Considering Privacy in Predictive Modeling Applications

Extended Abstract

Troy Raeder
Dstillery
37 E 18th St
New York, NY, USA
troy@dstillery.com

Brian Dalessandro
Dstillery
37 E 18th St
New York, NY, USA
briand@dstillery.com

Claudia Perlich
Dstillery
37 E 18th St
New York, NY, USA
claudia@dstillery.com

ABSTRACT

Large-scale data applications are increasingly a part of daily life. For example, the GPS in your phone that can tell you the fastest way to the airport incorporating real-time traffic data, Netflix suggests movies based on your entire viewing history, and the spam filters in email software learn individual spam preferences. Some of these applications rely on methodologies that are more 'data hungry' than others. Predictive modeling based on fine-grained data, which powers many of these applications, often requires a great deal of *data*, but relatively little understanding of any individual data point. In this work, we examine some simple modeling decisions that can make predictive modeling more privacy friendly without jeopardizing its performance and ultimate value.

Categories and Subject Descriptors

I.5.5 [**Information Systems**]: Pattern Recognition—*Implementation*

General Terms

Security, Human Factors

Keywords

Privacy, Machine Learning, Feature Hashing

1. INTRODUCTION

A variety of modern technologies rely on data - lots of it. Increasingly, the data being used across many industrial applications is generated by, and aptly describes the behavior, of people. We leave a digital trail of bits and pieces behind. Many of these data streams can deliver a lot of social good. The location profiles emitted by a GPS enabled smart phone can be use to recognize traffic jams, from which some analytical service might provide suggestions for alternate driving routes. Additionally, the movies you have watched and the products you have purchased are frequently used to better tailor future consumption experiences.

One of the most effective analytic solutions for data-driven decision-making is predictive modeling. Applications include recommender systems for movies and products, targeted advertising, fraud detection and many others. In order to build such predictive models, one needs to collect data in a very specific format, with both the outcome that is being predicted (e.g. "is this email spam or not") and the information available at prediction-time (such as for spam detection, the sender, recipient, subject, and body of the email). Standard predictive modeling systems have a few requirements for data collection that are critical to predictive success:

- The system needs to link the outcomes to the predictive information (features): If I know that 10 of your messages were spam but not which ones, I cannot build a model.
- Having more examples to learn from is better: With only few emails, the model cannot identify the word patterns that are indicative of spam. In fact the rarer the outcome, the more examples that will be needed.
- More *granular* information generally leads to better model performance: Knowing that you watched "Harry Potter and the Sorcerer's Stone" is much more valuable for recommending the newest Harry Potter movie than just knowing that you watched a kids movie.
- There is a close relationship between granularity and number of observations. Prior research [2] has shown a significant benefit to having large amounts of fine-grained data for prediction.

Several of these requirements have direct privacy implications. The ability to connect outcomes and features often necessitates some form of persistent user tracking. Most applications build up feature and outcome data over time (for example, past purchases), so it is necessary to keep a running history associated to a particular individual. On the Internet, the most common solutions are login-based and cookie-based tracking. The evidence that more examples indeed lead to better performance suggests a need for massive data collection and retention, but the length of time for which data is stored is a common privacy concern. The requirement of granularity can also be concerning as it makes re-identification notably easier. There may be many people who have seen 7 kids movies and 14 action movies, but very few will have watched my exact set of 21 movies. Even the

KDD '14 New York, NY, USA

famed Netflix prize fell under controversy when researchers were able to re-identify individuals by name after comparing movie's rated on Netflix to profiles from other publicly available movie rating websites.[1]

Every organization that stores and analyzes granular user data should make it a priority to design and build their systems with privacy rights in mind. The above issues are not new, but they are reminders that even those firms with the best intentions may find themselves on the unfriendly side of protecting personal data rights. In the interest of balancing personal privacy rights with successful predictive modeling this abstract explores recent developments in large-scale machine learning and their impacts on user privacy. Through the lens of operating a large scale machine learning system for behavioral ad targeting, we provide practical and empirical insight into what machine-learning systems do *not* need in order to be successful. The remainder of this paper discusses these developments and gives strategies we have employed in the domain of display advertising to mitigate the privacy concerns listed above.

