Adaptive Sampling Scheme for Learning in Severely Imbalanced Large Scale Data

Wei Zhang WZHANG@ADOBE.COM
Said Kobeissi SKOBEISS@ADOBE.COM
Scott Tomko STOMKO@ADOBE.COM
Chris Challis CHALLIS@ADOBE.COM
Adobe Systems, Mclean, USA

Editors: Yung-Kyun Noh and Min-Ling Zhang

Abstract

Imbalanced data poses a serious challenge for many machine learning and data mining applications. It may significantly affect the performance of learning algorithms. In digital marketing applications, events of interest (positive instances for building predictive models) such as click and purchase are rare. A retail website can easily receive a million visits every day, yet only a small percentage of visits lead to purchase. The large amount of raw data and the small percentage of positive instances make it challenging to build decent predictive models in a timely fashion. In this paper, we propose an adaptive sampling strategy to deal with this problem. It efficiently returns high quality training data, ensures system responsiveness and improves predictive performances.

Keywords: Imbalanced Data, Large Scale Data, Machine Learning Application, Digital Marketing

1. Introduction

Businesses see growing benefits from big data and machine learning. Marketers use machine learning to target specific customer segments, e.g., customers who are likely to purchase some product. Financial industries can use machine learning for fraud detection. In practice, however, the problem of imbalanced data (the size of one class is significantly larger than another) often emerges. Some applications, such as fraud detection Phua et al. (2004) biomedical applications Oh et al. (2011) and anomaly detection Chandola et al. (2009), are susceptible to severe data imbalance. In the digital marketing field, the data imbalance problem is ubiquitous as events of interest are typically just a small portion of the overall population. For example, in a seven day period, a retail website got over 42 million visits, but only received about 16 thousand purchases. When businesses want to predict which customer will make a purchase, these customers who made purchases are defined as positive instances and the rest are defined as negative instances, then a predictive model (e.g. logistic regression) can be learned using the data. In this case, the ratio of positive/negative instances is less than 1:2500. The imbalance problem can be even worse; when selecting higher return customers with "revenue ≥ 100", the ratio is below 1:10000. Since the size of the negative class is much larger than the positive class, we also call the negative class the "majority class", and the positive class the "minority class". In general, standard learning algorithms expect balanced class distributions. When dealing with severely imbalanced data, they tend to overfit training data and perform unfavorably on unseen data He and Garcia (2009).

In the literature, the data imbalance problem can be classified into two types: *intrinsic* and *extrinsic* He and Garcia (2009). *Intrinsic* refers to the imbalance resulting from the nature of the data such as the above mentioned example of retail data. *Extrinsic* refers to imbalances due to other causes such as time and storage. For instance, a data stream is balanced overall yet its minority instances are not *i.i.d.* (independent and identically distributed). Taking digital marketing data as an example, the distribution of purchasing events is not *i.i.d.*. Because visits in different time periods could have different probability of placing orders, due to various factors such as customer behavior and time zone (geo-location) differences.

Because the amount of data is huge, it is too expensive to use all data to train a predictive model. Not only that, just loading all data instances from disks already has a big impact on system responsiveness. Therefore in order to ensure the system to be responsive for its users, we have to use a portion (sample) of data, rather than the entire data, to train predictive models. Due to the nature of big data, even a small portion of data contains a large number of data instances. For example, 0.1% of a data set of size 200 million has $200k$ (2×10^5) instances. This is usually enough for building a decent predictive model if it contains enough minority instances. However, taking a sample of data makes the imbalance problem even worse: 1. the absolute number of positive instances will be significantly reduced; 2. if the data is not *i.i.d.* distributed, the percentage of positive instances in a random sample could be even less than the original data. Due to the limited number of positive instances, different random samples could have quite different positive instances. Consequently, models estimated using these different samples will have high variances and are unlikely to be repeatable. In this paper, we propose an adaptive sampling scheme, which helps to handle large scale data that are severely imbalanced, e.g., website traffic data with hundreds of millions of visits but few events of interest. The adaptive sampling module generates more balanced data samples efficiently, yielding more repeatable and reliable predictive models.

