \cdot x_{tj}. \] \hspace{1cm} (2)

where \(x_{t0}\) is a constant term.

Let us consider minimizing the criterion [1]

\[ \sum_{t=1}^n \xi(x_t, y_t) \cdot e^{-y_t F(x_t)} \] \hspace{1cm} (3)

where

\[ \xi(x_t, y_t) := \exp \{ -y_t F(x_t) \}. \] \hspace{1cm} (4)
We shall assume that initial values of $F(x_t)$ are set to zero. The following Taylor-approximation is valid under assumption that values of $u(x_t)$ are small
\[
\exp \{-y_t u(x_t)\} \approx \frac{1}{2} \left( \left( y_t - u(x_t) \right)^2 + 1 \right). \tag{5}
\]
Therefore, we can apply quadratic-minimisation (QM) model in order to minimize (3). Then, we optimize value of the threshold parameter $\Delta$ for $u_t$, and find corresponding decision rule $f_t \in \{-1, 1\}$. Next, we will return to (3)
\[
\sum_{t=1}^n \xi(x_t, y_t) \cdot e^{-c y_t f(x_t)} \tag{6}
\]
where optimal value of the parameter $c$ may be easily found
\[
c = \frac{1}{2} \log \left\{ \frac{A}{B} \right\}. \tag{7}
\]
where
\[
A = \sum_{y_t = f(x_t)} \xi(x_t, y_t), \quad B = \sum_{y_t \neq f(x_t)} \xi(x_t, y_t).
\]
Finally (for the current boosting iteration), we update function $F$:
\[
F_{new}(x_t) \leftarrow F(x_t) + c \cdot f(x_t), \tag{8}
\]
and recompute weight coefficients $\xi$ according to (4).

**Remark 1** Considering test dataset (labels are not available), we will not be able to optimize value of the threshold parameter $\Delta$. Respectively, we can use either an average (predicted) value of $\Delta$ in order to transform decision function into decision rule, or we can apply direct update:
\[
F_{new}(x_t) \leftarrow F(x_t) + c \cdot u(x_t) \tag{9}
\]
where value of the parameter $c \leq 1$ must be small enough in order to ensure stability of the algorithm.

### 2.3 LogitBoost Algorithm

Let us parameterize the binomial probabilities by
\[
p(x_t) = \frac{e^{2F(x_t)}}{1 + e^{2F(x_t)}}. \tag{10}
\]
The binomial log-likelihood is
\[
y_t^* \log \{p(x_t)\} + (1 - y_t^*) \log \{1 - p(x_t)\} = -\log \{1 + \exp \{-2y_t F(x_t)\}\}. \tag{10}
\]
The following relation is valid
\[
\exp \{-2y_t F(x_t)\} = \xi(x_t) z_t^2 \tag{11}
\]
where
\[
z_t = \frac{y_t^* - p(x_t)}{\xi(x_t)}, \quad \xi(x_t) = p(x_t)(1 - p(x_t)).
\]
We can maximize (10) using method with Newton’s step, which is based on the matrix of second derivatives. As an alternative, we can consider standard weighted QM-model:

$$\sum_{t=1}^{n} \xi_t (z_t - u_t)^2.$$  \hspace{1cm} (12)

After solution \(u(x_t)\) was found, we update function \(p(x_t)\)

$$p(x_t) \left\{ \begin{array}{ll} 1 & \text{if } h_t \geq 1; \\ h_t & \text{if } 0 < h_t < 1; \\ 0 & \text{if } h_t \leq 0 \end{array} \right.$$  \hspace{1cm} (13)

where \(h_t = p(x_t) + \xi_t u(x_t)\). Then, we recompute weight coefficients \(\xi_t\), and return to the minimization criterion (12).

Let us consider update of function \(F\) assuming that \(0 < h_t < 1\). By definition,

$$F_{\text{new}}(x_t) = \frac{1}{2} \log \left\{ \frac{h_t}{1 - h_t} \right\}$$

$$= \frac{1}{2} \log \left\{ \frac{p(x_t)}{1 - p(x_t)} \right\} + \frac{1}{2} \log \left\{ 1 + \frac{u(x_t)}{1 - p(x_t)u(x_t)} \right\}$$

$$\approx F(x_t) + \nu \cdot u(x_t), \quad \nu = 0.5.$$  \hspace{1cm} (14)

**Remark 2** Boosting trick (similar to the well-known kernel trick): as an alternative to QM-solution, we can apply in (9) or (14) decision function, which was produced by another method, for example, Naive Bayes or Decision Trees.

