**New York University - Stern School of Business**
**Working paper CBA-16-01**

# Matrix-Factorization-Based Dimensionality Reduction in the Predictive Modeling Process: A Design Science Perspective

Jessica Clark and Foster Provost

Department of Information, Operations and Management Sciences

NYU Stern School of Business

September 29, 2016

**Abstract**

Dimensionality Reduction (DR) is frequently employed in the predictive modeling process with the goal of improving the generalization performance of models. This paper takes a design science perspective on DR. We treat it as an important business analytics artifact and investigate its utility in the context of binary classification, with the goal of understanding its proper use and thus improving predictive modeling research and practice.

Despite DR's popularity, we show that many published studies fail to undertake the necessary comparison to establish that it actually improves performance. We then conduct an experimental comparison between binary classification with and without matrix-factorization-based DR as a preprocessing step on the features. In particular, we investigate DR in the context of supervised complexity control. These experiments utilize three classifiers and three matrix-factorization-based DR techniques, and measure performance on a total of 26 classification tasks.

We find that DR is generally not beneficial for binary classification. Specifically, the more difficult the problem, the more DR is able to improve performance (but it diminishes easier problems' performance). However, this relationship depends on complexity control: DR's benefit is actually eliminated completely when state-of-the-art methods are used for complexity control.

The wide variety of experimental conditions allows us to dig more deeply into when and why the different forms of complexity control are useful. We find that $L_2$-regularized logistic regression models trained on the original feature set have the best performance in general. The relative benefit provided by DR is increased when using a classifier that incorporates feature selection; unfortunately, the performance of these models, even with DR, is lower in general. We compare three matrix-factorization-based DR algorithms and find that none does better than using the full feature set, but of the three, SVD has the best performance.

The results in this paper should be broadly useful for researchers and industry practitioners who work in applied data science. In particular, they emphasize the design science principle that adding design elements to the predictive modeling process should be done with attention to whether they add value.

# 1 Introduction

This is a design science paper (Hevner et al. 2004) which deeply explores the role of matrix-factorization-based Dimensionality Reduction (DR) in the predictive modeling process. Via a review of literature where DR is utilized for predictive modeling, we contextualize DR as a technological artifact worthy of detailed analysis: it is popular but seems to be somewhat misunderstood, and therefore is sometimes used incorrectly. The intended audience for this research is not solely "big data" researchers or people developing new DR techniques (although our findings are certainly relevant to that community), but more broadly applied data science researchers and industry practitioners who leverage predictive modeling.

In the business analytics literature on predictive modeling, dimensionality reduction is widely accepted as a beneficial step in preprocessing and exploring the feature space (however, note that such use is not restricted to business analytics research; we discuss two examples below from social media analysis and web security). Shmueli and Koppius (2010) provide a guide to utilizing predictive modeling in IS research. DR is a prescribed part of their process because, "reducing the dimension can help reduce sampling variance (even at the cost of increasing bias), and in turn increase predictive accuracy." The authors subsequently codify dimensionality reduction as one of their steps for building a predictive model; however, they do not specify when one should or should not apply it. Consider another article in *Management Science*: "Given the large number of potential demographics available, data dimension reduction is an important factor in building a predictive model that is easy to interpret, cost effective, and generalizes well" (Kim et al. 2005).

These two quotes summarize the main reasons to use DR in predictive modeling. Most importantly for our purposes, it can improve predictive performance by reducing error due to variance (for now, we will set aside other benefits, such as interpretability and reduced processing time). The two overlapping alternatives to matrix-factorization-based DR that accomplish the same goal (and are frequently used in conjunction with it) are supervised regularization and feature selection. We will investigate the effect of DR in the context of these methods of complexity control as well as use them as a basis for comparison. That is to say, the baseline models we will compare against will not be "vanilla" (unregularized) models, since these are not commonly used. We will be comparing against models that leverage some form of supervised complexity control, as well as investigating

the interactions between DR and various forms of regularization.

Despite DR's popularity, there is some evidence that there is a lack of clarity when deciding to apply it or not. Consider the intriguing recent paper by Kosinski et al. in PNAS (Kosinski et al. 2013). The authors demonstrate that it is possible to predict important and sensitive personal traits of individuals using only the items that they have Liked on Facebook. Users can indicate support for a wide variety of content items, including media, brands, Facebook content, people, and statements by Liking them. A key part of the authors' methodology employs the process this paper studies: conduct DR (via SVD) on the user-Like matrix before building logistic regression models using the resulting reduced-dimensional features to predict the desired traits.

Consider as a second example a paper published in the most recent proceedings of the top data science conference: it uses both website text and network characteristics to identify sites that sell counterfeit and black-market goods such as pharmaceuticals (Der et al. 2014). Again, they apply exactly the procedure we study (and that is prescribed in the aforementioned papers). The original feature count for describing the websites is about 65,000; after applying DR the authors find that most of the variance in the features can be accounted for using only 72 of the reduced dimensions. They then build predictive models on the resulting reduced-dimension space to classify sites, reporting high precision and recall.

Both of these data sets initially might appear to be ideal candidates for DR: the dimensionality is very high, the instances are largely sparse, and the data intuitively seem like they ought to contain latent information that could be captured via dimensionality reduction; therefore, there would seem to be a large potential benefit to reducing the sampling variance. However, neither of these papers assesses whether using DR actually improves the predictive performance; indeed, for all we know, the predictive performance might have been better without the added complexity of the DR. A natural question is whether the same (or better) performance could have been achieved with supervised regularization alone, which would thus simplify the predictive modeling process.

As the process described in our examples—building a predictive model on the dimensions resulting from DR via matrix factorization—has become common both in prescriptions in the literature and in predictive modeling practice, it is important to examine it closely from a design science perspective. Thus, our main contribution is an experimental comparison in the context of state-of-the-art complexity control techniques, across a suite of 26 real-world binary classification tasks,

using three different matrix-factorization-based DR methods (Singular Value Decomposition, Non-negative Matrix Factorization, and Latent Dirichlet Allocation) and three classification algorithms ($L_2$-regularized logistic regression, $L_1$-regularized logistic regression, and Random Forests).

This comparison broadly shows that DR tends to hurt more than help predictive performance for binary classification problems. Therefore, using DR without assessing whether it is adding value is a mistake—one that numerous studies published in top journals have made![1] The primary conclusion we draw from our results is a basic design science lesson: one *should* assess a method like DR that adds complexity to the predictive modeling process to ensure that it adds value. Additionally, the empirical study yields five specific conclusions about DR for predictive modeling:

1. The harder the binary classification problem, the more using DR will improve performance. For easier problems, DR tends to reduce performance versus the original features.

2. DR's advantage is eliminated in the context of problem-specific regularization (complexity control hyperparameters chosen using nested cross-validation). Thus, supervised regularization is more effective than MF-based DR for reducing predictive error due to variance via complexity control—we find that performing regularization carefully does not reduce predictive performance, even on easier problems.

3. DR has a greater relative advantage when used in conjunction with classifiers that leverage some form of internal feature selection; unfortunately, such classifiers tend to perform relatively poorly, even with DR.

4. The results are not limited to one form of matrix-factorization-based DR. Of the three, SVD is generally the best-performing, and so if it is necessary or desirable to utilize DR for other reasons, SVD is a solid default method to use.

5. Of the 12 feature set/classifier combinations in our experiments, $L_2$-regularized logistic regression models trained on the original full feature sets had the best performance. Therefore, full-feature $L_2$-regularized logistic regression with the regularization hyper parameter chosen via nested cross-validation is a strong baseline for use with this type of data.

[1] While the predictive performances in cited papers are very strong, and thus we are not calling into question the conclusions of their studies, they may be improved with the methodological change of not reducing the dimensionality of their feature space at all.

We mostly limit this experimental comparison to large, sparse data and binary classification tasks. Specifically, we focus on the very large, sparse behavioral data now common in business applications. Similarly to the motivating examples listed above, such data might seem to be ideally suited for DR's use. Further, the main focus of the paper is on binary classification (rather than numerical regression or multi-class classification). Since the primary contribution of the study is the lesson that one should carefully consider whether to apply DR (especially in the context of regularization), depth is more important than breadth. We are *not* claiming that DR will never help—but that we can show an important class of problems where it does not seem to and that researchers don't seem to understand that well.

Thus, the implications of this research to researchers and practitioners are clear. It is crucial to be careful when utilizing DR to improve predictive modeling performance; careful regularization is key and generally much more effective for complexity control. That is: the added complexity of the DR stage in the predictive modeling process may well be unnecessary for improving predictive performance and indeed may be reducing performance. Predictive modelers should take the tenets of design science seriously, keep in mind the principle of Occam's Razor, and perform lesion studies (Langley 2000) in order to ensure that analytics systems are exactly as complex as they need to be, and no more so. Introducing extraneous components will at best increase processing time, and at worst can result in worsened performance.

