A Survey on Postive and Unlabelled Learning

Gang Li
Computer & Information Sciences
University of Delaware
ligang@udel.edu

Abstract

In this paper we survey the main algorithms used in positive and unlabeled learning. The existing methods can be divided into three classes. The first class is a two-step strategy, which tries to identify some reliable negative examples in the unlabeled data first, and then applies supervised learning algorithms on positive data and reliable negative data. The second class tries to weight the positive and unlabeled examples, and estimate the conditional probability of positive label given an example. The third one just treats the unlabeled data as highly noisy negative data. At last we conclude and discuss the future work.

1 Introduction

In supervised machine learning, a binary classifier is usually trained on positive data P and negative data N. But in many situations, positive data is available while negative data is not, and unlabeled data U can be obtained easily. For example, in the field of bioinformatics, there are many corpora in which biological entities are annotated, such as gene, protein, disease, etc. If we want to train a gene name recognizer, then only positive examples (annotated gene) are available, as well as a large set of unlabeled data which we can get by dictionary matching. Another example is a company may have a list of current customers and would like to identify potential customers in some database, then the information of current customers can be viewed as positive data and the people in the database as unlabeled data which contains potential customers as well as non-potential ones. The problem is that traditional supervised learning methods often requires both positive and negative data, but in many cases negative data is not of interest and thus not collected. However, unlabeled data can be easily obtained. We want to learn a binary classifier based on positive data P and unlabeled data U.

There are mainly three classes of methods proposed for positive and unlabeled learning. The most common one is a two-step strategy, including (Liu et al., 2003), (Yu et al., 2002) and (Li and Liu, 2003). (Liu et al., 2003) gives a nice summarization of this kind of methods. The first step usually uses some simple classifiers to identify a set of reliable negative examples RU from unlabeled data, and then applies traditional supervised learning algorithm on P and RU to build a classifier. The second class is to view unlabeled data as negative data, and weight positive and negative examples to estimate the some probability queries of training data. Logistic regression (Lee and Liu, 2003) and weighted SVM (Elkan and Noto, 2008) are two algorithms used in this class of methods. The last class of methods regard the unlabeled data as highly noisy negative data, e.g., there are many true positive examples in it. Biased SVM (Liu et al., 2003) is used in this class of methods.

The rest of the paper are organized as following. In Section 2, we will describe the two-step strategy and go through four methods used in the first step, because this is the most important part for solving this problem, and then list the supervised learning algorithms used in the second step. In Section 3, we will review two methods of the second class, using logistic regression and weighted SVM , respectively. In Section 4, we will review Biased SVM for the third class of methods, which uses an asymmetric cost function to tolerate some noise in negative data and learn a classifier. Finally we conclude and discuss the future work in Section 5.

2 Two-step Strategy

The two-step strategy is very straight-forward. Because we don't have negative data, and unlabeled data contains both positive and negative examples, we can try to identify some reliable negative data RN from U. Traditional supervised learning algorithms then can be applied on P and RU to learn a classifier.

2.1 Methods for Step 1

We will review four methods for the first step in this subsection: Rocchio, Naive Bayesian Classifier, Spy technique and 1-DNF method. We take document classification problem to explain these methods.

2.1.1 Method 1: Rocchio

Rochhio classification algorithm (Rocchio, 1971) is an early method. It was used to classify documents in a document set D. Each document d is represented by a feature vector in which each feature is some IR score, such as tf-idf score. The classes in the documents set are characterized by a prototype vector. Let C_j be the jth class of documents, $c_j$be the prototype vector of C_j. Then c_j is computed in training as in formula below. $\alpha$and β are the weights for the documents in C_j and $D-C_j$. In testing, for a document d, some similarity function f can be used to compute a similarity score between d and c_j. The class with the highest similarity score is assigned to d.

$$\overrightarrow{c_j} = \alpha\frac{1}{|C_j|}\sum_{\overrightarrow{d}\in C_j}\frac{\overrightarrow{d}}{||\overrightarrow{d}||} - \beta\frac{1}{|D-C_j|}\sum_{\overrightarrow{d}\in D-C_j}\frac{\overrightarrow{d}}{||\overrightarrow{d}||}$$

Rocchio algorithm is used to build a prototype vector for P and U, respectively. Then the prototype vectors are used to classify unlabeled examples in$U$. If the example has a higher similarity score with the negative prototype vector than with the positive one, then it's regarded as reliable negative example to form RU.