**Remark 3** Approximation (14) coincides with update formula of [1], and is valid under condition that value of \(u(x_t)\) is small enough. Based on the experimental results, we can recommend the following range \(0.1 \leq \nu \leq 0.3\) depending on the particular dataset. Also, it appears to be reasonable [1] to restrict values of \(z_t\) in (12).

## 3 Experiments

The IEEE ICDM 2008 Data Mining Contest was based on real data: detection of a nuclear explosion by the measurements of some important isotopes. The training dataset is strongly imbalanced with 623 positive cases out of total number of 8695. There are 2491 cases in the unlabelled test dataset. The datasets have 4 numerical features named Xe-131m, Xe-133m, Xe-133 and Xe-135. Particular measurements were obtained from 5 stations named V, W, X, Y and Z and, clearly, have very different patterns. Respectively, we transformed available data to the completely numerical set with 9 features where first 5 features are binary and indicate particular station. Also, we replaced labels ”B” by -1 and ”B+E” by 1.

Area under receiver operating curve (AUC) was used as an evaluation criterion. We have found that training simulation results are not very important. For example, using randomForest function in R we can easily produce train-AUC=0.85, but test results were below 0.68 because of the strong overfitting.

Figure 1 illustrates convergence of an average AUC in the case of cross validation with 100 random folds (CV-100). Table 1 shows main test results corresponding to LogitBoost and Decision Trees models in R.

In addition, we considered least-squared linear regression in MATLAB. Note that in the case of linear regression it was necessary to transform original features using logarithmic function. Note, also, that we computed regression coefficients specifically for any particular station.

The final decision function was calculated as an average of 200 component decision functions each of which was based on the balanced subsets with all positive cases and the same number of the randomly selected negative cases. As a component classifier selected LogitBoost model in R with \(\nu = 0.25\) and \(m = 100\) where \(m\) is a number of boosting iterations.

## References


An Instance-based Nearest-neighbor Approach to Classifying Nuclear Explosion Data

D. Oviatt, D. Frandsen, K. Clement, and C. Giraud-Carrier
Brigham Young University
Department of Computer Science
Provo, UT 84602, USA
{xkordave,dallin.frandsen,kendellclement}@gmail.com, cgc@cs.byu.edu

Abstract

In addressing regulation of compliance with the Comprehensive Test Ban Treaty (CTBT), we propose a classification method for detecting explosions based on Xenon isotope levels. Our method involves a small preprocessing step in which we balance the composition of the dataset. After balancing, we run an instance-based nearest-neighbor algorithm to classify the data. We show that the IB1 instance-based clustering algorithm with dataset composition balancing is an effective, accurate, and robust method for classifying nuclear explosions based on Xenon isotope levels.

1. Introduction

The Comprehensive Test Ban Treaty (CTBT) is the most comprehensive ban on nuclear activities, banning all nuclear explosions in all environments for any purpose whatever. Although 180 countries have already signed the treaty, 9 are still expected to sign before the associated ban may be enforced. When these countries have signed, only a short delay (180 days) will be granted until the treaty comes into full force. At such a time, as with any other treaty or law, monitoring will become necessary to enforce the ban.

Several remote sensing stations have already been deployed, whose purpose is to monitor the atmosphere for traces of Xenon isotopes indicative of nuclear explosions. Clearly, the more automatic and accurate this monitoring, and in particular the detection of illicit nuclear activity, the more effective the enforcement of the ban. To kick-start the development of automatic detection systems, Health Canada has created a dataset of both explosion and background (i.e., no explosion) observations at the various stations, and issued a challenge to the data mining community to come up with an accurate classifier over all stations (task 1), to identify an optimal classifier for each station (task 2), and finally to submit results on an unseen set of test observations for independent evaluation (task 3).

This paper describes our methodology and our results for this challenge. It is loosely organized around a standard data mining process: section 2 describes the analysis of the dataset and preprocessing tasks we applied to prepare the data for further analysis; section 3 focuses on model building and the algorithm we selected for that purpose; section 4 presents and discusses our results; finally, section 5 concludes the paper.