2. Related Work

Kubat and Matwin Kubat et al. (1997) proposed to tackle the imbalance problem by down-sampling the majority class while keeping the minority class. Ling and Li Ling and Li (1998) combined under-sampling of majority class with over-sampling of minority class. However, this does not significantly help minority class recognition Chawla et al. (2002). The Synthetic Minority Over-sampling Technique (SMOTE) algorithm Chawla et al. (2002) improves class balance by creating synthetic positive instances using interpolation. It is widely used because of its effectiveness and simplicity. One problem is that these algorithms need to know the ratio between the minority class and the majority class *a priori*, in order to set how much over-sampling/under-sampling should be done and how many synthetic instances to create. Users of our system can set all kinds of target variables, e.g., “place an order” or “spend over 1000 dollars”, to build predictive models for their business goals. The ratio will be different for different target variables, thus it will not be readily available without scanning the data, which may be expensive. In addition, the SMOTE algorithm may not be suitable for digital marketing data, where the data has a high dimensionality (hundreds to thousand) as they cover all aspects of customer visits, including both numerical and categorical attributes (variables), for example: "Page views", "Time", "Geolocation"(e.g., "Japan"), "Browser type"(e.g., "Firefox 42.0") and "Campaign type"(e.g., "discount 20%"). It has been shown that SMOTE is less effective for high dimensional data Wallace et al. (2011) Blagus and Lusa (2012). Moreover, SMOTE operates on the entire available data. It would be too costly if we apply them directly on the original raw data. One solution is to take a random sample of the original data first, then apply the SMOTE algorithm to make it more balanced. However, this is not very helpful as we will show in the experiments section. The reason can be explained using the previous data as an example. With a positive/negative ratio of 1:2500, a sample of $200k$ instances will only have ~ 80 positive instances. Statistics literature Harrell et al. (1984) states that one needs a decent number of minority instances to have stable logistic regression estimates, "one in 10" is the minimum requirement and "one in 20" rule has been suggested Steyerberg and Eijkemans. For example, to use regression analysis on 40 predictor variables, $40 \times 20 = 800$ minority instances are needed to avoid overfitting. Obviously, due to the high dimensionality of our data, 80 instances are too few to capture the distribution of the positive class. So it is impractical to generate enough reliable positive instances using interpolation.

3. Our Approach

In order to achieve the high system performance, both in term of response time and accuracy, we propose an adaptive sampling scheme which ensures that sufficient minority instances will be collected efficiently. In addition, the majority class will be uniformly sampled to address the potential problem that data is not *i.i.d.*.

3.1. Adaptive Sampling Scheme

Our adaptive sampling scheme is summarized in Algorithm 1. It treats data as a stream of positive and negative instances.

The notation of the algorithm is as follows: N is the target sample size for building a predictive model, for example $200k$. M is the number of parameters to be estimated in the model, e.g., the number of channels in the attribution model or the number of predictor variables in the regression model; N_r is the number of desired positive instances, which is set to be $N_r = 50M$. In other words, we use a "one in 50" rule to determine the number of positive instances. It requires more positive instances than the "one in 20" rule, and we find empirically this leads to reliable model estimation. N_n is the number of negative instances collected from data so far. N_p is the number of positive instances collected from data so far. N_t is the total number of negative instances scanned so far.

