Exploiting Contexts to Deal with Uncertainty in Classification

Bianca Zadrozny
Fluminense Fed. Univ.
Computer Science Dep.
Niterói,Brazil
bianca@ic.uff.br

Gisele L. Pappa
Fed. Univ. of Minas Gerais
Computer Science Dep.
Belo Horizonte, Brazil
glpappa@dcc.ufmg.br

Wagner Meira Jr.
Fed. Univ. of Minas Gerais
Computer Science Dep.
Belo Horizonte, Brazil
meira@dcc.ufmg.br

Marcos André Gonçalves
Fed. Univ. of Minas Gerais
Computer Science Dep.
Belo Horizonte, Brazil
mgoncalv@dcc.ufmg.br

Leonardo Rocha
Fed. Univ. São João Del Rei
Computer Science Dep.
São João Del Rei, Brazil
lcrocha@dcc.ufmg.br

Thiago Salles
Fed. Univ. of Minas Gerais
Computer Science Dep.
Belo Horizonte, Brazil
tsalles@dcc.ufmg.br

ABSTRACT

Uncertainty is often inherent to data and still there are just a few data mining algorithms that handle it. In this paper we focus on how to account for uncertainty in classification algorithms, in particular when data attributes should not be considered completely truthful for classifying a given sample. Our starting point is that each piece of data comes from a potentially different context and, by estimating context probabilities of an unknown sample, we may derive a weight that quantifies their influence. We propose a lazy classification strategy that incorporates the uncertainty into both the training and usage of classifiers. We also propose uK-NN, an extension of the traditional K-NN that implements our approach. Finally, we illustrate uK-NN, which is currently being evaluated experimentally, using a document classification toy example.

Categories and Subject Descriptors

I.2.6 [**Learning**]: Induction; H.2.8 [**Database Applications**]: Data Mining

Keywords

1. INTRODUCTION

One of the first assumptions we make when using data mining algorithms is that the data reflects the reality in a very accurate way. We know noise exists, but in most cases we have a lot of confidence on data. However, uncertainty is often inherent to data, and can be caused by a variety of factors. On one hand, it may be a product of imprecise measurements of a data source. For instance, data generated from sensor networks may be prone to sensor errors. Similarly, data inputs coming from (electronic) surveys can be incomplete, and often have disguised missing data (i.e. data dismissed by the user is treated as another valid value) [8].

Knowing that the data source can be responsible for data uncertainty, we still need to consider that data may be collected from *multiple sources* and/or over long periods of *time*. In this case, it is likely that the underlying process that generates the data is different for each of the sources or changes over time. This difference can lead to precision loss if we try, for example, to use data from one source to make predictions about a different source. We may face similar problems if we use data from the past, which may have become obsolete, to make predictions about the present.

In the past decade, a great deal of research has focused on how to measure data uncertainty (usually based on probabilities), and then to develop appropriate databases for storing and managing this data [2]. However, when it comes to analyzing and mining uncertain data, we still face a big challenge. Considering the data mining tasks of clustering, association and classification, clustering is certainly the one where more progress has been made. We can find enhanced versions of existing clustering methods, such as the UK-means [4], and the hierarchical density-based clustering proposed in [9].

There are also some efforts for classification (discussed in Section 2) and we focus on the same problem in this paper, more specifically on how to account for uncertainty while building and using classifiers. A classifier is seen as a function that maps the occurrence of attributes to a class. We consider that uncertainty can be related to different data dimensions, such as time, space or source. Without loss of generality, we will illustrate it using the scenario of text classification, where the attributes are terms and the classes are document categories. In this scenario we may distinguish at least two sources of uncertainty regarding attributes. The first source is the time when the documents being used for building the classifier were collected. We may see a second source of uncertainty by considering that these texts compose a digital library. In this case, the uncertainty may come from the reputation (or impact) of the venue or the author.