2. Data Preparation and Preprocessing

As we initially examined the composition of the datasets, it became obvious that it would be nearly impossible to correctly classify every single instance. Several instances have different labels, yet come from the same location, with the same Xenon measurements. Also, both the training and test datasets are rather skewed, with almost all samples showing traces of Xenon-133, and relatively few traces of the other Xenon isotopes, especially in locations V, W, X and Y. After plotting comparisons of isotopes, we noted that there was no clear clustering of location or classification type.

In an attempt to enhance our classification results by incorporating outside information, we researched the issue of Xenon isotope concentrations and their relation to explosion levels. We confirmed that Xenon-131m, Xenon-133, Xenon-133m, and Xenon-135 are the fission products of Uranium and Plutonium produced by neutron irradiation, and evidence of nuclear activity [2]. Because all four Xenon isotopes are indicative of explosions, we decided against dropping any of the four
isotope levels from our analysis.

Initially, we experimented with several clustering, association, and grouping techniques. We quickly discovered that all methods classified every instance as Background and Explosion (B + E), hence being completely unable to discriminate explosions from normal background (B) observations. Indeed, the dataset exhibits acute class imbalance. The ratio of Background signals to Background and Explosion signals is 623 to 8,072. In other words, the data contains 92.8% B + E observations and only 7.2% B observations. Classifying all instances as B + E thus yields a default accuracy of 92.8%.

In order to avoid the problem of naively classifying all instances as B + E, we explored several solutions. The most effective way to diversify the classifications was to modify the composition of the training dataset immediately before running our algorithm. To create the dataset to train our model, we kept all B + E instances from the original training dataset (8,072 instances), and sampled the 623 B instances with replacement until the number of B instances equaled the number of B + E instances (8,072 instances). This preprocessing technique allowed us to achieve higher accuracy results on the training data, and to produce more diverse results for the test data.

We classify instances from a specific location by training the algorithm on a balanced dataset composed of an equal number of B and B + E training instances from the same location. For tasks requiring the classification of the entire dataset, we do not modify instance values based on location. In addition, we made no adjustments based on location or to reduce noise.

3. Model Building

After experimenting with several classification algorithms (e.g., Bayesian Neural Networks, Decision Trees, Clustering), we selected IB1, an instance-based classification algorithm, to build our predictive model. The algorithm is found in the standard Weka package [3], and was originally introduced in [1].

IB1 finds the training instance that is closest, according to Euclidean distance, to the new instance to classify, and returns the label of that training instance as the predicted label of the test instance. When multiple instances within the training set are found to have identical, minimal distance to the new instance, the arbitrary decision is made to use the label of the first instance found to classify the new instance.

The IB1 algorithm is essentially a 1-nearest-neighbor classification algorithm, with a few differences. The most significant of these differences is the normalization of the ranges of the attributes prior to computing the distance. This prevents the classification from being biased towards or against a specific attribute. Other differences deal mainly with how the data is processed, and seem to have minimal impact on classification results.

Because IB1 produced very accurate classification, IBk, with k > 1 seemed like it would produce an improvement, as it could provide probabilities for the different possible classifications, as opposed to the IB1 method, which only classifies a new instance as 100% in a single class (or not). This is because with k > 1, the algorithm introduces a voting mechanism, so the nearest k neighbors give different probabilities of what the new instance should be classified as. Additionally, the (inverse of the) distances to the k-nearest instances in the training dataset may be used to weight each classification vote. We tested IBk for k values of 2, 5, 10, and 20. In none of these situations did IBk outperform IB1. A possible explanation for this is that the data instances of one classification type are sparsely distributed. As a result, adding more neighbors to decide the classification of the new instance only adds votes for an incorrect classification. Using multiple neighbors to decide a new classification in this kind of data might still be beneficial in the case of two or more equidistant nearest-neighbors.

4 Results

In this section, we report the results of our selected approach on the training and test data, as well as a brief analysis of the model’s efficiency.

4.1 Training Data

Considering that the goal of the model is to accurately predict explosion incidents (i.e., B + E observations), we may view a correct classification of explosion as a hit, the classification of an explosion as background as a miss, the classification of background as an explosion as a false alarm, and the correct classification of background as a correct rejection. This is captured graphically in the generic confusion matrix depicted in Figure 1.