## 2 DR's Role in Predictive Modeling

Let us clarify what we mean in this paper by predictive modeling. We specifically refer to using models induced from data to predict a predefined target variable.[2] The relative success of these models is assessed by computing some measure of generalization performance. Thus, as opposed to models whose purpose is to estimate the degree to which various covariates cause changes in the target variable, here we will focus solely on doing a good job of predicting the value of the target. Predictive modeling is important both for research and for practice. The aforementioned study by Shmueli and Koppius explains the variety of ways in which predictive modeling can be valuable for scientific research. For many business applications, it's useful to have an accurate prediction

[2] *Predict* here can refer to an estimation of the value of any variable for which we don't know the value, not just events that may happen in the future (Provost and Fawcett 2013).

for some phenomenon in order to take actions more cost-effectively. As mentioned above, DR is important in predictive modeling as a potential method for improving predictive performance.

There are two ways of accomplishing DR (Guyon and Elisseeff 2003, Liu and Motoda 1998): feature selection and feature engineering. Feature selection involves choosing existing features based on some criteria: examples include sequential forward selection, selecting the top information-gain features, and modeling techniques that incorporate feature selection internally, such as L1-regularization or tree-structured models (Guyon and Elisseeff 2003). Feature engineering in this context means creating new features using the existing ones. This can be done by modifying or combining the existing features, or by employing a dimensionality reduction algorithms: matrix-factorization-based techniques such as SVD or NMF (Shahnaz et al. 2006), nonlinear techniques such as LLE (Roweis and Saul 2000) and Isomap (Tenenbaum et al. 2000), and clustering-based techniques techniques (Dhillon et al. 2003). In this paper, we focus chiefly on Matrix Factorization (MF) techniques such as SVD.

Ostensibly, there should be no theoretical reason to focus on one form of DR. There have been numerous reviews and comparisons of DR techniques (Guyon and Elisseeff 2003, Liu and Motoda 1998), including some that focus chiefly on feature engineering techniques (Van der Maaten et al. 2009) and others that focus on feature selection methods (Blum and Langley 1997, Saeys et al. 2007, Forman 2003). Why, then, do we limit our analysis to an exploration of MF-based feature engineering techniques in particular? First, the analysis in this paper situates MF within the predictive modeling process, which frequently also directly incorporates some form of supervised complexity control such as $L_1$ or $L_2$-based regularization. Therefore, we treat MF-based DR as a separate component to be evaluated in the context of different types of regularization. That is, there is no need to choose between feature selection and DR for complexity control and it is common to do both (Kosinski et al. 2013).

Second, MF has a long history of use in predictive modeling by industry and research practitioners across many domains, including text classification (Yang 1995), facial image recognition (Turk and Pentland 1991), network security (Xu and Wang 2005), neuroscience (López et al. 2011), and social network analysis (Kosinski et al. 2013). Crucially for our intended audience of applied data science researchers and industry practitioners, popular software packages such as Python and MatLab contain off-the-shelf implementations of MF techniques for large, sparse data sets that

can be seamlessly integrated within the predictive modeling framework. This is not necessarily the case for other classes of techniques. Further, MF techniques have historically been highly effective in slightly different but related contexts such as collaborative filtering, information retrieval, and image processing. Therefore, there may be some confusion about the extent to which they are useful in any given predictive modeling context.

### 2.1 Matrix-Factorization-Based DR: General Background and Notation

Assume that we have $n$ data instances, each composed of the value for a binary target variable that we would like to predict, as well as the values for $d$ attributes arranged in a feature vector. Each instance $u$ has a positive association with at least one feature $i$. Assume that this association is binary. Therefore, the data can be represented as an $n \times d$ binary matrix $A$ such that $A_{ui} = 1$ when $u$ is associated with $i$ and 0 otherwise, plus a target vector $\mathbf{y}$ where $y_u \in \{0, 1\}$.

To give a motivating example, an instance might be a Facebook user where each attribute represents one item that she might Like. Facebook Likes are defined in the aforementioned study by Kosinski et al. as "a mechanism used by Facebook users to express their positive association with (or "Like") online content, such as photos, friends' status updates, Facebook pages of products, sports, musicians, books, restaurants, or popular Web sites." We can represent each user's feature vector as being 1 where the user has Liked the corresponding item and 0 otherwise. We want to predict demographic traits (such as gender) or psychographic traits (such as intelligence) using this Like information. To predict, say, gender, $A$ and $\mathbf{y}$ would thus be defined as

$$A_{ui} = \begin{cases} 1 & \text{if } u \text{ Liked } i \\ 0 & \text{else} \end{cases}, \mathbf{y}_u = \begin{cases} 1 & \text{if } u \text{ is female} \\ 0 & \text{else} \end{cases}$$

Assume that we utilize a MF-based technique to reduce the dimensionality of $A$. The resulting factorization is thus

$$A \approx L_k R_k \tag{1}$$

The $k$ columns of $L_k$ can be viewed as "latent factors" underlying the original data. Each instance will have a representation in the new $k$-dimensional space captured by $L_k$. The corresponding

rows of the $k \times d$ matrix $R_k$ are sometimes referred to as the "loadings" of the components, representing how strongly each data instance is associated with each latent factor. Once the matrix factorization has been computed, the resulting latent factors may be used as features in a predictive model. Section 4.2 discusses the specific MF techniques we use and their implementations.

# 3 Literature Review: Applications of DR for Predictive Modeling

Practitioners and researchers should understand the proper theoretical or empirical rationale behind why they are choosing to utilize a particular modeling technique, including DR, for a particular modeling application. In this section, we provide the context behind DR's use in the predictive modeling process, and establish that it is both commonly used and perhaps misunderstood. We find that there is a substantial amount of predictive modeling research which employs DR absent any explicit rationale (theoretical or empirical) for doing so. The empirical results in Section 5 further support the necessity for establishing such a rationale and for careful comparison.

## 3.1 Why Dimensionality Reduction?

Dimensionality reduction, in the very broadest sense, can be defined as a class of methods which reduce the size of the feature set used for modeling. In this paper we will consider dimensionality reduction to be beneficial if a predictive model built using the reduced space as features has superior predictive performance to a model built using the original features. The literature provides a strong theoretical basis for why we think this might happen.

Famously, dimensionality reduction can be employed to overcome the "Curse of Dimensionality" (Bellman 1961), which can manifest itself in numerous ways. In particular, traditional statistical modeling techniques break down in high-dimensional spaces. Techniques which depend on measuring similarity of instances fail since no two instances appear similar in high-dimensional space (Domingos 2012). Some algorithms have been developed that overcome this "Curse" (Gionis et al. 1999); however, DR can also be applied to the data. To avoid overfitting in these cases, this reasoning is commonly employed in papers that use such data.[3]

The second, related theoretical foundation for why DR may help in predictive modeling draws

[3]See (López et al. 2011), (West et al. 2001), and (Burl et al. 1994) for a few examples.

on the bias/variance tradeoff in predictive error (Friedman 1997). That is, that by reducing dimensionality, we reduce training variance at the cost of introducing some bias. The hope is that the overall error will be diminished (Shmueli and Koppius 2011). DR and other forms of complexity control improve predictive performance by trying to exploit this tradeoff (Friedman 1997, Scharf 1991, Domingos 2012, Kim et al. 2005). Feature engineering more generally may also try to reduce bias, usually at the expense of increased variance.

There are other reasons to compute dimensionality reduction besides improving predictive performance. While we do not consider accomplishing these tasks within the scope of this paper, they are well-explored in the literature and worth mentioning in the service of understanding dimensionality reduction as a design artifact. They are also frequently mentioned in papers where the main goal is predictive accuracy, whether as a side benefit or an additional objective. Delineating among these reasons is crucial since it may be necessary to trade off among goals. Being knowledgeable and cautious about these tradeoffs will hopefully lead to better use of DR.

DR can be used as a tool to compress large data sets, whether to save space or time (Thorleuchter et al. 2012); however, these methods are time consuming to compute and generally result very dense matrices (which, even with reduced dimensions, are likely to be larger in terms of storage space than the original feature matrices, if they are very sparse). Methods such as feature hashing (Shi et al. 2009) and random projections (Achlioptas 2001) provide ways of randomly selecting or combining the original data. They are designed to reduce the size of data sets, are fast to compute, and sometimes come with theoretical bounds on the amount of information lost due to compression (Shi et al. 2009, Achlioptas 2001), but are generally not employed to improve predictive accuracy. A final reason to compute DR is to detect interpretable latent structure in data. This can be important if it is necessary to explain the features in the model, perhaps to a domain expert or manager. Additionally, the latent factors can be an end goal themselves. Thus, DR can be seen as a way of preprocessing the data to visually inspect it, as described in Westad et al. (2003).