Step 1a keeps accumulating negative instances until the size reaches $N - N_r$. After that each incoming negative instance has a N_n/N_t probability of being put into the list of negative instances. N_n/N_t gradually decreases as N_t increases. Ultimately all negative instances which have been scanned have the equal probability $1/N_t$ of being put into the list. So we get a random sample of negative instances from the original data. Step 1(b)iiB only happens if the data is abundant with positive instances, i.e., we have collected enough

Algorithm 1 The adaptive sampling algorithm

```
1. Check the current data instance to see if it is a positive or negative:
   (a) If it is a negative instance, do reservoir sampling Vitter (1985):
       i. $N_t = N_t + 1$.
       ii. A. If $N_n < N - N_r$, add it to the list of negative instances, denote as $L_n$.
              $N_n = N_n + 1$.
           B. Otherwise replace an instance in $L_n$ with the current one:
              - Randomly generate an integer $i$ between 0 and $N_t - 1$.
              - If $i < N_n$, replace $L_n[i]$ with the current negative instance.
   (b) If it is a positive instance:
       i. If $N_p < N_r$, add it to the list of positive instances, denote as $L_p$. $N_p = N_p + 1$.
       ii. Otherwise,
           A. If $N_n + N_p \geq N$, the desired data sample has been collected. Combine
              $L_p$ and $L_n$ as the final sample, return with success.
           B. Else do reservoir sampling for positive instances.
2. If there is no data left, combine $L_p$ and $L_n$ as the final sample (that is the best we
   can do), return with warning. Otherwise, move to the next instance and go to 1.
```

positive instances before the sample has reached the target size. Step 2 exits when the data is extremely imbalanced, such that the entire raw data does not have the desired number of positive instances. This is not typical, but occurs sometimes when the criterion for positive instances is set to be very strict.

In our system, a cluster of machines are used to store and process data, N and N_r are divided by the number of machines when doing sampling within each machine. Once the sampling is done in all machines, the result samples are combined into one final sample. The data instances in this final sample are randomly shuffled before they are fed to the predictive module for training and testing.

As can be seen from the sampling algorithm:

1. When the data is severely imbalanced (very few positive instances), it will keep scanning, going through a large portion of data, to obtain enough positive instances. In the worst case when the positive instances are extremely rare, it might scan through the entire data to collect a more balanced sample. On the other hand, when the data is less imbalanced, it will stop sampling very early (e.g., stops after only checking 1% of data) because enough positive instances have been retrieved already.

2. This means that for more balanced data, the algorithm stops early and is very efficient, while for imbalanced data, the algorithm will run for as long as it takes to find the desired number of positive instances. In other words, it *adapts* to the distribution of the data. It does not matter if the source of imbalance is intrinsic or extrinsic, the adaptive sampling algorithm handles it without the need to know *a priori* the ratio

or the type of imbalance. In comparison, other methods, e.g., under-sampling the majority class, over-sampling the minority class, Synthetic Minority Over-sampling Technique (SMOTE), go through the entire data. Scanning the entire data is very expensive for the kind of big data that our system needs to deal with. For example, for the small web traffic test data (42 million high dimensional customer visits data), it takes ~ 1 minute to scan through the data with our testing setting of a two machine cluster. Actual data are often 10× or 100× larger, it would surely adversely impact our customer experiences. The proposed adaptive algorithm only needs to scan the entire data when absolutely necessary (the case of extremely rare positive instances). Most of the time, it finishes earlier and ensures the effectiveness and efficiency of the system.

3. More formally, suppose the number of instances in the raw data is n, the size ratio between the minority/majority classes is r, the complexity of SMOTE is $O(n)$. The proposed algorithm is $O(N_r/r)$, which is usually much less than $O(n)$. Using the previous web traffic data as an example, when building a model using 40 predictor variables, $N_r = 50 \times 40 = 2000$, if the ratio is 1/2500, $N_r/r = 5 \times 10^6$, which is much less than 42 million. Only in the worst case when there is not enough minority instances in the entire data set, the complexity is $O(n)$.

3.2. Discussion About Alternatives

One might also come up with two alternative solutions to the problem:

1. Do simple stratified sampling: keep sampling, and put positive instances and negative instances into two separate lists until they reach the desired size. Stop when both lists reach their target sizes.

2. Scanning through the entire dataset, and do reservoir sampling for both positive instances and negative instances. This will ensure a random sample of the original data, even when the original data distribution is highly non-*i.i.d.* (high extrinsic imbalance).