In order to evaluate our algorithm, we compare it to the default model that always predicts B + E. When using that algorithm to classify the training dataset, 92.8% of the data is classified correctly. The corresponding confusion matrix is shown in Figure 2. Although accuracy is rather high and the model produces 100% hits, it also produces a false alarm rate of 100%, which is clearly unacceptable.
While we are not experts in this domain, it seems that any enforcing agency—we use the term loosely—would have to react somehow to all cases of detection of explosion. When the detection is correct, this is fine; however, false alarms may trigger inadequate actions that may even lead to diplomatic incidents. For example, the innocent party may take offense at being wrongly accused—an unpleasant situation—, or enforcing parties may impose unwarranted sanctions against the alleged wrongdoer—a potentially more dangerous situation. On the other hand, if the system misses the detection of an explosion, then guilty parties would be able to pursue nuclear testing in spite of the ban. Although undetected, this situation would create inequity (with respect to nuclear arms for example) among nations, which goes against some of the philosophy and intent of the treaty. It is clear from this that the cost associated with false alarms and the cost associated with misses are not negligible and rather different from one another.

From the foregoing consideration, it is clear that any viable system should not simply maximize accuracy, but more importantly minimize false alarms and missed detections. This was our emphasis in building our predictive model.

When applying our nearest-neighbor classification algorithm to the training data, we achieved 92.3% accuracy, which is about the same as the default accuracy. However, the distribution of errors is very different as shown in the confusion matrix in Figure 3.

As stated above, the accuracy of our model is marginally lower than that of naively classifying all training instances as $B + E$. When taking into account the costs involved with predicting incorrectly, this slight drop-off in accuracy is actually a welcome trade-off. Our model reduces the false alarm rate from 100% to only 45%. Although this remains rather high, it is a significant improvement over the default. Furthermore, this improvement comes at a very small cost in terms of misses, with only 5% of actual explosions not being correctly detected by the model.

### 4.2 Model Efficiency

We used different percentages of the training data to train the model in order to test the efficiency of data use. To measure this, we randomly selected 20% of the training dataset and used that 20% as a hold-out, or validation, dataset, $V_S$. We used the remaining 80% of the training dataset, $T_S$, to draw training data for our model.

We used various fractions of $T_S$ to train the model, and for each one recorded the accuracy on $V_S$. The fractions considered are 20%, 40%, 60%, 80%, and 100%. Figure 4 is the resulting learning curve, which shows how the percentage of correctly-classified instances of $V_S$ changes depending on the percentage of $T_S$ used to develop the model.

It is evident that the percentages of correctly-classified instances increases only very little as the amount of training data increases. With only 20% of the training data, 88% of the data was correctly classified. The accuracy increases about 1% when 100% of the training data is used. In other words, we get roughly the same accuracy results from 20% of the training data as we get from 100% of the training data. This shows that our algorithm learns a great deal of information from a relatively little amount of data. More data does not significantly improve the results.
5 Conclusion

Detecting violations of the CTBT is a problem that should not be taken lightly. Consideration must be given not only to actions that will be taken based on correctly identified CTBT violations, but also to the consequences of falsely implicating CTBT compliant parties due to incorrect classification of data.

Our results show that the IB1 algorithm is an effective, accurate, and robust method for differentiating background radiation from nuclear explosion-caused radiation, based only on the concentration levels of four Xenon isotopes. Our model significantly reduces the number of false alarms while retaining most of the system’s overall accuracy. Based on these results on the training data, we are confident that our results on the test data will show similar quality.

References

A GA based Weighted Sampling Support Vector Machine (WSSVM)

Jianfei Wu*, Omar Elariss, Anne Denton, Dianxiang Xu
Department of Computer Science
North Dakota state university
Fargo, ND, 58105, U.S.A.
{jianfei.wu, omar.elariss, anne.denton, dianxiang.xu}@ndsu.edu

Christianson David, Stafslien Shane, Daniels Justin, Webster Dean
Center for Nanoscale Science & Engineering
North Dakota State University
Fargo, ND, 58105, U.S.A.
{christianson.a.david, stafslien.shane, daniels.justin, webster.dean}@ndsu.edu

Abstract

We present a system for verifying whether the Comprehensive Nuclear Test Ban Treaty (CTBT) is met or not. The evaluation is done based on the ICDM 2008 tasks two and three. The classification itself is performed by a Support Vector Machine (SVM) classifier. A special sampling technique is used to address the asymmetry of the data set. Normalization is used to address the problem of missing information due to the detection of radioxenon at International Monitoring System (IMS) stations that are far away from the source of explosion. Finally, genetic algorithm is used to optimize the parameters of the classifier.