### 3.2 (Mixed) Empirical Evidence from Predictive Modeling

As we will see in the papers discussed below, there is the perception that doing dimensionality reduction is generally a good idea for predictive modeling. Matrix factorization is very popular and successful in some fields closely related to predictive modeling, especially for large sparse

data. In particular, DR is especially commonly used in image processing (Turk and Pentland 1991), information retrieval (Deerwester et al. 1990), and collaborative filtering (Adomavicius and Tuzhilin 2005). These three fields share a commonality with the business analytics problems that we are particularly interested in: they all use high-dimensional data sets with individual features that might not contribute much information individually. Thus, the perception that DR is helpful for predictive modeling may be based in part on its success in those fields; however, these settings are not identical to predictive modeling.

There have been some papers in which a model built on a reduced-dimensionality feature set has been shown to achieve better generalization performance than a model built using the original features. Several of these papers have the objective of demonstrating the usefulness of one or more particular dimensionality reduction methods. Specifically, the dimensionality reduction is introduced: to reduce the noisiness of data (Ahn et al. 2007); to uncover latent meaning in the data (Whitman 2003, Blei et al. 2003); to overcome poor features (as in images) (Subasi and Ismail Gursoy 2010); to combat polysemy and synonymy, as in information retrieval (Karypis and Han 2000); or to speed up the modeling (Fruergaard et al. 2013). Other papers mention that a comparison has been undertaken but do not explicitly state the results (West et al. 2001).

Note, however, that none of these papers mentions that the predictive modeling used state-of-the-art methods for complexity control, as discussed above in Section 1. Without good complexity control, improved generalization performance may simply be due to the dimensionality reduction acting as a regularization mechanism—and being compared to poorly regularized modeling. We explore this further below in Section 5.

On the other hand, there are papers that demonstrate dimensionality reduction for predictive modeling performing worse than not using DR, such as Raeder et al. (2013) and Xu and Wang (2005). Others report that DR resulted in mediocre performance (Guyon et al. 2009). Most tellingly, however, is the aforementioned survey by van der Maaten et al. of linear and nonlinear dimensionality reduction techniques. This survey utilized both "natural" and "synthetic" data sets.[4] They found that on the majority of their "natural" datasets, classification error was not improved by doing any form of dimensionality reduction on the feature space. However, they hedge

[4]Their results differ from ours, below, in that (i) they do not use sparse behavioral data, (ii) their datasets are limited in size to fewer than 10,000 instances, and (iii) they do not use state-of-the-art methods to choose the size $k$ of the reduced space, a key limitation.

their bets by explaining this as likely being due to an incorrect choice of $k$.

Beyond this mixed evidence, we find many papers that apply dimensionality reduction to their feature sets without explicitly mentioning a comparison to models built on the original feature sets. It is important to note that there are reasons not to just do DR automatically, even ignoring the effect on predictive performance. The various methods for DR are very time consuming and computationally expensive. It also can obscure interpretability of the original features (latent factors are not guaranteed to be interpretable!). While some of these papers do give rationale for using DR, it is not certain that they are building the best models that they can (and therefore obtaining the strongest results possible) by ignoring this comparison.

While these latter papers do not mention an empirical comparison between DR and no-DR feature sets, some do provide a rationale. The reasons are generally the same that we have encountered before: to reduce the noise in the data (Hu et al. 2007); to make the dimensionality of the space more tractable (Thorleuchter et al. 2012, Coussement and Van den Poel 2008); or to reduce the computational expense of modeling (Tremblay et al. 2009). Sometimes DR is employed in the context of "side benefits" such as aiding in interpretation of a model (Khan et al. 2007) or speeding up prediction time (Lifshits and Nowotka 2007). Others do not provide either empirical or theoretical justification for applying DR to their feature sets (Kosinski et al. 2013, Cai et al. 2013, Der et al. 2014, Arulogun et al. 2012).

# 4 Experimental Test-Bed

So far, we have shown that there is mixed evidence in the literature regarding the efficacy of dimensionality reduction for improving performance in predictive modeling. We have also shown that it is a reasonably popular data pre-processing technique, to the extent that numerous papers utilize it without any empirical justification. Here, we carry out an empirical comparison on one collection of binary classification problems based on sparse behavioral data, found in the aforementioned paper by Kosinski et al. The thrust of their research is that private traits *can* be predicted better than at random using Facebook Likes. Our intention here is not to critique these existing (strong) results, but to use the domain as a testbed of multiple, real predictive modeling problems using the sort of sparse behavioral data that is increasingly seen in business analytics applications. In the

process, we demonstrate that the results can be strengthened by not employing DR at all, as DR broadly tends to reduce predictive performance. Additionally, for research where results depend on prediction quality, the implication is that caution should be exercised when considering applying dimensionality reduction, since it may decrease predictive performance.

### 4.1 The Data

We recap the details of data collection from the Kosinski et al. paper. They collect data via a Facebook application that allows users to take personality tests and also makes the users' Likes and profile information available to the researchers. Thus, they can formulate a user-Like matrix $A$ such that $A_{ij} = 1$ if user $i$ Liked item $j$. They have generously made the resulting user-Like matrix as well as the personality test results and anonymized profile information available to researchers.

In their paper, the user-Like matrix includes all users who have Liked at least 20 items and all Likes that have been Liked by at least 2 users, yielding $n = 58,466$ users and $d = 55,814$ Likes. The data have been incrementally updated since publication of their results. Therefore, our data have larger $n$ and $d$ than what is in their paper, specifically $n = 211,018$ users and $d = 179,605$ Likes. The corresponding matrix contains roughly 35 million unique user-Like pairs, resulting in a matrix with 99.91% sparsity (meaning .09% of the entries in the matrix are filled). The average user Liked 195 items and the average Like was Liked 166 times.[5]

We replicate the authors' predictions for all of the binary target variables. These include "single vs. in relationship", "parents together at 21", "smokes cigarettes", "drinks alcohol", "uses drugs", "Caucasian vs. African American", "Christianity vs. Islam", "Democrat vs. Republican", "gay", "lesbian", and "gender". Counts for the number of labeled users and Likes for each binary target can be found in Table 1 of Appendix A. We followed the labeling procedure undertaken by Kosinski et al. for all target variables except "Caucasian vs. African American," which was labeled by visual inspection of profile pictures not available to us; we used values from a different survey item.

Additionally, we included predictions for the numerical variables that they report. In order to provide a consistent comparison, we binarized all variables by setting a threshold, above which the target was set to 1, and 0 otherwise. These numerical variables are "satisfaction with life", "intelligence", "emotional stability", "agreeableness", "extraversion", "conscientiousness", "openness",

[5]We store this resulting matrix $A$ as a SciPy sparse matrix (Jones et al. 2001–) in Python.

"density of friendship network", "number of Facebook friends", and "age". Some of the thresholds for binarization were set based on an intuitive value, such as age $\geq 30$; others were set to be the median or top-quartile value for that variable to ensure variety in base rates. More information on the various numerical variables and their thresholds can be found in Table 2 of Appendix A.

### 4.2 Dimensionality Reduction Methods

Kosinski et al. reduce the dimensionality of the user-Like matrix using Singular Value Decomposition (SVD), a massively popular dimensionality reduction method which is widely available in commonly-used data mining software packages such as Python, MatLab, and R. To the extent that dimensionality reduction is used in Information Systems research, this is usually the method of choice (Kim et al. 2005). It has also been used for dimensionality reduction across many other domains, including text classification (Yang 1995), facial image recognition (Turk and Pentland 1991), network security (Xu and Wang 2005), and neuroscience (López et al. 2011).[6]

SVD is popular for a reason: it has highly desirable properties. The SVD components (sometimes referred to as latent factors) are usually sorted in decreasing order of the amount of sample variance that they account for in the original data matrix. In many practical applications, most of the sample variance is accounted for by the first few singular vectors; therefore, it is common to choose some $k << n, d$ and compute the truncated SVD which consists of the first $k$ singular vectors. Importantly, the resulting truncated SVD matrices can be viewed as representing some latent structure in the data. Additionally, SVD is computed by optimizing a convex objective function; the solution is unique and equivalent to the eigenvectors of the data matrix. Thus, a further benefit of using SVD is that the decomposition can be computed once for the maximum desired $k$ and the resulting components selected in decreasing order of their value. Because of these advantages as well as SVD's popularity, the main experiments in this paper focus on SVD as the DR method.

However, SVD is far from the only DR method used by researchers or practitioners, even within the class of MF-based DR techniques. These methods do not necessarily come with the

[6] Van der Maaten et al. (2009) surveyed numerous linear and nonlinear dimensionality reduction techniques across a variety of classification problems and found that using principal components analysis (PCA) features as input yielded the best classification accuracy of all of the dimensionality reduction techniques (often in practice PCA and SVD are used interchangeably). The authors also found that with non-sparse, medium-sized data (up to 10,000 instance), not reducing the dimensionality of the features actually yielded the best performance in the majority of cases—a finding that we will explore in greater detail below.

same desirable guarantees as SVD, though. Other methods generally do not have a unique solution; therefore the resulting components may be different depending on how the optimization algorithm is initialized. Further, the number of components must be specified *a priori*: computing factorizations with different numbers of components will result in completely different components. Additionally, these components will not be sorted in any order of relevance; all components may account for equal variance in the data (or be equally relevant to the target variable). Finally, and crucially for the intended audience of this research, other methods are not as widely implemented in popular off-the-shelf software packages, especially for sparse data structures. Therefore, for this simple reason, they are far less useful to business analytics researchers and industry practitioners who simply want to utilize DR to improve predictive performance, rather than study DR itself.