2. DISPLAY ADVERTISING

At Dstillery, we run an automated online display ad targeting platform. We have described the system in much greater detail in prior publications [6], so we give only a brief overview here. Our customers are consumer brands; they pay us to first identify Internet browsers with some likely intent to purchase their product and then target them with display ads. Specifically, we do not share the data but actually execute the campaign through the programmatic buying environments [3]. Typically, customers evaluate us on some sort of *post-view conversion rate*, which is the rate that users convert, without the need for a click, *after seeing an ad*. What exactly counts as a "conversion" differs by campaign, but usually it requires buying something from the customer's web site.

We target browsers mainly based on partial histories of they web use. The term *browser* here refers to an individual browser cookie. If a user deletes his or her cookies or uses a different computer, the user becomes a completely different browser from the point-of-view of our system. For each of our campaigns, we place pixels on the customer's website. These pixels call back to `dstillery.com` and allow us to track any user who visits the customer's site. Additionally, we have partnerships with data providers, which give us a snapshot of the browser's general web browsing activity.

Our main classification models compute brand-propensity scores based on the user's browsing activity. If many of the URLs in a browser's cookie correlate positively with conversion, the browser gets a high score, and if the user's browsing history correlates negatively with conversion, he or she gets a very low score. The targetable audience for a given campaign is some number of the highest-scoring browsers, with the exact number depending on the campaign's budget.

Under these constraints (i.e., that our system needs both a URL history and a conversion history), respect for user privacy ultimately rests on data management strategies. Since we cannot collect *less* data (and maintain the minimum performance that generates customer value), we need to be conscientious about how we store, and how we use the data that we do collect. The remainder of this section outlines data management strategies that we find to be sound for both performance and privacy.

[1] `http://www.wired.com/2009/12/netflix-privacy-lawsuit/`

2.1 Online Learning with Feature Hashing

Our primary classification algorithm is Logistic Regression, trained incrementally using stochastic gradient descent [7]. Incremental updates have a number of benefits in our application. In addition to relaxing the requirement that training data sets fit in memory, incremental training allows us to begin making intelligent targeting decisions immediately, mere hours after the start of a new campaign.

Rather than store a giant map from URLs to model indices (since a feature's index in the feature vector needs to be consistent in order for incremental updates to work), we simply hash the URL into a very-high-dimensional space and use this hash as the index. This approach is called *feature hashing* [9] and is used in a number of large-scale applications [1].

Multiple variants of feature hashing have been proposed [8, 9], with later variants having better theoretical properties. The three general strategies are:

- Simply choose a dimensionality k and hash the original feature name into an hashed index space. If a hash collision occurs within the same instance, increment the value of the feature by 1.
- Execute the above strategy, but replicated it with d separate hash functions.
- Execute the above strategies, and when a collision occurs within the same instance, increment the value of the feature by a second hash function that takes values of either -1 or 1.

From a privacy perspective, online learning with feature hashing has very attractive properties. In addition to eliminating the feature map, we no longer need to explicitly enumerate features *at all*. We can learn models that perform well without even knowing what the final feature set actually is, or knowing the set of websites that any individual browser has actually visited. This is not a perfectly secure system by any means. We or anyone else could hash the names of the most popular websites, for example, and know with high probability (barring collisions) whether a particular cookie contains those sites. Regardless, feature hashing greatly reduces the level of detail at which data is stored and reduces the risk to the end user.

Our experiments have validated that we can implement feature hashing with only a negligible impact on performance. Table 1 shows the average Area Under the Receiver Operator Curve and average Lift at 5% for several variants of the feature hashing design across a canonical set of campaign datasets. For this analysis we varied the dimensionality of the hash space, varied the number of hash replications used and varied whether or not we implemented the second binary hash, as in [9]. We can see in this table that several of the design variants result in minimal change in performance. Changing the size of the hash space has the most significant effect on performance. Our original feature space has dimensionality around ten million URLs. When we use a hash space of similar dimensionality we essentially

Hash	Hash Dim	Has Coinflip	Hash Replications	Avg AUC	Avg Lift at 5%
No	NA	NA	NA	0.728	8.53
Yes	5E+02	Yes	1	0.656	5.17
Yes	5E+03	Yes	1	0.708	7.75
Yes	5E+04	Yes	1	0.713	7.90
Yes	5E+05	Yes	1	0.715	7.94
Yes	5E+06	Yes	1	0.723	8.36
Yes	5E+07	Yes	1	0.728	8.53
Yes	5E+07	No	1	0.727	8.48
Yes	5E+07	Yes	2	0.725	8.53
Yes	5E+07	Yes	5	0.723	8.49
Yes	5E+07	Yes	10	0.722	8.48

Table 1: Comparison of model performance under different feature hashing approaches.

get no dilution in performance for any of the hashing strategies explored. We only see a degradation when we reduce the feature space significantly.