1. Introduction

The CTBT compliance by nations worldwide is a crucial step for nuclear disarmament. As a result, a verification regime was designed by CTBT to monitor and detect any nuclear testing activities [3]. This regime is comprised of a network of international sensors (ISM) for the purpose of nuclear event detection. From a data mining perspective, the detection process is a classification problem of a radioactive noble gas concentration. Our paper addresses the classification task of identifying whether the noble gas isotopes concentration is a result of a Background (B) source or a background plus explosion (B+E) source.

The structure of the classifier is shown in Fig. 1. The data set is composed of four attributes, namely Xe-131m, Xe-133m, Xe-133, and Xe-135. In the first step, the subset of data for a particular station is extracted and used for the training and testing of the classifier. Then, we use the genetic algorithm to iteratively train the model. In each iteration, the data set is first separated to n (where n is either 3 or 5) subsets based on the size of the ’B’-labeled subset. Because of the extremely small number of ’B’-labeled instances, we introduce a constraint such that each subset has the same number of ’B’-labeled instances. We also train a traditional SVM model in order to compare its performance with the WSSVM classification model. In each iteration, an n-fold cross validation was performed on both models. After the end of an iteration, the fitness value is used as an input to the next iteration in order to optimize both models. Note the models are trained in different machines.

Figure 1. The structure of the classifier model
The proposed algorithm has several steps including Ave-normalization, Weighted sampling SVM classification, and parameter optimization.

2.1. Ave-normalization

In many scientific experiments, the data can not be properly measured due to the fact that scientific equipments may not be sensitive enough to correctly measure week experiment responses. In this case, an attribute value ‘0’ is usually substituted. Normalization as a preprocessing step for a multi-attribute data set, is basically used to balance the effect of each attribute such that no attribute will severely dominate the classification procedure.

Instead of using a Z-score normalization [5], we choose to use a new normalization approach which is entitled as ‘Ave-normalization’:

\[ V_i' = \frac{V_i}{\text{mean}(V)} \tag{1} \]

\( V_i \) is an original attribute value, \( \text{mean}(V) \) is the average for this attribute, and \( V_i' \) is the normalized attribute value for \( V_i \). The intuition behind Ave-normalization is that the 0’s among different attributes should not have any distance since they may represent some ‘un-measurement’ responses. In addition, all the attributes are equally weighted by normalizing the averages to 1. These changes slightly affect the standard deviation for each attribute.

Fig. 2 compares the AUC value for the training data set of each station and the combined whole data set, by using two different normalization methods followed by the same classification procedure as shown in Fig. 1.

2.2. Weighted sampling SVM classification

The problem of class imbalance has been already addressed by several papers [1, 2]. In the training data set, instances labeled with ‘B+E’ are much more than those with ‘B’. Building a classifier model on the whole data set (either the data set of one station or all the stations) not only decreases the accuracy of the model, but worsens the performance. In our approach, we first extract a subset of ‘B+E’-labeled instances which are neighbors of those ‘B’-labeled instances. The procedure is straightforward: for all ‘B’-labeled instances, find their K-nearest ‘B+E’-labeled neighbors. \( K \) equals to \( \text{ceil}(\frac{N_{B+E}}{N_B}) \), where \( N_{B+E} \) and \( N_B \) are the size of ‘B+E’-labeled set \( S_{B+E} \) and ‘B’-labeled set \( S_B \) respectively. We refer to the new ‘B+E’-labeled set as \( S_{B+E}' \):

\[ S_{B+E}' = \bigcup_{i=0}^{N_B} \sigma_i(S_{B+E}, K) \tag{2} \]

\( \sigma \) is a K-nearest neighbor selector and \( S_{B+E}' \) is the union of all these neighbors.

Fig. 3 shows an example of this selection. Building a model that focuses on instances among the two sets, \( S_{B+E}' \) and \( S_B \), manifests the boundary, and the performance can be greatly improved as well.