2.1.2 Method 2: Naive Bayesian Classifier

The naive Bayesian method is used very often in classification. Let D be the set of documents we want to classify, $C = c_1, c_2, \ldots, c_n$ be the predefined classes of the documents, $V = x_1, \ldots, x_{|V|}$ be the vocabulary where x_i is a word. Naive Bayesian (NB) classifier computes the conditional probability $P(c_j|d_i)$ for a given document d_i. The class $c_j$with the highest probability is assigned to the document. First, we compute the probability $P(c_j)$ for each class as below.

$$P(c_j) = \frac{\sum_{i=1}^{|D|} P(c_j|d_i)}{|D|}$$

The probability $P(c_j|d_i)$ is just 1 if d_i belongs to c_j, otherwise it's 0. We use $N(x_t, d_i)$ to denote the number of word x_t appearing in document d_i. Then we can compute the probability for a word x_t given a class $P(x_t|c_j)$ by

$$P(x_t|c_j) = \frac{\sum_{i=1}^{|D|} N(x_t, d_i)P(c_j|d_i)}{\sum_{s=1}^{|V|}\sum_{i=1}^{|D|} N(x_s, d_i)P(c_j|d_i)}$$

Some necessary smoothing techniques can be used in case some words in V don't appear in a certain class of documents.

At last, we assume the words probabilities are independent given the class, then we get the NB classifier as below. $x_{d_i,k}$ is the word in kth position in the document d_i.

$$P(c_j|d_i) = \frac{P(c_j)\prod_{k=1}^{|d_i|} P(x_{d_i,k}|c_j)}{\sum_{r=1}^{|C|} P(c_r)\prod_{k=1}^{|d_i|} P(x_{d_i,k}|c_r)}$$

Given a document d_i, we go through all the words of it using the formula above, and compute $P(c_j|d_i)$. The class with the highest probability is assigned to d_i. To identify a reliable negative data set RN from unlabeled data U, we train a NB classifier on P and U, and use it to classify U. If $P(Positive|x) < P(Unlabelled|x)$, then we extract example x as a reliable negative example.

2.1.3 Method 3: The Spy Technique

The Spy technique is described in (Liu et al., 2002). It first randomly selects a set of positive examples S from P, and then put S into U. Then the examples in S act similarly with the unknown positive examples in U. It then runs Expectation Maximization (EM) algorithm on $P - S$ and $U + S$ to train a NB classifier. The EM method consists of two steps, the Expectation step and the Maximization step. In the Expectation step the classifier predicts label for examples using the third formula in Section 2.1.2, and in Maximization step parameters of the classifier are re-computed using the first and second formula in Section 2.1.2. It iterates these two steps until the parameters are stabilized. Then the trained classifier is run on S, and we can get a probability threshold h below which the example should be classified as negative. At last we use the NB classifier with h on U to identify reliable negative examples.

2.1.4 Method 4: The 1-DNF Technique

The 1-DNF (Yu et al., 2002) method tries to extract some positive features by comparing P and U. It counts the feature frequencies in P and U and use those features with higher frequencies in P than in U as positive features. Then it check all the examples in U, and extract those examples containing no positive features as strong reliable negative examples.

2.2 Methods for Step 2

Once we obtain the reliable negative data set RU, the problem becomes a traditional binary classification problem that can be solved by supervised learning algorithm. Here we just list 4 methods used in step 2, but since they are discussed a lot in many machine learning papers, we won't go into details about them.

1. Run SVM only once on sets P and RU.
2. Run EM algorithm on P and RU.
3. Run SVM on P and RU iteratively, until no more reliable negative data can be found.
4. Run SVM on sets P and RU, and select a best classifier in the generated models.