In this paper, we focus on uncertainty regarding *time*, but the concepts introduced here may also be applied to other

data dimensions. In the context of text classification, documents are produced over long periods of time. We call each of these points in time (e.g., a day or a year) a *context*[7]. We assume that we know the context where each example was collected, but we do not know whether the underlying process that generates the data changes from one context to the other. Therefore, when we try to classify an example from one context, the examples from the other contexts are considered uncertain, since the data distribution in each of the contexts may be different.

Note that our concept of uncertainty is *relative*, i.e., it depends on the context of the example we are trying to classify, which we call the *target context*. This relative uncertainty is given by the probability that an example from a different context was generated from the same underlying distribution as the data in the target context.

Based on this formulation, we propose a lazy classification strategy that estimates and incorporates these relative uncertainties into the classification process.

2. RELATED WORK

A variety of methods for representing, storing, querying and managing data uncertainty have been created in the past decade [2]. When mining uncertain data, modifications of some of the most well-know algorithms of clustering [4], association [5] and classification[11] were already proposed.

The few methods previously introduced to deal with classification in uncertain data have a strong dependence on data representation. Some of these works assume that uncertainty was already estimated, and is stored in a probabilistic database or an appropriated model [11]. Other works first estimate uncertainty, and then add it to the classification algorithm being created. [11], for instance, adapted the rule induction algorithm Ripper to deal with uncertain attributes. [1] proposed a general framework for mining data uncertainty grounded on density-based transforms, and instantiated it to the classification task. [3] proposed a Total Support Vector Classification (TSVC) approach that considers the inputs were corrupted with noise. The TSVC takes advantage of this information to compute the degree of separation between two classes.

As observed, the few classification works dealing with data uncertainty propose specific solutions for specific algorithms. In this work, in contrast, we introduce a general approach that can be used to estimate uncertainty using different existing algorithms, and then apply this uncertainty to modify a variety of classification algorithms. The methods to modify current algorithms and estimate the probabilities are described in the next sections.

3. APPLYING CONTEXT PROBABILITIES TO CLASSIFICATION

Assume we have a training set T consisting of examples of the form (x, y, c), each drawn independently from a distribution D with domain $\mathcal{X} \times \mathcal{Y}$ where $\mathcal{X}$ is the attribute space, $\mathcal{Y}$ is a discrete class label space and $\mathcal{C}$ is a discrete context label space. Note that the context should reflect the aspects that are known to introduce uncertainty in the relationship between the attributes and classes in the domain under consideration. Examples of such aspects are time (e.g., year of publication of a document), source (e.g., the journal where a paper is published) or a combination of both.

We would like to classify a test example (x_t, y_t, c_t), belonging to a target context c_t. An obvious approach would be to select only the training examples (x, y, c) for which $c = c_t$ and use a standard classification algorithm. In this case, we would be certain that the training data is correct. However, we would be missing the opportunity of using examples from contexts which are similar to the target context. This can be specially harmful in situations where we have many contexts and only a few training examples belonging to each context. On the other extreme, we could choose to ignore the context information and use the whole training set for classification. However, if the contexts are very dissimilar this can lead to poor classification performance, since the classification will be based on incorrect data relative to the target context.

Next we will show that it is possible to use all the training examples, as long as we weight them by the probability that they belong to the same context as the example we are trying to classify.

Standard classifier learners try to find a classifier h to minimize the expected value of a loss function over the distribution of examples given by

$$E_{x,y\sim D}[l(h(x), y)].$$

The loss function is, in many cases, given by an indicator of error $I(h(x) \neq y)$, but we make the analysis more general by considering an arbitrary loss function.

Here, we would like the classifier learner to minimize the expected value of the loss only for test examples that belong to a target context c_t. Thus we would like it to minimize

$$E_{x,y,c\sim D}[l(h(x), y)|c = c_t].$$

Let D_{c_t} be a new distribution such that

$$D_{c_t}(x, y, c) \equiv P(c = c_t|x, y)\frac{D(x, y)}{P(c = c_t)}.$$

The following theorem shows that if we change the distribution of examples from D to D_{c_t}, by weighting each example by the ratio $P(c = c_t|x, y)/P(c = c_t)$, we will minimize the expected error for the desired target context c_t.