Further more, before training the classifier and to avoid overfitting, we randomly sample set \( S_{B+E}' \) \( T \) times (\( T \) equals to 5 in all our experiments), where each sample size equals to \( N_B \), denoted by \( s_j \) (\( j \in [1, T] \)). So for each data set that combines \( s_j 'B+E' \) set and the ‘B’ set \( S_B \), we run the SVM on a seperate test set with the same SVM parameters, \( C \) and gamma. The predicted labels and probabilities
(the certainty of predicted labels) for the test set is modeled from the SVM classification results as follows:

\[
\begin{align*}
\text{Label}_i &= H\left(\sum_{j=1}^{T} (\text{label}_{ij})\right) \\
\text{posib}_i &= \sum_{j=1}^{T} (\text{posib}_{ij} + k \times \text{CohenDT}_j) \\
\text{posib}_i' &= \frac{\text{posib_i} - \text{min}(\text{posib})}{\text{max}(\text{posib}) - \text{min}(\text{posib})}
\end{align*}
\] (3)

\[\text{CohenDT}_j = \frac{\left|\text{mean}(s_j) - \text{mean}(S_B)\right|}{\sqrt{\text{std}(s_j)^2 + \text{std}(S_B)^2}}\] (4)

Notice that the denominator is an approximation for the denominator of Cohen’s D since \(s_j\) and \(S_B\) have the same size. Also, the numerator is the absolute value of the numerator of Cohens D. The idea behind this is: In classifying the testing data set with multiple SVM runs, we also consider how ‘good’, i.e., how far between ‘B’ set \(S_B\) and each ‘B + E’ set \(s_j\), is the data inputted to each SVM run. This is done to evaluate each classification result and calculate the final results by weighting them using equation 3.

### 2.3. Parameter optimization

We use the genetic algorithm to optimize both parameters C and gamma for SVM, as well as \(k\) and \(e\) for the weighted sampling SVM model. Because the number of ‘B’-labeled instances in the training data for each station varies a lot, and also because we try to build the model using different percentages of the training data set (task 3), i.e., 20%, 40%, 60%, 80% and 100%, the number of fold we use in cross validation varies from station to station, and also varies in different percentage subsets. For all stations, we set the population and the generation size to 1000, and we assign the values of the fitness functions to the AUC values.

### 3. Implementation and Results

The code was implemented using MATLAB, while the experiments were carried out on computers ranging from 3.4GHz Pentium 4 processors with 3GB RAM to Xeon processor with 8 GB RAM. The SVM with a Radial Basis Function (RBF) kernel implementation [4] was used. In task 2, the WSSVM model was tested on each station using the respective parameters for each station. Then the AUC value for each station was calculated based on \(\text{Label}_i\) and \(\text{posib}_i\), as in equation (3). Further more, the AUC value for the whole data set was calculated based on \(\text{Label}_i\) and \(\text{posib}_i\) (by concactinating the two values of all stations together). Notice \(\text{posib}_i\) of each station has a range of [0, 1], this allows the concatenation of the values for all the stations together during the process of calculating the AUC value for the whole data set.

Fig. 4 shows the AUC values where different percentages of data are used in order to build the model (the rest of the data is used to test the model in order to calculate the AUC values). From Fig. 4 we can see that the AUC values for all stations basically have an increasing trend as the percentage of data used for training increases. One exception is at 60%, where station W has a very sharp decrease in the AUC value. We tried to improve on these values through

### Figure 4. AUC values & Percentage of data used in building the model, using provided software

![Figure 4. AUC values & Percentage of data used in building the model, using provided software](image)

### Figure 5. AUC values & Percentage of data used in building the model, using our own software

![Figure 5. AUC values & Percentage of data used in building the model, using our own software](image)
training the model multiple times in order to get different parameters. Also we tried to test the model on different testing sets. No noticeable improvements were detected. Fig. 5 shows that using our AUC calculation software, the AUC value for station W is not that low.

Fig. 6 compares the result between the proposed approach and the traditional SVM classifier, by averaging their AUC values of all percentage subsets. The WSSVM results show slight improvement on the SVM when sampling is taken into consideration. Additional improvements are shown when the proposed approach is compared to the SVM on the whole data set.

In task 3, the WSSVM model was tested on the unlabelled data set using parameters trained from different percentage subsets of the training data set. In addition, the parameter sensitivity was tested on the WSSVM model trained with 40% of training data set. The corresponding AUC values are summarized in Table 1.

From Table 1, we can see the fluctuation of parameters do not significantly effect the AUC values.

4. Conclusions

In this paper we have proposed a classifying system that identifies the violations of CTBT. The classifier differentiates between (B) and (B+E) observations based on radionuclide concentration. The use of the Ave-normalization and the WSSVM model showed improvement on the regular SVM classifier. Further improvement can be done on the weighted sampling, this can be seen through the low AUC values of station Z, by not only randomly sampling the boundary instances but considering the labels topology.

