RUNNING PU ON SPARK WITH SCALA

QI, Xiaoxu

2016. 12. 29

SCHEDULE

- What is “PU”
- How to perform “PU”
- Works on Spark
- Experiment Result
- Basic Data Operation on Spark
- Introduction to Hope
- Reference
- Questions & Answers

WHAT IS “PU”

- ***P***ositive and ***U***nlabelled Samples
- Partially Supervised Classification
- Unbalanced training samples: normally the ctr is between 0.01% and 9%
- Small amount of positive samples
- Should not consider all the unlabelled samples as negative samples(not exposed or not interested)
- To dig out “potential" users who might be interested in the launch

HOW TO PERFORM PU

- Step 1: find the samples which can be considered as "Reliable Negative" samples
  - Naive Bayesian
  - EM
  - SVM
  - Decision Tree
  - LR

1. $RN = NULL$;
2. $S = Sample(P, s\%)$;
3. $Us = U \cup S$;
4. $Ps = P - S$;
5. Assign each document in Ps the class label 1;
6. Assign each document in Us the class label -1;
7. I-EM(Us, Ps); // This produces a NB classifier.
8. Classify each document in Us using the NB classifier;
9. Determine a probability threshold th using S;
10. **for** each document $d \in Us$
11. **if** its probability $Pr(1|d) < th$ **then**
12. $RN = RN \cup \{d\}$;

Figure 2: The Spy technique in S-EM.

HOW TO PERFORM PU

- Step 2: exploit the ***R***eliable ***N***egative samples

1. Each document in P is assigned the class label 1;
2. Each document in RN is assigned the class label -1;
3. Each document $d \in Q$ ($= U - RN$) is not assigned any label initially. At the end of the first iteration of EM, it will be assigned a probabilistic label, $Pr(1|d)$. In subsequent iterations, the set Q will participate in EM with its newly assigned probabilistic classes.
4. Run the EM algorithm using the document sets, P, RN and Q until it converges.

- Now we have Positive samples, Negative samples and Unlabelled samples. With P and N samples, a classification model can be trained to label more samples from unlabelled samples.

- Then we update positive samples set and negative samples set to train a new model. Until it converges….

HOW TO PERFORM PU

- Strategy for converge
- converge of EM: the centroid is stable, the cost of changing centroid does not decrease, etc.
- Based on the probability score of the spy in Unlabelled samples
- Based on posterior probability
- Based on percentage of positive samples over the overall samples

WORKS ON SPARK

- Available features:
- To run program on spark, we need to initialise a SparkConf
- To query SQL, we need to initialise a HiveContext
- To speed up the calculation, RDD should reside in cache

WORKS ON SPARK

- Label and transform the samples to LIBSVM format.

```
    val traindata = userrdd.map(x =>
      if (seedlist.contains(x.get(0))) {
/.../
        LabeledPoint(1.0, Vectors.dense(x.getDouble(1),x.getDouble(2),x.getDouble(3),x.getDouble(4),

      } else {
//        println("Negative")
        LabeledPoint(0.0, Vectors.dense(x.getDouble(1),x.getDouble(2),x.getDouble(3),x.getDouble(4),
      } ).cache()
```

- Official manual for MLLIB:

- https://spark.apache.org/docs/2.0.1/api/java/org/apache/spark/mllib/classification/LogisticRegressionModel.html

WORKS ON SPARK

- Train LR model and predict the labels for unlabelled samples
- In PU STEP 1

```
/.../

    val model = new LogisticRegressionWithLBFGS().run(traindata)
//    print("Model")
    println(model.getClass.getSimpleName)
//    println(model.weights)
//    println()

    model.clearThreshold()
    var predictionMap = scala.collection.mutable.Map[Long,Double]()
    var in:Long = 0
//    traindata.collect().foreach(t => print(model.predict(t.features)))
    val preditionResult = traindata.map{case LabeledPoint(label,features) => val prediction = model.predict(features)
      (prediction,label,features)}
```

WORKS ON SPARK

PU STEP 1

- GOAL: Find the most unlikely positive samples, reliable negative samples, to initialise a classifier.

```
breakable {
  sortedPredictionMap.keys.foreach(u => {
    if (seedlist.contains(useridMap.get(u).get.asInstanceOf[Long])) {
      pos_dpid += useridMap.get(u).get.asInstanceOf[Long]//fix
    }
    else if (cnt < pos_cnt) {
      if (!pos_dpid.contains(useridMap.get(u).get.asInstanceOf[Long])) {
        cnt = cnt + 1
        neg_dpid += useridMap.get(u).get.asInstanceOf[Long]
        println(u, useridMap.get(u), sortedPredictionMap.get(u))
      }
    } else {
      println("Finish building of initial positive list and negative list in STEP 1")
      break()
    }
  })
}
seedlist.foreach(u=>{
  pos_dpid += u
})
```

WORKS ON SPARK

PU STEP 2

- Run the classification iteratively
- Update positive/negative sample lists
- Converge depends on EM [number of selected samples are unpredictable]
- Probability threshold depends on the noise level [lack of priori knowledge]
- To simplify, in every iteration, the ratio of selected positive samples to overall samples is predefined.