The first alternative also has the complexity $O(N_r/r)$ as it needs to scan that amount of data to collect enough positive instances. In practice, it is a little faster than the proposed algorithm. Because it just uses the first $N - N_r$ negative instances in the original data, and stops processing further. However, it is only optimal when the data distribution is *i.i.d.*. When the data is not perfectly *i.i.d.*, it brings the risk of having a biased negative distribution. This happens in web traffic data as people across the world live in different time zones, thereby visits in day and night time are likely to come from different geo-locations. Accordingly, the negative instances in the unseen test data may be out of the distribution of the training sample. As a result, it could lead to inferior results. The second alternative, although appealing as it can handle highly non-*i.i.d.* data, requires always scanning the entire data. It is too expensive for our application. Our solution is much more efficient than the second alternative. And it is virtually as efficient as the first alternative, while bringing robustness against moderately non-*i.i.d.* data distributions.

4. Experiments

Although it is apparent that the proposed algorithm generates better samples and leads to better predictive models, we need to verify its advantage through experiments. Each time after obtaining a data sample, we randomly split it and use 80% of it for building the predictive model. Then we evaluate the model on the remaining 20% data, so that users can immediately see the predictive performance on unseen data. The confusion matrix is used to illustrate the predicative performance as shown in Table 1 and Table 2. A good prediction system will predict "true negative" as "negative", and "true positive" as "positive". Thus off-diagonal entries in the confusion matrix represent wrong predictions, and larger numbers in the diagonal entries indicate better performance. Accordingly we can define "TP" as the number of correctly classified positives, "TN" as the number of correctly classified negatives, while "FP" as the number of negatives incorrectly classified as positives and "FN" as the number of positives incorrectly classified as negatives. We also use *precision* and *recall* to quantitatively measure the performance. *Recall* is defined as $\frac{TP}{TP+FN}$, measuring the percentage of the true positive identified as positive. *Precision* measures the percentage that predicted positives are true positives. It is defined as $\frac{TP}{TP+FP}$.

Table 1: Performance of predictive models built based on random sampling vs. adaptive sampling, on their respective test samples.

	Random sampling		Adaptive sampling	
	Predicted negative	Predicted positive	Predicted negative	Predicted positive
True negative	39925	39	39839	161
True positive	344	16	1023	4977
Recall	4.44%		82.95%	
Precision	29.09%		96.86%	

The algorithm was integrated into an analytics software system and tested extensively by quality engineers and 3^{rd} party users on different web traffic datasets. All tests showed that predictive models built based on the adaptive sampling perform significantly better than models built based on the random sampling, especially for severely imbalanced data. We use a real retail traffic data with over 10 million records as one example to demonstrate the superiority. In our experiments, the target sample size is set to be $200k$. For each data sample, either obtained by adaptive sampling or random sampling, we use the standard SMOTE algorithm Chawla et al. (2002) to create synthetic positive instances, to make the sample more balanced. Thus the size of the sample test set is a little more than $40k(200k \times 20\%)$. Table 1 shows a typical performance comparison, when building a L2 regularized logistic regression model for predicting the target variable "place an order" (i.e., the customer makes a purchase, happening about 1 in 1190 cases in this dataset) using 151 predictor variables. The same predictor variables are used for both the adaptive sampling and random sampling. The sizes of the two test sets are different because there are much more positives in the sample generated by the adaptive sampling algorithm. So after we use the SMOTE algorithm to increase the number of positives, the number of added synthetic positives is much larger for the case of the adaptive sampling. We also tested these models on another random sample of the data, so we have a direct comparison using the same test set. The result is shown in Table 2. From these results, we can see that when using random sampling, we got rather poor performance. It has a very low *recall* (meaning positive instances are hardly identified) and low *precision* (few predicted positives are true positives). The reason is that a random sample of the original data contains very few positive instances, so the resulting model is not generalizable to unseen test data. When using the adaptive sampling scheme, enough positive instances were collected and the predictive performance is satisfactory. Positive instances, which are of great value to businesses, were mostly identified correctly. And both *precision* and *recall* are remarkably higher. We built other models such as decision tree based on the sampled data and reached the same observation. The running time of adaptive sampling is data dependent. In this experimental setting, the adaptive sampling takes 10 times longer than the random sampling, because it needs to scan more data to collect enough positive instances. But it is the price that we have to pay to ensure a high quality sample so that the learned model is generalizable. Otherwise, the model would be of no use.