Method 2 is already described in Section 2.1.3. Method 3 and 4 are similar. Both of them train a SVM classifier on P and RU, and run it on $U - RU$. We use Q to denote the newly found negative examples in $U - RU$. In the next iteration, they continue to train a SVM classifier on P and $RU + Q$, until no more reliable negative examples can be found in $U - RU$. The difference is that method 3 chooses the last SVM classifier as the final classifier, while method 4 tries to select the best classifier out of the array of generated SVM classifiers in each iteration, because the last one may not be the best due to overfitting. In experiments, SVM methods are better than EM method, because EM trains a NB classifier iteratively. There are two assumptions made by NB classifier which are not true in reality, 1) words are independent given a class, 2) a document are generated by a single underlying class. The more we run NB, the more errors may be cumulated.

3 Methods based on Weighted Positive and Unlabeled Data

This class of methods try to model the positive label probability given an example, $P(y = 1|x)$. They either weight both positive and unlabeled examples to learn a weighted probability, which is lower bounded by 0.5, or weight unlabeled examples to learn a general function, and to obtain a good estimate of $P(y = 1|x)$ from P and U.

3.1 Weighted Logistic Regression

Let y be the actual label of an example x, y' be the predicted label of x. The weighted logistic regression method uses logistic regression to model the probability of positive label given an example $P(y = 1|x)$ based on weighted positive and unlabeled examples. It weights the examples in order to make the weighted $P(y' = 1|y = 1, x)$, which is the expected true positive ratio, greater than 0.5, and make the $P(y' = 1|y = 0, x)$, which is the expected false positive ratio, less than 0.5. Thus logistic regression can model the probability $P(y = 1|x)$ and then classify the examples by threshold 0.5.

The probability of positive label given a positive example is $P(y' = 1|y = 1, x)$, and the probability of negative label given a positive example is $P(y' = -1|y = 1, x)$. (Lee and Liu, 2003) proves that if we weight these two probabilities by $P(y' = -1)$ and $P(y' = 1)$ respectively, the weighted $P(y' = -1|y = 1, x)$ is greater than 0.5. First, we have the weighted probabilities

$$P_1 = P'(y' = 1|y = 1, x)P(y' = -1)$$
$$P_2 = P(y' = -1|y = 1, x)P(y' = 1)$$

The weighted probability of positive label given it's a positive example is computed as

$$P'(y' = 1|y = 1, x) = \frac{P_1}{P_1 + P_2}$$

Thus it's possible to learn a linear function $g(x)$ to model $P'(y = 1|x)$ by the sigmoid function using logistic regression on the weighted examples. In training, Maximum Likelihood Estimation is used and the cost is optimized using simple gradient descent since the cost function is convex. The experiments result showed that this is an effective method.

3.2 SVM based on Weighted Unlabeled Data

In this method, Elkan et al. tries to estimate the probability $P(y = 1|x)$ where the x is an example in the data set and y is the actual label of it. Let s be the label state of an example in training set, i.e., if x is labeled in P, $s = 1$, otherwise $s = 0$. They assume that P and U are drawn randomly from $p(x, y, s)$ and claim that a general function $h(x, y)$ under distribution $p(x, y, s)$ can be estimated as below.

$$\begin{aligned} E(h) = \frac{1}{m}(&\sum_{x,s=1} h(x, 1) \\ &+ \sum_{x,s=0} w(x)h(x, 1) + (1 - w(x))h(x, 0)) \end{aligned}$$

m is the cardinality of the training set, $w(x)$ is a weight estimated on a development set by a non-traditional classifier trained on P and U. The non-traditional classifier can be trained by any supervised learning algorithm. Based on this formula, each example in U can be viewed as a weighted positive example as well as a weighted negative example. Using SVM to train a classifier on P and weighted U can yield a well-calibrated classifier, which models $P(y = 1|x)$ approximately. Each example in U is used twice in training.

This method requires the classification algorithm outputs probability directly, which should be in [0,1]. But SVM classifier's output is not the probability of the label given an example, so some post-processing methods are needed to convert classifier's outputs into probability. In (Elkan and Noto, 2008), Platt scaling is used.