Of course, every alternative DR method does come with special advantages, and some have been found to perform well in certain practical contexts (Xing and Girolami 2007, Shahnaz et al. 2006). To ensure that our results are not dependent on the choice of DR method, we also explore a few additional methods. Non-negative Matrix Factorization (NMF) requires that the resulting components must be non-negative. This constraint usually results in sparse (and therefore, interpretable) components, and was originally developed for image processing applications (Lee and Seung 1999). Latent Dirichlet Allocation (LDA) is a generative probabilistic model for matrix factorization that may capture a more realistic picture of the relationships in the data than SVD does. It was originally developed for text modeling (Blei et al. 2003). Both have shown promising predictive performance in certain contexts (Shahnaz et al. 2006, Bíró et al. 2008).

### 4.3 Dimensionality Reduction Parameters

We compute the SVD of the user-Like matrix using SciPy's sparse SVD package for Python. Similarly, we compute the NMF and LDA decompositions using Python's sklearn package with default settings. There are two slightly different options for these computations. Only a subset of the set of users have labels for any task. It is possible to filter $A$ separately for labeled users for each target variable, then compute the DR of the resulting sub-matrix of $A$. The other option is to compute the DR of the entire $A$ matrix, then filter the resulting factors for each target variable's labeled values. We use the second option, as it uses the maximum amount of data.

An important computational aspect of using DR in predictive modeling is choosing the optimal

value for $k$, the number of components that will be used as features in the model. There is substantial evidence that utility peaks with a certain number of dimensions.[7] In fact, "selection of number of dimensions" is mentioned as being important in almost every paper that utilizes dimensionality reduction for predictive modeling, which we discuss below. There is no *a priori* way of discovering the optimal number of dimensions. Furthermore, certain recent papers have indicated that for large-scale problems, the optimal number of dimensions may be very large, larger than previously considered. This is one area where large, sparse behavioral data may cause differences from what is traditionally understood in DR application.[8]

There are numerous ways for estimating the optimal value for $k$. First: use an operational criterion; that is, pick the $k$ that optimizes predictive performance on held-out data. Another way is to rank the singular values (elements of $S$) and pick the top few based on visual inspection, for instance, by looking for a "knee" in the singular values (Hoff 2007, Burl et al. 1994).[9] It is very important, however, to consider the troublesome statistical multiple comparisons problem (Jensen and Cohen 2000) that is inherent to choosing the "best" $k$ after the fact—similarly to how we would avoid choosing other modeling components by observing their performance on the testing data. One could choose $k$ by doing some sort of (nested) cross-validation. For instance, Owen and Perry (2009) show a method for holding out data, computing SVD on the non-held-out data, and selecting $k$ so as to minimize the reconstruction error between the held-out data and its SVD approximation. Finally, $k$ can be chosen such that the resulting latent dimensions account for a predetermined portion of the total variance in the data. This is done by Der et al. (2014), who select sufficient factors to account for 99% of the variance.

Our experiments include careful selection of this parameter. Kosinski et al. state that they utilize the top $k = 100$ components as features for most of the predictive models and $k = 30$ for some of the variables for which there is less data. Their selection is based on an operational criterion: visual inspection of predictive accuracy vs. number of components for a few target variables flattens out around 100. Since we have more labeled instances than they did we use $k = 100$ as our default choice for $k$ in our predictive models.

[7] For some examples, see Deegalla and Bostrom (2006), Hu et al. (2007), and Coussement and Van den Poel (2008).

[8] Examples where performance continues to improve past a small number for $k$ can be seen in Sarwar et al. (2000), Raeder et al. (2013), and (Koren et al. 2009).

[9] Burl et al. (1994) also use visual inspection of the latent factors themselves, given that the data are images.

The selection of the number of components is more computationally intensive for the NMF and LDA factorizations. Because there is not one unique factorization with the factors sorted in order of relative importance, selecting the number of components requires computing separate decompositions for every value for $k$.

### 4.4 Predictive Methods, Evaluation, and Complexity Control

Kosinski et al. use logistic regression with 10-fold cross-validation to predict the binary target variables, using the top $k = 100$ SVD components as inputs. They report the area under the ROC curve (AUC) as their measure of predictive performance, which is equivalent to the probability that their model will rank any positive instance higher than any negative instance. To replicate their results, we do the same, training and evaluating our models using scikit-learn's Logistic Regression package in Python (Pedregosa et al. 2011). We also train and evaluate models built on the original user-Like matrix $A$. To benchmark our results, we compared our SVD model results to the Kosinski et al. results. This comparison can be found in Appendix A, Table 1. The AUC's don't match up exactly, which is to be expected due to the updated data; however, they are very close. Since the goal of these experiments was to do a systematic comparison of SVD features to unreduced features, minor deviations from results presented in the original paper should not matter.

Training LR (and other types of models) with no regularization can result in overfitting (Provost and Fawcett 2013), and—importantly for this study—the degree of regularization can greatly impact predictive performance (Bishop et al. 2006). In practice, training a regularized LR model involves finding the set of weights $\mathbf{w}$ that minimizes:[10]

$$C \sum_{i=1}^{n} \log\left(1 + \exp\left(-y_i \mathbf{w}^T \mathbf{x}_i\right)\right) + ||\mathbf{w}|| \tag{2}$$

The second term in equation (2) is the regularization term: it penalizes the magnitude of a norm of the weight vector. $C$ is the regularization parameter, which controls the trade-off between minimizing the logistic loss and the amount of regularization. Thus, higher values of $C$ imply less

[10] Equation 2 is often written as

$$\sum_{i=1}^{n} \log\left(1 + \exp\left(-y_i \mathbf{w}^T \mathbf{x}_i\right)\right) + \lambda ||\mathbf{w}||,$$

where $\lambda$ is the regularization parameter; however, here we adopt the convention taken by scikit-learn.

regularization and lower values of $C$ yield more regularization. Either the $L_1$ or $L_2$ norm may be used. These two types of regularization are known as $L_1$ and $L_2$ regularization, respectively, and there is an important difference between the two: $L_1$ regularization tends to result in sparse coefficient vectors, with many of the coefficients set to zero. Thus, $L_1$ regularization functions as a kind of implicit supervised feature selection (Ng 2004). In fact, the two types of regularization result in such different results that we treat them separately in our experiments.

Kosinski et al. do not discuss regularization in their paper, but we assume their models use some form of complexity control since it is not standard practice to use unregularized logistic regression for binary classification. We begin by using $L_2$-regularization with $C = 1$ (the default value in scikit-learn) to validate the performance of our SVD models and for our first set of experiments. Again, since our experimental results match theirs, this seems to be a reasonable choice.

Careful regularization can dramatically change the results of a regression; therefore, we also experimented with using a state-of-the-art procedures for selecting the optimal parameter. Specifically, we performed nested cross-validation paired with a grid search to properly optimize and evaluate the effect of regularization on the performance. Nested cross-validation adds an inner loop of cross-validation to each of the folds of standard cross-validation. In the inner-loop we do 3-fold cross-validation to estimate the performance of each parameter combination in our grid. For our experiments using $L_2$ regularization, $C$ is chosen from $\{.01, .1, 1, 10\}$. In the outer loop, a new model is trained using the parameter combination that yielded the highest AUC from the inner loop and then is tested on the test data. Thus, the parameter selection is part of the training process and the test data are not compromised (Provost and Fawcett 2013).

Because of $k$'s importance to the success of DR, we perform additional experiments that go one step farther and treat $k$ as a model parameter to be selected using held-out data. In order to avoid the problem of multiple comparisons and treat the choice of $k$ as part of the training process, we (A) use nested-cross validation to select $k$ from $\{50, 100, 200, 500, 1000\}$ within each fold, while holding $C$ constant to the default setting. Then, for completeness, we (B) select $k$ and the regularization parameters in a huge round of nested cross-validation, which performs the full grid search over all combinations of values for $C$ and $k$ and selects the best combination based on three-fold cross validation within each of the ten outer folds.

$L_2$-regularized LR (which we will abbreviate as $L_2$-LR going forward) is only one among many

possible predictive algorithms. Because DR methods encompass both feature selection and feature engineering, it is instructive to investigate the performance of SVD and other MF techniques in the context of feature engineering algorithms as well as $L_2$-LR. As mentioned above, $L_1$ regularization implicitly performs feature selection because the optimization performed in training usually drives many feature coefficients to zero. Another class of methods that effectively incorporates feature selection is tree structured models. Random Forests is a popular ensemble method which aggregates numerous tree-structured classifiers each generated from a random subset of the training examples (Breiman 2001). Tree structured models are developed by performing feature selection at every stage. Therefore, we also include experiments with $L_1$-LR (varying $C$ among $\{.01, .1, 1, 5, 10, 15, 20, 25, 30\}$) and Random Forests (controlling complexity by changing the parameter for the minimum number of objects allowed to be in a leaf node for each tree among $\{2, 4, 8, 16, 32, 64, 128, 256, 512\}$).