Compared to the baseline of no feature hashing, we can achieve both better system maintenance and privacy friendliness with no customer impact. This is an ideal scenario and highlights how a methodology that was invented to improve the scalability of predictive modeling on Big Data has a secondary benefit of being more consumer friendly. A predictive model doesn't need to contextualize the data it learns from - it only requires consistency. Feature hashing provides such a mechanism.

2.2 Incremental Learning

While feature hashing reduces the amount of *context* we need to store, incremental training reduces the amount of data we need to store in the first place. Like with most predictive applications, our targeting models perform best when trained on more data sampled over longer time periods. Training models on large batches of data, say 60 days, requires storing all 60 days of data on disk. Incremental training requires only that we store one batch worth of data for each model, where the size of a batch is completely configurable. One position in the privacy conversation has been "the right to be forgotten". In an incremental learning setting, the model acts as an aggregation of all the previously seen data. It captures the state of knowledge and allows to maintain high predictive performance with minimal requirements on data retention.

In spite of the reduced granularity and size of the training data, our one-day incremental logistic regression model still performs well and outperforms its predecessor at Dstillery [4], a batch-trained Naive Bayes-like algorithm without feature hashing. Figure 1 shows median lift over random targeting (over the course of a month) on a consistent set of campaigns for both the batch and incremental models. The incremental logistic regression consistently performs about 20% better than the batch Naive Bayes model.

2.3 Anonymous vs Semantic Data

In many popular applications involving large sparse data, features can be annotated and grouped by some sort of semantic similarity. Words can be grouped by topics or parts of speech, movies can be grouped by genre or cast, and websites can be grouped by topic or textural similarity, just as examples. It is natural to assume that knowing semantic information about a word, movie, or URL would be helpful in prediction problems. Within our domain, there are two different opportunities for semantically annotating our data:

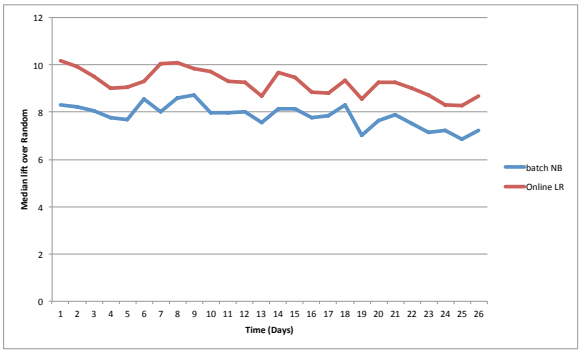

Figure 1: Performance comparison (measured as lift over random) between incremental logistic regression with feature hashing and a batch-trained Naive Bayes.

semantic analysis of web sites and demographic labeling of browser cookies.

Resources are available for us to both add context to our URL variables or attach demographic information to cookies, but we have found that neither approach significantly improves the predictive performance of our models. Figure 2 compares the size and performance (for one campaign) of segments built from our targeting models against commercially-available third-party behavioural and demographic segments. Our own "segments" in this case, are simply groups of cookies ordered by model score (i.e., top X% of the population according to the model, and then the next X% and so on). One can see from the figure that at a given scale, our own brand-specific segments consistently perform better than the closest available demographic segment.

In prior work [5], we evaluated both open-source and commercial categorization methods for URLs for the purpose of dimensionality reduction and found that they generally did not perform as well as methods that were agnostic to context. In separate experiments, we have repeatedly found that augmenting our high-dimensional feature set of raw URLs with contextual information from those URLs provides negligible benefit to our targeting algorithms. Taken together, our results suggest that in modern machine learning systems, a massive amount of anonymous data can be at least as predictive as more personally-identifiable data such as demographics and semantically-aware browsing behavior.