THEOREM 3.1. *For all distributions, D, for all classifiers, h, for any loss function $l = l(h(x), y)$, if we assume that $P(c = c_t) > 0$ then*

$$E_{x,y,c\sim D_{c_t}}[l(h(x), y)] = E_{x,y\sim D}[l(h(x), y)|c = c_t]$$

PROOF.

$$\begin{aligned} &E_{x,y,c\sim D_{c_t}}[l(h(x), y)] \\ &= \textstyle\sum_{x,y} l(h(x), y)\frac{P(c=c_t|x,y)}{P(c=c_t)}P(x, y) \\ &= \textstyle\sum_{x,y} l(h(x), y)P(x, y|c = c_t) \\ &= E_{x,y\sim D}[l(h(x), y)|c = c_t] \end{aligned}$$

□

Therefore, to obtain a lazy classifier that uses all the available training data, while taking the context information into account, we can use any standard lazy learning method (such as K-NN), giving an importance weight to each example (x, y) that is proportional to $P(c = c_t|x, y)/P(c = c_t)$. As discussed in [12], for most classifiers there are standard ways to incorporate importance weights. When this is not possible, we can use sampling to create a training set that obeys the distribution given by the weights. The sampling can be performed in a lazy fashion, i.e., for each test example that

arrives we can do the sampling using the weights relative to its context c_t.

We note that the ratio $P(c = c_t|x, y)/P(c = c_t)$ reflects the likelihood that a given example with attributes x and class y is generated from context c_t. Thus, training examples from contexts that are more similar to the target context c_t will have higher weights, while examples from contexts that are less similar to the target context will have lower weights. If the context corresponds to time, we would expect that examples closer in time to the test example would get higher weights, since the distribution of attributes and classes is likely to change smoothly from year to year.

4. ESTIMATING CONTEXT PROBABILITIES

Up to now, we have assumed that we know the context probabilities $P(c = c_t|x, y)$ and $P(c = c_t)$, for each context c_t and example (x, y), so that we can calculate weights for each training example when classifying a test example from context c_t.

In practice, however, the context probabilities are not known but we can estimate them using the training data. The value of $P(c = c_t)$ for each context c_t can be estimated as the proportion of training examples belonging to c_t. More concretely, if we have N training examples of the form $(x_i, y_i, c_i), i = 1 \ldots N$, the value of $P(c = c_t)$ can be estimated as

$$\hat{P}(c = c_t) = \frac{\sum_{i=1}^{N} I(c_i = c_t)}{N},$$

where $I(\cdot)$ is an indicator function that returns 1 if its argument is true and 0 otherwise.

The probabilities $P(c = c_t|x, y)$ can also be estimated from the training data. In this case, however, the estimation is not as trivial as in the case of $P(c = c_t)$, since we must obtain a function of (x, y), considering that x is an arbitrary attribute space. This can be done by using a classifier that outputs class membership probability estimates. If we have N training examples of the form $(x_i, y_i, c_i), i = 1 \ldots N$, we can simply feed them to a classifier learning method using (x_i, y_i) as the attribute vector and c_i as the class. The classifier obtained in this fashion will be able to output context probabilities for any pair (x, y) and target context c_t. As long as we have enough training data for each target context, we should be able to predict accurate probabilities using an appropriate classifier learning method such as boosted trees, random forests or SVMs [10]. Note that this step is not done in a lazy fashion, but as a pre-processing step to obtain weights before the actual classification starts.

5. THE UK-NN ALGORITHM

The K-NN algorithm [6] is a lazy method that uses a distance metric (such as the Euclidean distance) to compare a new test instance to those available in the training set. The class of the new instance is set as the majority class of the k closest instances to the new example. These k closest instances are called the k nearest-neighbors.