Finally, we also include a brief section that includes numerical regression results. The results in Section 5.6 were generated using $L_2$-regularized linear regression, also known as Ridge Regression. Similarly to logistic regression, ridge regression allows for selection of a regularization parameter $\lambda$ which controls the complexity of the regression. Here, we select the regularization parameter from $\{.01, .1, 1, 5, 10, 15, 20, 25, 30\}$. Kosinski et al. report performance in terms of Pearson's $r$ for numerical regression results, and we follow that as well.

# 5 Experimental Results

This section shows the results of numerous experiments on the suite of Facebook user-Like prediction problems described in Section 4.1, using the methodology and settings from Sections 4.2, 4.3, and 4.4. We compare DR-feature predictive performance versus full-feature predictive performance on the 21 predictive tasks with varied settings: default regularization (hyper)parameters, $k$ chosen using nested cross-validation, and $k$ plus the regularization parameters chosen using nested cross-validation. We also repeat these experiments using different predictive modeling algorithms and different DR techniques, show some preliminary results for alternative predictive tasks (numerical regression and multiclass classification), and further validate our results on an additional six large, sparse data sets.

The results in this section provide strong evidence that modeling using DR features is not generally a good idea for binary classification on large, sparse data. This leads to our core finding: despite DR's popularity, it should not be standard practice to use it for modeling as there is an important class of problems where it decreases performance. Careful regularization can be more universally beneficial than DR. This result holds across all of the DR techniques that we experiment with, especially on easier predictive problems. However, for modeling algorithms that incorporate feature selection, DR can improve performance over the original feature set, although this type of feature set/modeling algorithm choice does not have the best overall performance. The results in this section should be useful to both researchers and practitioners as they demonstrate that undertaking such a comparison is necessary to achieve optimal predictive performance, and that parameter choice can have dramatic effect when using DR for predictive modeling.

### 5.1 $L_2$-LR with Default Regularization and Fixed $k$

The first set of experiments compare performance using scikit-learn's default regularization settings for $L_2$-regularized logistic regression. Kosinski et al. do not discuss regularization in their paper; here, performance on the $k = 100$ SVD feature set using the default regularization ($L_2$-LR with $C = 1$) is roughly comparable to theirs. To illustrate the overall extent to which SVD affects predictive performance, Figure 1 plots the full-feature performance on the $x$-axis and the SVD performance on the $y$-axis. Each point represents one prediction problem's AUC pair. Points above the $y = x$ line on the plot are cases where SVD improves performance; points below are where SVD decreases performance.

The first thing to notice is that SVD does indeed improve performance for many of the targets. The Wilcoxon signed rank test (Wilcoxon et al. 1963) tests for the null hypothesis that pairs of data points drawn from different populations have a median difference of zero. For these pairs of points, the median difference is 0.01 and the resulting $z$ score from applying the test is 0.886, which means that we fail to reject the null hypothesis at $p = .01, .05.$, and .1. These results imply that overall, SVD neither helps nor hurts predictive accuracy. However, also note that there appear to be several regimes within the distribution. Instances toward the far top-right of the plot are actually hurt more than helped by SVD and vice-versa for instances at the bottom-left.

Delving more deeply into these results, notice that SVD tends to decrease performance for

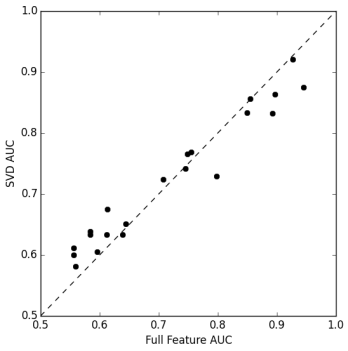



Figure 1: Comparing SVD versus full feature set using LR with default regularization settings. SVD neither helps nor hurts in general. The benefit of SVD seems to relate to the difficulty of the predictive modeling problem. The z-score from Wilcoxon signed-rank test is 0.886 (not significant at $p = .1$).

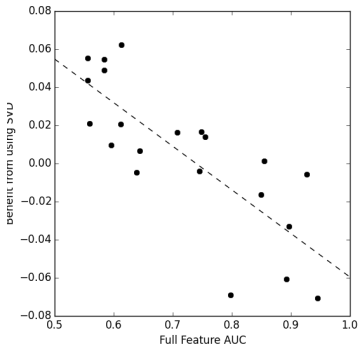



Figure 2: The difficulty of the original predictive problem strongly relates to the amount of benefit from using SVD. When using default regularization, easier predictive problems (with high AUC on the full-feature task) have diminished predictive performance from using SVD; SVD benefits harder predictive problems.

"easier" problems, that is, those for which the AUC using either feature set is relatively higher. The results in Figure 2 bear this idea out. Figure 2 plots the difficulty of the predictive problem versus the degree to which SVD improves predictive accuracy (SVD AUC - full-feature AUC). There is a strong linear relationship between the two sets of points. This implies that for easier problems, caution should be exercised when applying SVD to the feature set, as it may decrease performance; however, for more difficult problems, SVD can improve predictive accuracy when using default regularization settings. This is perhaps because the "more difficult" tasks suffer due to overfitting; SVD in these cases reduces variance sufficiently to improve generalization performance.

### 5.2 Problem-specific Number of SVD Components

As is discussed above, correctly choosing $k$ (the number of SVD components used as features in predictive modeling) is crucial. Given the near-ubiquity of mentions of the importance of this parameter, one might justifiably wonder whether the results in the last subsection were simply due to a non-optimal choice for $k$. Thus, this section reports on a set of experiments using nested cross-validation (Provost and Fawcett 2013) to select $k$. We continue to use $L_2$-LR and the default value for $C$ for both feature sets in this section.

Figure 3 again plots the full-feature performance versus the SVD-feature performance, using

nested cross-validation to select $k$ within every fold for every predictive task instead of $k = 100$ for all SVD tests (as above). As would be expected, the results for SVD are slightly better this time—that is, in most cases, treating $k$ as a hyperparameter and selecting it carefully yields better generalization performance. The median benefit from using SVD here is 0.012, and the $z$-score is 1.686 (significant at $p = .1$).

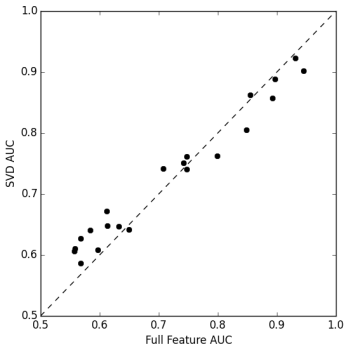



Figure 3: Comparing SVD versus full feature set using default regularization settings and $k$ selected using nested cross-validation. SVD here appears to perform slightly better relative to full-feature performance than in Figure 1: the z-score from the Wilcoxon signed rank test is 1.686 (significant for $p = .1$).

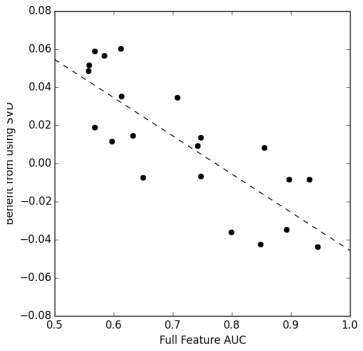



Figure 4: Relating the difficulty of the original predictive problem to the amount of benefit from using SVD, given task-specific choice of $k$. Again, harder predictive problems here have positive benefit from using SVD; this is not necessarily the case for easier problems.

Also note that there still appear to be two regimes within the tasks shown in Figure 4. Once again, more difficult predictive problems appear to be more likely to be improved with use of SVD, while the opposite is true of easier predictive problems (those for which the full-feature AUC is high). Even when choosing $k$ using state-of-the-art predictive modeling techniques, SVD does not necessarily help with predictive performance, especially on easier tasks. The lesson here should thus be that utilizing SVD necessitates also selecting $k$ through nested cross-validation (although such a comparison does add to the total computation time). Of course, these results—while illuminating—are subject to the symmetric criticism that the regularization parameters were not chosen well.

### 5.3 Problem-specific Regularization

While the prior result is itself useful, it draws to mind the intuition that SVD is implicitly performing regularization as Shmueli and Koppius noted: that it can "reduce sampling variance (even at the

cost of increasing bias), and in turn increase predictive accuracy" (Shmueli and Koppius 2011). If one wants better "regularization," it makes sense to move beyond the default regularization settings for the predictive modeling. Thus, let us now repeat the experiments utilizing problem-specific regularization as well as problem-specific choice of $k$. As mentioned in Section 4.4, logistic regression's performance can be dramatically affected by the choice of regularization parameter. The state-of-the-art (S.O.T.A.) practice for selecting these parameters is nested cross-validation.