References

Abstract

In this paper, we treat the ICDM contest 2008 tasks as imbalanced classification tasks. An ensemble under-sampling technique which was used for web spam detection [1] is adopted to deal with the class-imbalance problem. Decision stump, a weak one-level binary decision tree, is chosen as the weak classifier. A popularly used classifier combination technique, Adaboost, is used to ensemble the weak classifiers.

1. Introduction

Classification is the process of predicting group membership for data instances. It has been widely studied in pattern recognition, machine learning and data mining, etc. Many classifiers have been presented. In our participation in the ICDM contest 2008, we have not deviated from the classification paradigm. Popular classification techniques include decision trees, neural network and support vector machines (SVM), etc. We use the implementation of decision stump given in Weka[4] as the base classifier. Decision stump is a weak classifier consisting of a one-level binary decision tree with categorical or numerical class label. It can be used as components in ensemble learning techniques.

The class imbalance problem refers to the issue that a dataset is dominated by a class or classes that have significantly more samples than other classes. It has been observed in a variety of domains, such as economic, medical, text classification, etc. “When classes are imbalanced, existing learning methods often produce classifiers that do little more than predict the most common classes”. It is recognized as a crucial problem in machine learning and data mining, because it will hinder the performance of regular classifiers. As shown below, the training set for ICDM contest 2008 is a typically class-imbalance problem.

Under-sampling technique has been popularly used in class-imbalance learning. It uses only a subset of the major class examples for training the classifier, alleviating the class-imbalance distribution. However, by under-sampling major class examples, potentially useful information contained in the ignored examples would be neglected. In [1], Geng etc proposed an ensemble strategy to overcome the deficiency and keep the efficiency of under-sampling, which was used for web spam detection. In this paper, we adopted the ensemble technique for the class-imbalance problem in this contest.

Boosting is a re-sampling technique that has been proven effective when used in conjunction with a wide range of techniques for training classifiers. In boosting, we first create a classifier on the training set with accuracy greater than average, and then add new component classifiers to form an ensemble whose joint decision rule has arbitrarily high accuracy on the training set. In general, it creates successive component classifiers with a subset of the training data that is “most informative” given the current set of component classifiers. Classification of a test point X is based on the output of the component classifiers.

For this contest, we used adaboost as meta classifier, which is a most popularly studied boosting technique. In Adaboost, the weak classification is iteratively added until an iteration threshold is met. Each training sample receives a weight indicating the probability of being selected as training data. At each iteration, the weight of each misclassified sample is increased, so that the new classifier would focus more on the misclassified samples.

2. Task Description

The ICDM 2008 contest is about developing data mining technologies to detect nuclear explosions. Compliance verification of CTBT (Comprehensive Nuclear-Test-Ban Treaty) employs the remote detection and measurement of radioxenon (radioactive forms of a noble gas, xenon) for detecting nuclear explosions. The problem aims to distinguish between those radioxenon measurements that are due purely to normal environmental emissions or background (B) from those measurements that contain the signature of an explosion combined background (B+E).

Data Description

The data used to train the model for the explosion detection task is collected from five CTBTO (Comprehensive Nuclear-Test-Ban Treaty Organization) monitoring sites. The data sets are composed of two classes, B (Background) and B+E (Background plus Explosion). A few hundred to a few thousand cases of observations are provided from each site. Each case is represented by an alpha numeric index. The numeric portion traces the background...
measurement or background measurement combined with synthesized explosion observation used to create the datum. The alpha portion of the code refers to one of five real-world measurement sites that have been collecting measurements of radioxenon concentrations daily for an extended period of time and to the qualitative degree of complexity of the background radioxenon observed at these sites. The five CTBTO sites are represented by V, W, X, Y, Z respectively.

Totally there are 8695 samples provided for training the classifiers, in which 623 cases are labeled as B (background) and 8073 cases are labeled as B+E (background plus explosion). The distribution of samples from each site is shown in TABLE 1.

As is shown in TABLE 1, the samples distribution of each class is severely imbalanced. The ratio between class B and class B+E varies from about 1:10 (site X) to about 1:50 (site W). So distinguishing background (B) from explosion combined background (B+E) is a typically imbalanced classification problem.