Consider that we want to create a modified version of the K-NN, named uK-NN, that implements the approach described in Sections 3 and 4. Assume we have already estimated the probabilities of each training document that belong to a context c_t, using the Naive-bayes algorithm. The classical K-NN algorithm can be modified to incorporate uncertainty in at least three ways. In the first one, the probabilities of the instances to belong to a certain context c_t can be used to weight the votes of the instances. In this way, the neighbors of a given test instance (x,y) would remain unchanged, but a weighted majority voting process would be used to establish the class of the test instance.

The result of the voting process of the K-NN usually returns $max(S_c)$, where S_c represents the score of the class c and is given by $max(\sum_{i=1}^{k} V_c(i))$, where $V_c(i)$ returns 1 if the class of the instance i is c and 0 otherwise. In the modified version of K-NN, S_c would be redefined as $max(\sum_{i=1}^{k} \frac{P(c=c_t|x,y)}{P(c=c_t)} V_c(i))$.

Another way to modify the K-NN to deal with uncertainty is to use this same context probability to weight the distances between two neighbors. Consider that the cosine distance metric is used to calculate the distance between two instances. In this case, given a test instance with attribute values $p_1, p_2, \ldots, p_n$, and a training instance with attribute values $q_1, q_2, \ldots, q_n$, the distance between them is given by

$$d_{pq} = \frac{\sum_{i=1}^{n}(p_i \times q_i)}{\sqrt{\sum_{i=1}^{n} p_i^2 \times \sum_{i=1}^{n} q_i^2}}.$$

In order to incorporate data uncertainty to the algorithm, this distance would be modified to

$$wd_{pq} = \frac{d_{pq}}{\frac{P(c=c_t|x,y)}{P(c=c_t)}}.$$

At last, the two previously described approaches could be combined, weighting both the distances and the voting processes.

5.1 Example: Document Classification

In order to illustrate the advantages that the proposed uKNN may offer, this section shows a toy example where dealing with uncertainty helps classification. We use Naive Bayes to estimate the context probabilities and the u3-NN algorithm (uk-NN with k=3) to perform the actual classification.

This example is inspired by text classification problems. We assume a very simple situation, where documents are represented by a set of three terms and they may belong to one of two classes (1 or 2). Each document has been published in one of two different years (2000 and 2001) which are taken to be two different contexts. More specifically, we assume we have a training set of 6 documents as shown in Table 1, where each line represents a document. The first column gives a document ID, the second column gives the set of terms in the document, the third column gives the document class and the fourth column gives the document publication year.

Suppose we would like to classify a test document whose set of terms is {a,c,e} and whose publication year is 2001. Ignoring the context, we can classify it based on the 3-NN strategy, using as similarity measure the number of terms in common. According to this measure, the three closest documents to {a,c,e} are the documents with IDs 2 (class 2), 5 (class 1) and 6 (class 1). Therefore, using majority voting, the document would be classified as belonging to class 1.

Now, in order to use our proposed framework, we need to estimate context probabilities for the target context, which in this case is year 2001. Therefore, we need to estimate for each document belonging to the training set the probability

Table 1: A toy database of documents used as a training set for classification.

ID	Terms	Class	Year
1	{a,b,d}	1	2000
2	{a,b,e}	2	2001
3	{b,c,d}	2	2000
4	{b,d,e}	2	2001
5	{c,d,e}	1	2000
6	{a,b,c}	1	2001

Table 2: Probabilities of the training examples estimated by the Naive Bayes

ID	P(Year=2001| Terms,Class)	w
1	0.4	0.8
2	0.953	1.906
3	0.229	0.458
4	0.6	1.2
5	0.1	0.2
6	0.64	1.28

$P(Year = 2001|Terms, Class)$. We do this using the Naive Bayes algorithm and show the results in the second column of Table 2. We also need to estimate the overall probability $P(Year = 2001)$, which can be done by simply counting the number of documents belonging to year 2001 and dividing it by the total number of documents. For the dataset in Table 1, we have $P(Year = 2001) = 0.5$. The last column of Table 2 shows the ratio $P(Year = 2001|Terms, Class)/P(Year = 2001)$ for each example, which is the weight w that we will use in the classification step.