WORKS ON SPARK

PU STEP 2

- Train classification model iteratively and update the positive / negative sample list
- Strategy: the positive samples persist and the negative samples are changeable in each iteration

```
//    println("Iterate score map in descending order")
    breakable {
      sortedPredictionMap2.keys.foreach(u => {
        if (pos_dpid.contains(useridMap.get(u).get.asInstanceOf[String])) {
//          pos_dpid.distinct
        }
        else if (cnt < sumcnt * iteration_ratio) {
          cnt = cnt + 1
          pos_dpid += useridMap.get(u).get.asInstanceOf[String]
//          println(u, useridMap.get(u), sortedPredictionMap.get(u).get.asInstanceOf[Double])
          minProbability = sortedPredictionMap2.get(u).get.asInstanceOf[Double]
          if (neg_dpid.contains((useridMap.get(u).get.asInstanceOf[String]))) {
            neg_dpid -= useridMap.get(u).get.asInstanceOf[String]
          }
        } else {
          println("Finishing updating positive list in STEP 2")
          break()
        }
      })
    }
```

WORKS ON SPARK

Log Type: stdout

Log Upload Time: 星期四 十二月 29 12:34:23 +0800 2016

Log Length: 3393

```
setAddStackTraceToMessage = true
[20161229-1231]Starting data extraction in HiveSQL
[20161229-1232]Starting to label data in main
[20161229-1232]CALL pu_step1(userrdd: RDD[GenericRowWithSchema], traindata: RDD[LabeledPoint], seedlist:List[String]): (ListBuffer[String],ListBu
[20161229-1232]Starting PU STEP 1
[20161229-1232]Finishing Model Training in PU STEP 1
[20161229-1232]Finishing prediction in PU STEP 1
[20161229-1232]Finish building of initial positive list and negative list in STEP 1
[20161229-1232]Finishing pu_step1
[20161229-1232]Beginning iteration for PU STEP 2 in main
[20161229-1232]CALL pu_step2(userrdd: RDD[GenericRowWithSchema], traindata: RDD[LabeledPoint], indata: RDD[LabeledPoint], pos_dpid: ListBuffer[St
[20161229-1232]Starting PU STEP 2
[20161229-1232]Finishing Model Training in PU STEP 2
[20161229-1232]Finishing prediction in PU STEP 2
[20161229-1233]Finishing updating positive list in STEP 2
Finishing updating negative list in STEP 2
[20161229-1233]CALL pu_step2(userrdd: RDD[GenericRowWithSchema], traindata: RDD[LabeledPoint], indata: RDD[LabeledPoint], pos_dpid: ListBuffer[St
[20161229-1233]Starting PU STEP 2
[20161229-1233]Finishing Model Training in PU STEP 2
[20161229-1233]Finishing prediction in PU STEP 2
[20161229-1233]Finishing updating positive list in STEP 2
Warning: Reach boundary of positive and negative samples in STEP 2, break()
[20161229-1233]CALL pu_step2(userrdd: RDD[GenericRowWithSchema], traindata: RDD[LabeledPoint], indata: RDD[LabeledPoint], pos_dpid: ListBuffer[St
[20161229-1233]Starting PU STEP 2
[20161229-1233]Finishing Model Training in PU STEP 2
[20161229-1233]Finishing prediction in PU STEP 2
[20161229-1233]Finishing updating positive list in STEP 2
Warning: Reach boundary of positive and negative samples in STEP 2, break()
[20161229-1233]CALL pu_step2(userrdd: RDD[GenericRowWithSchema], traindata: RDD[LabeledPoint], indata: RDD[LabeledPoint], pos_dpid: ListBuffer[St
[20161229-1233]Starting PU STEP 2
[20161229-1233]Finishing Model Training in PU STEP 2
[20161229-1233]Finishing prediction in PU STEP 2
[20161229-1233]Finishing updating positive list in STEP 2
Warning: Reach boundary of positive and negative samples in STEP 2, break()
[20161229-1233]CALL pu_step2(userrdd: RDD[GenericRowWithSchema], traindata: RDD[LabeledPoint], indata: RDD[LabeledPoint], pos_dpid: ListBuffer[St
[20161229-1233]Starting PU STEP 2
[20161229-1234]Finishing Model Training in PU STEP 2
[20161229-1234]Finishing prediction in PU STEP 2
[20161229-1234]Finishing updating positive list in STEP 2
Warning: Reach boundary of positive and negative samples in STEP 2, break()
([,/user/hadoop-necpm/pu/output/uid20161229-1231/result,] Finishing PU process with ,5868, positive samples selected, within ,5, iterations!)
Hello World
```

BASIC DATA OPERATION ON SPARK

- DataFrame
- RDD [should be cached to improve performance]
- ROW
- Map
- List
- Some
- …

REFERENCE

- 1. Building Text Classifiers Using Positive and Unlabelled Examples
- 2. Partially Supervised Classification of Text Documents
- 3. http://spark.apache.org/docs/latest/api/scala/index.html#org.apache.spark.sql.Dataset
- 4. https://spark.apache.org/docs/2.0.1/api/java/org/apache/spark/mllib/classification/LogisticRegressionModel.html

“

QUESTIONS & ANSWERS
THANKS

QI, Xiaoxu

”