To do a full fair comparison of SVD versus full-feature performance, the experiments in this subsection compare $L_2$-LR's predictive performance using SVD with both regularization parameters and $k$ chosen using nested cross-validation over a complete grid search covering every combination of $C$ and $k$ (20 total possible combinations).

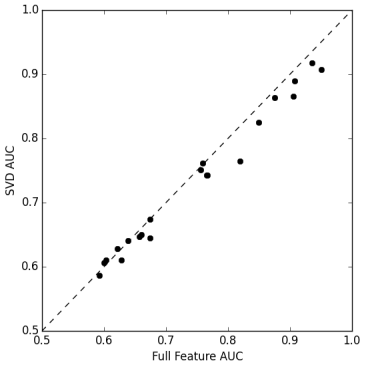



Figure 5: SVD versus full feature performance with regularization settings (and $k$) selected through nested cross-validation. Here, SVD at best gives little advantage and in many cases significantly diminishes performance: SVD is significantly worse ($z$-score from Wilcoxon signed-rank test is -3.076). This result applies regardless of the difficulty of the predictive task.

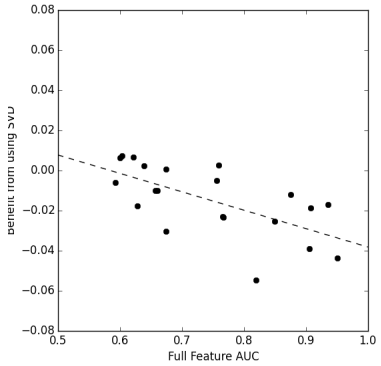



Figure 6: Relationship between difficulty of the original predictive problem and amount of benefit from using SVD, given task-specific choice of $k$ and regularization parameter $C$. The slope is much lower than before; choosing the regularization parameter carefully has a greater effect on the full-feature performance.

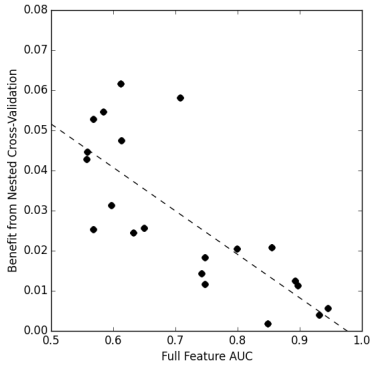



Figure 7: Improvement over default settings when using nested cross-validation to select regularization parameters, when modeling with the original feature set. While utilizing SVD can result in diminished performance for some tasks (depending on the difficulty of the original task; see Figures 4 and 6), choosing regularization parameters carefully only improves performance.

Figure 5 plots the full-feature performance vs. the SVD-feature performance; however, this time all models were built using nested cross-validation to select parameters and evaluate the results. The figure shows that using S.O.T.A. predictive modeling, there is no clear benefit to using SVD at

all. In fact, SVD decreases predictive performance in the majority of cases! Applying the Wilcoxon signed-rank test, we obtain that the median difference is -0.012, z = -3.076, which is significant for $p < .01$. Thus, we reject the null hypothesis—SVD does indeed hurt predictive performance.

With the addition of problem-specific regularization, there no longer appear to be two regimes within the tasks. Similarly to in the last two sections, Figure 6 depicts the relationship between the benefit from using SVD and the difficulty of the predictive problem. While there does still appear to be a linear relationship, there are very few tasks that do benefit from using SVD features, and for those tasks, the benefit is nearly zero.

This result is further illustrated in Figure 7, which shows the extent to which using nested cross-validation to select $C$ improves predictive performance over the default parameter when modeling using the full feature set. Figures 4 and 6 demonstrate that while modeling using SVD features can sometimes improve predictive performance for more difficult tasks, there is a danger that it may also diminish performance. Similarly, Figure 7 also shows us that the benefit of task-specific regularization parameter choice relates somewhat to the original task difficulty (easier problems benefit less); however, unlike use of SVD features, more careful regularization never diminishes performance! This is because complexity control via $L_2$-regularization is supervised and therefore relates directly to the predictive task at hand, while the SVD components are generated in an unsupervised fashion and therefore may not relate well to any specific predictive task. Therefore, it is both safer and more effective to use time and computational resources to careful selection of regularization parameters rather than computation of SVD.

### 5.4 SVD vs. Feature Selection Methods

The results in the prior subsection investigate the benefits of using SVD in the context of $L_2$-regularized logistic regression. Other methods directly incorporate feature selection, a second type of dimensionality reduction separate from the MF-based feature engineering that is the focus of our initial analysis. Feature selection and feature engineering are both employed for complexity control in predictive modeling and so they are ostensibly used for the same goal; however, it is important to investigate the two in context because researchers frequently use them in conjunction. Thus, this section replicates the experiments of Section 5.3 with $L_1$-LR and Random Forest models replacing $L_2$-LR.

Recall that $L_1$-LR performs implicit feature selection by driving most features' coefficients to zero, resulting in sparse coefficient vectors. Figure 8 plots a comparison of full-feature performance versus SVD performance using $L_1$-LR for both, using nested cross-validation to search over both $k$ and $C$. Here, SVD performs significantly better than the full-feature models! The median benefit from using SVD is 0.011, and the $z$-score from a Wilcoxon signed rank test is 2.485 (significant at $p = .05$). It is worth noting that unfortunately the feature selection models tend to perform worse than the full-feature $L_2$-LR performed above. The mean AUC for $L_2$-regularized logistic regression with full features is .745, while the mean AUC for $L_1$-LR with SVD features is .739.

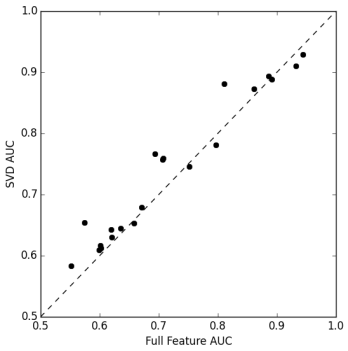



Figure 8: SVD vs. full feature performance using $L_1$-LR. SVD does provide substantial benefit here: the median benefit from using SVD is 0.011, $z$ =2.485.

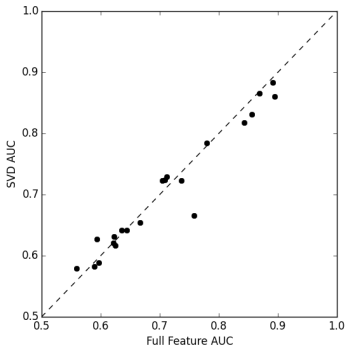
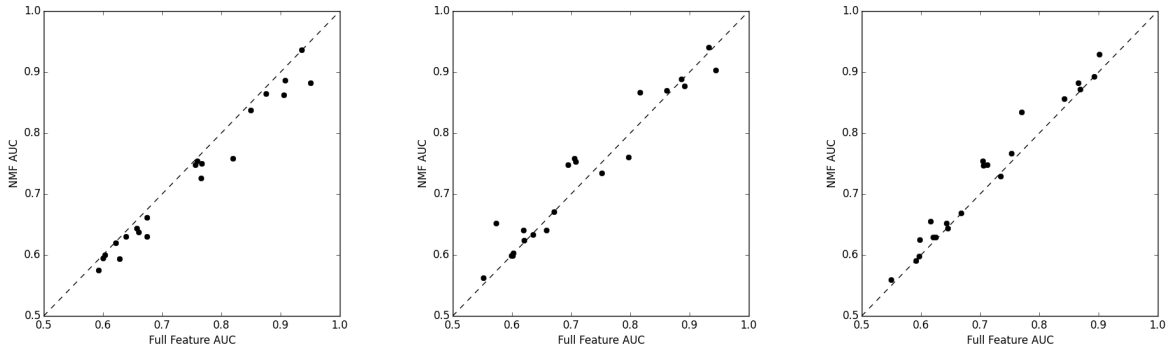



Figure 9: SVD vs. full-feature performance using random forest models. The two are not statistically significantly different; but the performance of both sets of models are worse than the models built using either form of LR.

Similarly, Figure 9 compares full-feature performance vs. SVD performance using Random Forest models. The median benefit from using SVD here is -0.004, $z$ =-0.678 (not significant at $p = .1$). There is not a significant difference between the two; however, note that RF has far worse performance on average than the other two (mean AUC across 21 tasks is .710). Feature selection is directly forced in RF models; if the poor performance of $L_1$-LR could lead us to conclude that feature select methods are not ideal for large, sparse behavioral data, then it makes sense that RF models would perform even worse.

The relatively better performance of SVD in the context of feature selection models makes some sense. Due to the sparse nature of the original features, a limited subset of them is not likely to include all information available to train the best model. Because the SVD features each include information from potentially all of the original features, perhaps the necessary information




(a) $L_2$ logistic regression (b) $L_1$ logistic regression (c) Random Forest

Figure 10: NMF features vs. full features, for various modeling algorithms.

is included even though the modeling algorithm selects a subset of the SVD components.