In summary, this method first trains a non-traditional classifier on P and U, and uses it to estimate the weight $w(x)$ on a development set. Next, weight the unlabeled examples and model the probability $P(y = 1|x)$ using SVM.

4 Methods based on Noisy Negative Data

4.1 Biased SVM

Biased SVM regards the unlabeled data as highly noisy negative data. Soft-margin SVM is a good way to handle noisy data. It uses a slightly different objective function and constraints in optimization as below.

$$\begin{aligned} min \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C_+\sum_{i=1}^{k-1}\xi_i + C_-\sum_{i=k}^{n}\xi_i \\ s.t. \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, i = 1, 2, \ldots, n \\ & \xi_i \geq 0, i = 1, 2, \ldots, n \end{aligned}$$

The slack variable ξ is used to tolerate some in-margin examples or even errors. C_+ and C_- are regularization terms for positive data and unlabeled data. Since unlabeled data are used as noisy negative data, C_- should be small and C_+ should be large. Small C_- will allow more errors on the negative side since ξ for negative data can be larger than ξ for positive data with regard to optimization. Thus it can learn a classifier between positive and noisy negative data. In training, C_+ and C_- are selected on a development set.

5 Conclusion

In this survey we reviewed the main algorithms used in Positive and Unlabeled Learning. These algorithms are already used in practical projects and achieved good results in (Cerulo et al., 2010) and (Li et al., 2011). They all use information in the unlabeled data set U to get a better classifier. The two-step strategy is the most common one since it's easy to explain and implement. The Biased SVM and weighted SVM (SVM-WU) are the two state-of-art methods for this problem. Both of them use a development set to estimate some parameters for learning the classifier. In comparison, SVM-WU only needs to go through the development set once to get the weight, while Biased SVM needs to iterates on development set until convergence, to get the regularization term C_+ and C_-.

For future work, it would be interesting to see how these algorithms perform on more practical problems. It may also be beneficial to look into the probabilistic queries carefully and question their reasonability. For example, SVM-WU assumes that we can learn $P(s = 1|x)$ on P and U. However, features in x may not affect s much, and we can try to include more features that may really affect s.

References

Luigi Cerulo, Charles Elkan, and Michele Ceccarelli. 2010. Learning gene regulatory networks from only positive and unlabeled data. *BMC Bioinformatics*, 11:228.

Charles Elkan and Keith Noto. 2008. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '08, pages 213–220, New York, NY, USA. ACM.

Wee Sun Lee and Bing Liu. 2003. Learning with positive and unlabeled examples using weighted logistic regression. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML*, page 2003.

Xiaoli Li and Bing Liu. 2003. Learning to classify texts using positive and unlabeled data. In *Proceedings of the 18th international joint conference on Artificial intelligence*, IJCAI'03, pages 587–592, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Wenkai Li, Qinghua Guo, and C. Elkan. 2011. A positive and unlabeled learning algorithm for one-class classification of remote-sensing data. *Geoscience and Remote Sensing, IEEE Transactions on*, 49(2):717–725.

Bing Liu, Wee Sun Lee, Philip S. Yu, and Xiaoli Li. 2002. Partially supervised classification of text documents. In *Proceedings of the Nineteenth International Conference on Machine Learning*, ICML '02, pages 387–394, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Bing Liu, Yang Dai, X. Li, Wee Sun Lee, and P.S. Yu. 2003. Building text classifiers using positive and unlabeled examples. In *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*, pages 179–186.

J. J. Rocchio. 1971. Relevance feedback in information retrieval. In G. Salton, editor, *The Smart retrieval system - experiments in automatic document processing*, pages 313–323. Englewood Cliffs, NJ: Prentice-Hall.

Hwanjo Yu, Jiawei Han, and Kevin Chen-Chuan Chang. 2002. Pebl: positive example based learning for web page classification using svm. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 239–248, New York, NY, USA. ACM.

Bangzuo Zhang and Wanli Zuo. 2008. Learning from positive and unlabeled examples: A survey. In *Information Processing (ISIP), 2008 International Symposiums on*, pages 650–654.