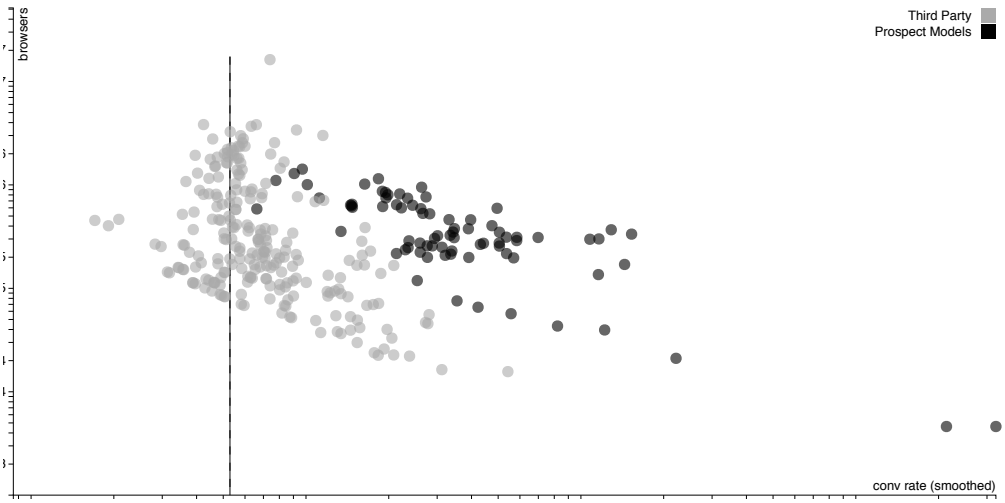

Figure 2: Performance and scale of logistic regression model segments (black) vs demographic segments (gray).

3. DISCUSSION

It is possible today to attach age, gender, interests and some reasonably accurate estimate of location to a large number of cookies. The ability to re-identify an anonymized cookie only increases with the number of descriptive features attached to it. Beyond cookies, this phenomenon applies to any "anonymized" data set, as we have learned from the lawsuits stemming from the release of the Netflix data. The methods we presented here may be used by firms to reduce the risk that the data they use or release may subvert a user's right to data privacy. Our experience at Dstillery shows that we can lean in this direction and not even consider it a sacrifice - our models perform quite well in the absence of the contextual or semantic labeling of data.

Like with any machine learning task, there is no 'one-size-fits-all' approach to improving privacy and ethically mining consumer data. Some predictive models, such as credit default models, are used to deny products or services to individuals who score poorly. Such models are legally prohibited from incorporating any variable, such as race, gender, or sexual orientation, that deals with membership in a protected class. In this case, feature hashing creates a black box that makes it more difficult to audit the models and ensure they abide by the appropriate regulations.

Every firm that engages in the collection, storage and usage of individual-level data for predictive modeling has the responsibility to balance respect for individual privacy and creating shareholder value. Often times this balance isn't quantifiable, but methodological transparency to both experts and consumers is a positive signal to all stakeholders that this balance is actively being sought.

4. REFERENCES

[1] A. Agarwal, O. Chapelle, M. Dudík, and J. Langford. A reliable effective terascale linear learning system. *arXiv preprint arXiv:1110.4198*, 2011.

[2] E. Junqué de Fortuny, D. Martens, and F. Provost. Predictive modeling with big data: Is bigger really better? *Big Data*, 1(4):215–226, 2013.

[3] C. Perlich, B. Dalessandro, R. Hook, O. Stitelman, T. Raeder, and F. Provost. Bid optimizing and inventory scoring in targeted online advertising. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 804–812. ACM, 2012.

[4] F. Provost, B. Dalessandro, R. Hook, X. Zhang, and A. Murray. Audience selection for on-line brand advertising: privacy-friendly social network targeting. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 707–716. ACM, 2009.

[5] T. Raeder, C. Perlich, B. Dalessandro, O. Stitelman, and F. Provost. Scalable supervised dimensionality reduction using clustering. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1213–1221. ACM, 2013.

[6] T. Raeder, O. Stitelman, B. Dalessandro, C. Perlich, and F. Provost. Design principles of massive, robust prediction systems. In *Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1357–1365. ACM, 2012.

[7] T. Schaul, S. Zhang, and Y. LeCun. No more pesky learning rates. *arXiv preprint arXiv:1206.1106*, 2012.

[8] Q. Shi, J. Petterson, G. Dror, J. Langford, A. Smola, and S. Vishwanathan. Hash kernels for structured data. *The Journal of Machine Learning Research*, 10:2615–2637, 2009.

[9] K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg. Feature hashing for large scale multitask learning. In *Proceedings of ICML*, pages 1113–1120. ACM, 2009.