Now we will consider a version of 3-NN that uses weighted voting, that is, each neighbor votes proportionally to its weight w. In this case, when we classify the test document, the three closest documents are IDs 2 (class 2), 5 (class 1) and 6 (class 1) with weights 1.906, 0.2 e 1.28, respectively. Therefore, we have a total sum of weights of 1.906 for class 2 and 1.48 for class 1. The document will be classified as belonging to class 2. This classification takes into consideration that in 2001 terms b and e are correlated with class 2 (and not with class 1 which was the case in 2000).

We chose the weighted voting strategy for sake of simplicity, but we note that weights can also be incorporated in the nearest-neighbors classification algorithm through a change in the similarity measure. We could, for example, multiply the similarities by the weights.

6. CONCLUSIONS

This paper proposed a new method to estimate data uncertainty and incorporate it to well-known classification algorithms. We introduce the notion of relative data uncertainty, related to data contexts. A context is defined by the data dimension we are interested in. For instance, in the case of document classification, the context can be related to the time the document was produced or the source that produced the document. The idea is that we know the context of an example, but we do not know if the underlying process that generated the data changed from one context to the other. These changes can introduce uncertainty to data when using examples of different contexts to produce a single classification model.

We showed how the K-NN algorithm can be modified to incorporate such changes of context, but a similar process could be used to modify any other lazy classification algorithm. Here we focused on how the temporal evolution of documents can be used to improve their classification. Similarly, the source of the documents could be considered the source of uncertainty, and combining both dimensions in certain applications may also be effective. We are currently performing experiments to corroborate the theoretical framework proposed here.

7. ACKNOWLEDGMENTS

This research is partially funded by the Brazilian National Institute of Science and Technology for the Web, and by the authors's individual research grants from CNPq, CAPES, FINEP, and FAPEMIG.

8. REFERENCES

[1] C. C. Aggarwal. On density based transforms for uncertain data mining. In *Proc. of ICDE*, pages 866–875. IEEE Computer Society, 2007.

[2] C. C. Aggarwal and P. S. Yu. A survey of uncertain data algorithms and applications. *IEEE Trans. on Knowledge and Data Engineering*, 21(5):609–623, 2009.

[3] J. Bi and T. Zhang. Support vector classification with input data uncertainty. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, pages 161–168, 2004.

[4] M. Chau, R. Cheng, B. Kao, and J. Ng. Uncertain data mining: An example in clustering location data. In *Proc. of 10th PAKDD*, pages 199–204, 2006.

[5] C. K. Chui, B. Kao, and E. Hung. Mining frequent itemsets from uncertain data. In *Proc. of 11th PAKDD*, 2007.

[6] T. Cover and P. Hart. Nearest neighbor pattern classification. *Knowledge Based Systems*, 8(6):373–389, 1995.

[7] L. C. da Rocha, F. Mourão, A. M. Pereira, M. A. Gonçalves, and W. Meira Jr. Exploiting temporal contexts in text classification. In *CIKM*, pages 243–252, 2008.

[8] M. Hua and J. Pei. Cleaning disguised missing data: a heuristic approach. In *Proc. of the 13th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 950–958. ACM, 2007.

[9] H.-P. Kriegel and M. Pfeifle. Hierarchical density-based clustering of uncertain data. In *Proc. of the 5th ICDM*, pages 689–692. IEEE Computer Society, 2005.

[10] A. Niculescu-Mizil and R. Caruana. Predicting good probabilities with supervised learning. In *Proc. of the 22nd ICML*, pages 625–632, 2005.

[11] B. Qin, Y. Xia, S. Prabhakar, and Y. Tu. A rule-based classification algorithm for uncertain data. In *1st MOUND 2009 at ICDE*, 2009.

[12] B. Zadrozny, J. Langford, and N. Abe. Cost-sensitive learning by cost-proportionate example weighting. In *Proc. of 3rd ICDM*, pages 435–442, 2003.