TABLE 1. Sample distribution of different sites

<table>
<thead>
<tr>
<th></th>
<th>V</th>
<th>W</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>115</td>
<td>14</td>
<td>210</td>
<td>40</td>
<td>244</td>
</tr>
<tr>
<td>B+E</td>
<td>1589</td>
<td>763</td>
<td>2090</td>
<td>1169</td>
<td>2461</td>
</tr>
</tbody>
</table>

Four radioxenon isotopes, Xe-131m, Xe-133m, Xe-133, Xe-135, are measured in a radionuclide monitoring procedure for each sample. That is, each sample from both training dataset and test dataset is composed of four features, each feature representing the activity concentration of one radioxenon isotope. We directly used these features for training and testing, no special treatment is conducted.

3. The learning algorithm

As discussed in section 2, the classification between Background and Background plus explosion is a typically imbalanced class distribution problem. Thus, we adopt imbalanced classifiers for this ICDM contest.

Techniques to deal with class imbalance problem include cost sensitive learning and sampling strategies. Among these strategies, under-sampling technique has been shown to be effective and efficient. However, as it only chooses a subset of the over represented class for training classifiers, it may discard potentially useful major class samples, thus failed to make full use of all the helpful information in the training data. For web spam detection tasks, [1] presented an ensemble under-sampling (ERUS) strategy, which iteratively undersampled the training dataset. By such iteration, almost all the major samples could have the chance of being trained, therefore better performance is achieved. In this paper, we adopt similar strategy for nuclear explosions detection.

The framework of the ensemble under-sampling classification procedure for this contest is presented in Fig. 1. Firstly, the training dataset is randomly undersampled N times with different seeds (N is tunable, in the contest, we set N to 11), generating N different training sets which are relatively balanced distributed. Secondly, N classifiers are generated by learning on the N different training sets. Each classifier is trained by adaboost algorithm with decision stump chose to be the weak classifiers. Thirdly, the N classifiers gathered last step are combined together, by linearly averaging their outputs. Then, the performance of the combined classifier is evaluated on the test data set.

4. Experimental Results

In this section, we report the experimental results of the classification framework presented last section. The figures of merit are computed by the software released by the organizer. Many figures are computed. In this paper, we report AUC, F1-measure and MCC only.

There are totally three tasks in this year’s contest. Task 1 is to develop a classifier over the entire set of stations, parameters for each station may be tuned separately. Task 2 is to identify an optimal algorithm for each station. Task 3 is a testing task, evaluating various aspects of performance of classifiers developed for Task 1 and Task 2 on a separate testing set.

For Task 2, we train a classifier for each station under the framework presented in section 3. For the samples from each station in the training file, we randomly choose 80% as training set and the remaining 20% as testing set. The training set is undersampled by ERUS and an adaboost classifier is trained.
with decision stump chosen as weak classifier. The test performances of classifiers for five stations are shown in Table 2.

For Task 1, we simply combined the five classifiers trained for Task 2 together, and for the test samples, first determine their stations, and corresponding classifiers are chosen. The overall performance is also shown in TABLE 2.

Figure 2 shows the ROC curve of the classifiers trained for Task 1 and Task 2. In this graph, Task 1 represents the ROC curve for Task 1. StnV represents the ROC curve for station V in Task 2, and so on.

TABLE 2. Experimental Result for Task 1 and Task 2

<table>
<thead>
<tr>
<th></th>
<th>AUC</th>
<th>F1-measure</th>
<th>MCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1</td>
<td>0.6968</td>
<td>0.1937</td>
<td>0.1199</td>
</tr>
<tr>
<td>V</td>
<td>0.8417</td>
<td>0.3165</td>
<td>0.3499</td>
</tr>
<tr>
<td>W</td>
<td>0.8824</td>
<td>0.1116</td>
<td>0.1642</td>
</tr>
<tr>
<td>X</td>
<td>0.7088</td>
<td>0.2278</td>
<td>0.156</td>
</tr>
<tr>
<td>Y</td>
<td>0.8048</td>
<td>0.1463</td>
<td>0.1286</td>
</tr>
<tr>
<td>Z</td>
<td>0.6395</td>
<td>0.1017</td>
<td>0.1001</td>
</tr>
</tbody>
</table>

For Task 3, when evaluating the classifiers developed for Task 1 and Task 2, as an independently testing file is provided, we used all the samples in the training file for training classifiers. However, as we don’t have the labels for the test file, the performances for Task 3 are not reported here.

References