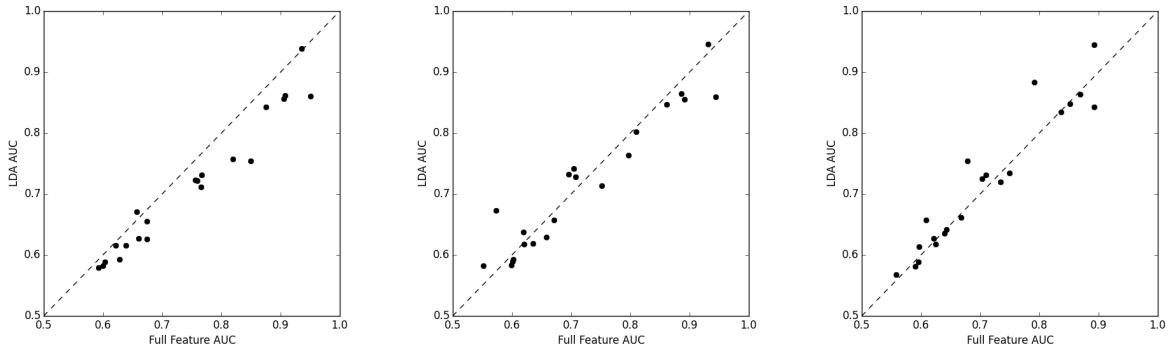
This section's results using modeling methods that incorporate feature selection illuminate two practical facts. First, when performed in conjunction with $L_1$-LR or random forest modeling, DR has better performance relative to the full feature set than it does when used with $L_2$-LR. That is, we can't confidently conclude that DR is *worse* than the full feature set (although, neither can we conclude that it is better!). Second, feature-selection-oriented methods in general do not perform as well. $L_2$-LR on the full feature set dominates all of the methods tried so far.

### 5.5 Other DR Methods

As discussed in Section 4.2, SVD is far from the only MF-based DR algorithm. We also include results for NMF (Non-negative Matrix Factorization) and LDA (Latent Dirichlet Allocation).

Figure 10 replicates the experiments from Sections 5.3 and 5.4, with NMF replacing SVD. The results are very similar to those found using SVD. With $L_2$-LR, the median benefit from using NMF is -0.013 and the Wilcoxon signed-rank test $z$ =-3.945 (so, the full feature performance is better at $p = .01$). With $L_1$-LR, the median difference is 0.002, $z$ =1.303 (not significant at $p = .1$). Finally, for RF models, the median difference is 0.01, $z$ =3.597 (NMF better at $p = .01$). We can conclude that similarly to SVD, $L_2$-LR performs better when using all original features, but that modeling algorithms incorporating feature selection perform better when using DR features.

LDA also produces qualitatively similar results. Figure 11 again shows the DR feature vs. full feature performance across modeling tasks, using the various modeling algorithms. For $L_2$-LR, the




(a) $L_2$ logistic regression (b) $L_1$ logistic regression (c) Random Forest

Figure 11: LDA features vs. full features, for various modeling algorithms.

median difference is -0.033, $z$ =-3.841 (the full features are significantly better at $p = .01$). With $L_1$-LR the median is -0.012 and $z$ =-0.643 (not significant at $p = .1$). For Random Forest models, the median is -0.002 and $z$ =0.782 (not significant at $p = .1$). Again, $L_2$-regularized logistic regression is better with the full features; we can't conclude either feature set is better when modeling with the other algorithms.

Note that the SVD performance is on average the highest out of the three DR techniques (with $L_1$-regularized logistic regression, the averages are .739 for SVD, .729 for NMF, and .716 for LDA). This is perhaps because of the advantages listed in Section 4.2; additionally, NMF and LDA have many more parameters that could be tuned other than $k$. These methods may be improved with further tuning; of course, this would add to the computation time needed to use them.

This section's experiments with the alternative DR methods NMF and LDA not only strengthen our prior results regarding DR's performance relative to the original features, they also provide additional insights. Specifically, we still can't conclude that DR features perform better than original features; this additional preprocessing step frequently degrades predictive performance. Further, none of the DR methods perform better than $L_2$-LR on the full feature set. NMF and LDA, if anything, perform worse than SVD and thus the additional complexity they introduce (they are more time-consuming to compute and have more parameters to tune) is unnecessary.

Table 1 summarizes the performance of the various DR methods and modeling algorithms across the 21 tasks in the Facebook data set. In general, $L_2$-LR using the full feature set has the best performance, and is significantly better than modeling with the DR features across all DR methods.

Table 1: Summary of DR vs. full features, for various modeling algorithms and DR methods.

| | DR Method | DR Features | Full Features | Not Sig. at $\alpha = .95$ |
|---|---|---|---|---|
| L2-LR | SVD | | ✓ | |
| | NMF | | ✓ | |
| | LDA | | ✓ | |
| L1-LR | SVD | ✓ | | |
| | NMF | | | ✓ |
| | LDA | | | ✓ |
| Random Forest | SVD | | | ✓ |
| | NMF | ✓ | | |
| | LDA | | | ✓ |

Conversely, with the other two (feature-selection-based) modeling algorithms, either DR is better or there is not a statistically significant difference. Because DR has marginal benefit at best, it should perhaps be avoided in order to keep the predictive modeling process as simple as possible.

### 5.6 Other Tasks: Numerical Regression and Multiclass Classification

We have so far limited our analysis to classification problems with binary target variables, and found that there is very strong evidence that DR provides little value at best for this task. As examples of cases where DR might provide more value, it is worth investigating the efficacy of SVD for a few different predictive tasks: numerical regression and multi-class classification. Fortunately, the Facebook data set has 10 numerical target variables (which we had binarized for the previous sections). Although we have not as done as deep of an investigation on these tasks, these more limited analyses should provide intriguing avenues for future research.

Figure 12 shows Pearson's r score for full features vs. SVD resulting from numerical modeling via Ridge Regression for the 10 predictive tasks. The median benefit from SVD is .048 (and, in fact, SVD performs better than the full feature set for all but one target variable), so it appears that SVD does provide benefit for numerical regression.

In addition to numerical regression, the numerical target variables allow us to perform multiclass classification by grouping instances into roughly equal-sized classes based on their numerical value. Just as the results in prior sections grouped the instances into two classes, we grouped them into four, eight, and sixteen classes for the results below. The performance reported is the average AUC using one-versus-rest classification for each of the classes (with $L_2$-LR using nested cross-validation

to select regularization parameters and $k$). Intriguingly, it appears that SVD provides more benefit as the number of classes in the data increases.

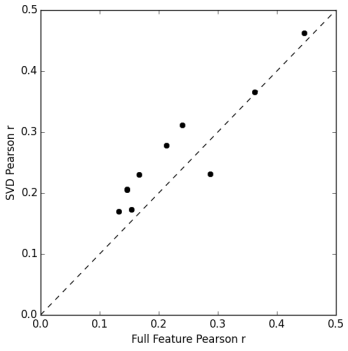



Figure 12: Pearson's r score for numerical regression: SVD vs. full features. SVD provides positive benefit for all but one of the ten numerical target variables.

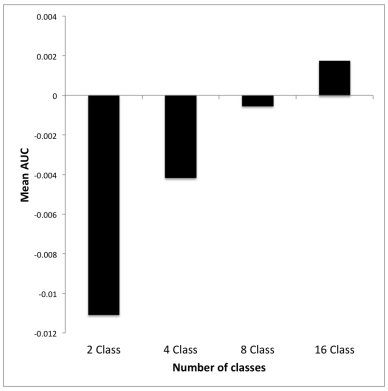



Figure 13: Mean improvement in performance by using SVD for various numbers of classes. The amount of improvement that can be gained by using SVD increases with the number of classes in the data.

We hypothesize that the cause of the difference between these two tasks and binary classification with respect to DR's performance is the amount of variability in the target variable itself. Binary classification has the fewest possible values for the target variable to take on. Although multiclass classification and numerical regression are different (specifically, numerical regression orders the instances while multiclass classification does not), they both yield more possible values for the target variable. Since the primary reason for employing DR is to reduce predictive error from variance, it makes sense that DR might provide more benefit to higher-variance tasks. Again, we stress that these results are more limited than the deep investigation in the prior section, but serve as preliminary examples of situations where DR might provide positive benefit.