Table 2: Performance of models built with random sampling vs. adaptive sampling, on the same test set. The test data is obtained by randomly collecting 200k instances from the original dataset, without any post processing (such as SMOTE). So these numbers reflect the predictive performance on the original raw data. The resulting *precision* based on the adaptive sampling is not super high, which is not surprising for this highly imbalanced data. Because the number of negatives is overwhelmingly large, the absolute number of false positives is much bigger than the number of positives, although most negatives were correctly classified. Therefore $\frac{TP}{TP+FP}$ is relatively low. Note the *precision* is still much better than that of random sampling, and the difference in *recall* is more significant.

	Random sampling		Adaptive sampling	
	Predicted negative	Predicted positive	Predicted negative	Predicted positive
True negative	199589	242	198910	921
True positive	160	9	33	136
Recall	5.32%		80.47%	
Precision	3.59%		12.87%	

5. Conclusion

We have developed an adaptive sampling scheme for learning in severely imbalanced data, which frequently appears in a variety of applications, e.g., digital website traffic analytics and financial fraud detection. The adaptive sampling algorithm naturally handles both the intrinsic and extrinsic imbalance problem. It adapts to the distribution of the data and does not need to know the sizes and distributions of the minority class and the majority class *a priori*. When the data is less imbalanced, it will quickly return a balanced data sample so a predictive model can be built promptly, accordingly the system is super responsive. When the data is severely imbalanced, it will do a more thorough scan to collect sufficient minority instances. In addition, the reservoir sampling of negative instances ensures that we will get a more randomly distributed negative instances even if the original distribution is not *i.i.d.*. As a result, predictive models built base on it will be more generalizable to unseen data. The proposed sampling module enables reliable predictive analytic capabilities, which could not be achieved using random sampling. Extensive experiments have verified that it effective solves the imbalance problem on large scale data.

References

Rok Blagus and Lara Lusa. Evaluation of smote for high-dimensional class-imbalanced microarray data. In *Machine learning and applications (icmla), 2012 11th international conference on*, volume 2, pages 89–94. IEEE, 2012.

Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.

Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16: 321–357, 2002.

Frank E Harrell, Kerry L Lee, Robert M Califf, David B Pryor, and Robert A Rosati. Regression modelling strategies for improved prognostic prediction. *Statistics in medicine*, 3(2):143–152, 1984.

Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284, 2009.

Miroslav Kubat, Stan Matwin, et al. Addressing the curse of imbalanced training sets: one-sided selection. pages 179–186, 1997.

Charles X Ling and Chenghui Li. Data mining for direct marketing: Problems and solutions. 1998.

Sangyoon Oh, Min Su Lee, and Byoung-Tak Zhang. Ensemble learning with active example selection for imbalanced biomedical data classification. *IEEE/ACM transactions on computational biology and bioinformatics*, 8(2):316–325, 2011.

Clifton Phua, Damminda Alahakoon, and Vincent Lee. Minority report in fraud detection: classification of skewed data. *Acm sigkdd explorations newsletter*, 6(1):50–59, 2004.

Ewout W Steyerberg and Marinus JC Eijkemans. Prognostic modeling with logistic regression analysis. *network*, 10:11.

Jeffrey S Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*, 11(1):37–57, 1985.

Byron C Wallace, Kevin Small, Carla E Brodley, and Thomas A Trikalinos. Class imbalance, redux. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pages 754–763. IEEE, 2011.