### 5.7 Summary

This paper has done a thorough investigation of DR for binary classification on a testbed comprised of predictive modeling tasks drawing from a particular testbed of instances and features. To ensure the above results were not simply due to the choice of data set, we also replicated the same experiments on five additional large, sparse behavioral data sets with associated binary prediction tasks investigated by Junqué de Fortuny et al. (2013): predicting users' ages based on books rated

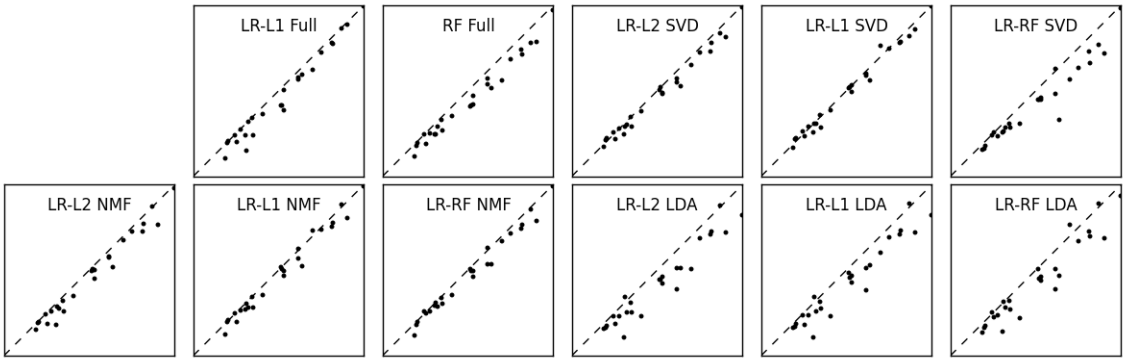



Figure 14: Each chart plots $L_2$-regularized logistic regression performance versus each of the other feature set/modeling algorithm combinations. Points falling below the $y = x$ line are where $L_2$-LR on the full feature set dominates. Clearly this method is superior to every other combination of feature set and modeling algorithm in terms of predictive performance.

(Ziegler et al. 2005); predicting users' genders based on movies rated;[11] predicting users' ages based on products purchased;[12] predicting whether a URL is malicious based on website features (Ma et al. 2009); and predicting volume of comments on pictures based on which users have "favorited" them (Cha et al. 2009).[13] Across these tasks, the models built using the original feature sets generally had better performance than those built using any DR feature sets. We have included the performance on these tasks below in the summary charts, but show individual results across the three modeling algorithms broken out by DR method, in Appendix B, Figure 1.

Thus, we are able to do a complete comparison of all predictive modeling methods across all of the tasks (21 Facebook modeling tasks and 5 from the other data sets). Figure 14 shows the $L_2$-regularized logistic regression performance using the full feature set versus performance for each of the other DR-type and modeling algorithms. These results emphasize that using the full feature set yields superior performance, especially for $L_2$-LR. Thus, we can recommend $L_2$-LR as the default baseline method of choice when doing binary classification with large, sparse behavioral data.

Table 2 shows the average performance for every type of feature set, across the three modeling algorithms explored, for each classification task. A result that immediately stands out is that $L_2$-LR has the best performance in the majority of tasks: 14 out of the total 26. This method also has the highest average AUC across all tasks (.754), and the lowest average rank among the 12

[11] `http://webscope.sandbox.yahoo.com/`

[12] `http://recsyswiki.com/wiki/Grocery_shopping_datasets`

[13] A listing of the sizes of these data sets, the number of nonzero elements, and the base rates can be found in Appendix A, Table 3.

feature/method combinations (2.27). Among the DR feature sets, SVD shows the best performance, especially when used with $L_1$-LR; it had the best performance in 6 classification tasks, the second highest mean AUC (.747), and the second-lowest rank (2.88) (although is important to note that 5 of those 6 tasks were very similar: they comprise the "Big 5" personality test survey items and were generated using the exact same set of individuals).

These two sets of results enable us to make some conclusions that should prove useful for those wishing to use DR in predictive modeling. First, for the best performance overall, use $L_2$-LR with the full feature set. If it is desirable to use matrix-factorization-based DR for any of the other benefits mentioned in Section 3.1 (such as potential data set compression or reduced modeling time), SVD is the superior method, and it performs best if $L_1$-LR is the modeling algorithm used. Similarly, $L_1$-LR is the best feature-selection-based modeling algorithm on these data, and it works best when paired with features generated by SVD (rather than the full feature set).

# 6 Discussion and Conclusions

Our review of the literature establishes that dimensionality reduction is both recommended for and used with the goal of improving predictive performance in applied data science research in Information Systems. This improvement may be achieved through a reduction in the error due to variance in modeling. However, we find that DR's use is often not well justified, in that users don't do (or at least don't mention) a comparison against modeling with the original full set of features using standard complexity control (which also attempts to control error due to variance). This misuse of DR is a direct violation of one of the basic principles of design science: the baseline design for a system should be the simplest one, and the addition of more complexity should be undertaken with caution. In our case, the simpler design is one that utilizes *only* regularization for complexity control; adding DR to the system increases its complexity.

This lack of understanding is particularly troubling in light of the results of our empirical comparison, and thus our thorough treatment of DR in the predictive modeling process as a design science artifact is necessary and will be highly useful to applied data science researchers and industry practitioners alike. The experiments in this paper comprise a comparison on a test-bed comprising 21 binary classification tasks using an intriguing set of Facebook data as well as five supplementary

Table 2: Rank Comparison across all methods, for all target variables and feature sets. $L_2$-LR using the full feature set is the best method for the majority of tasks (14 out of 26); $L_1$-LR using the SVD feature set also performs relatively well—best for 6 tasks (although, note that five of these six tasks are very closely related as they come from the same survey item and were answered by the exact same set of users) Note that presenting results in this fashion stands the risk of a severe multiple comparisons problem: comparing all other results to those from $L_2$-LR means that at least one of those methods may be superior by chance. Even so, the results are striking.

| | Full | | | SVD | | | NMF | | | LDA | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | L2 | L1 | RF | L2 | L1 | RF | L2 | L1 | RF | L2 | L1 | RF |
| Gender | **0.951** | 0.944 | 0.894 | 0.907 | 0.929 | 0.86 | 0.882 | 0.903 | 0.893 | 0.86 | 0.86 | 0.843 |
| Relationship | **0.66** | 0.658 | 0.645 | 0.65 | 0.653 | 0.642 | 0.637 | 0.64 | 0.644 | 0.627 | 0.629 | 0.641 |
| Gay | **0.908** | 0.886 | 0.855 | 0.889 | 0.894 | 0.831 | 0.886 | 0.889 | 0.882 | 0.862 | 0.865 | 0.848 |
| Lesbian | **0.819** | 0.797 | 0.758 | 0.765 | 0.781 | 0.666 | 0.758 | 0.761 | 0.767 | 0.757 | 0.763 | 0.735 |
| Parents at 21 | 0.657 | 0.574 | 0.622 | 0.647 | 0.654 | 0.631 | 0.644 | 0.652 | 0.655 | 0.671 | **0.673** | 0.658 |
| Ethnic | 0.85 | 0.81 | 0.78 | 0.825 | 0.881 | 0.785 | 0.838 | 0.867 | 0.834 | 0.754 | 0.802 | **0.883** |
| Smoker | 0.759 | 0.707 | 0.709 | **0.762** | 0.76 | 0.724 | 0.754 | 0.753 | 0.747 | 0.722 | 0.728 | 0.725 |
| Drinker | 0.766 | 0.693 | 0.711 | 0.743 | **0.767** | 0.73 | 0.75 | 0.748 | 0.748 | 0.731 | 0.732 | 0.732 |
| Drugs | 0.756 | 0.707 | 0.704 | 0.751 | 0.757 | 0.723 | 0.748 | **0.758** | 0.754 | 0.723 | 0.742 | 0.755 |
| Political | **0.875** | 0.861 | 0.843 | 0.863 | 0.873 | 0.817 | 0.864 | 0.87 | 0.857 | 0.843 | 0.847 | 0.834 |
| Religion | 0.935 | 0.932 | 0.891 | 0.918 | 0.91 | 0.884 | 0.937 | 0.941 | 0.929 | 0.938 | **0.946** | 0.945 |
| Network Density | **0.628** | 0.601 | 0.594 | 0.61 | 0.617 | 0.627 | 0.594 | 0.599 | 0.625 | 0.592 | 0.59 | 0.613 |
| Age | **0.905** | 0.891 | 0.869 | 0.866 | 0.889 | 0.865 | 0.862 | 0.877 | 0.872 | 0.856 | 0.856 | 0.864 |
| Friend Count | **0.765** | 0.752 | 0.736 | 0.742 | 0.745 | 0.723 | 0.726 | 0.734 | 0.729 | 0.711 | 0.713 | 0.719 |
| IQ | **0.675** | 0.62 | 0.635 | 0.644 | 0.643 | 0.641 | 0.63 | 0.641 | 0.652 | 0.626 | 0.637 | 0.635 |
| SWL | **0.593** | 0.551 | 0.559 | 0.587 | 0.584 | 0.579 | 0.575 | 0.562 | 0.56 | 0.579 | 0.582 | 0.568 |
| Openness | 0.674 | 0.671 | 0.666 | 0.674 | **0.679** | 0.655 | 0.662 | 0.671 | 0.669 | 0.655 | 0.657 | 0.662 |
| Conscientiousness | 0.621 | 0.62 | 0.622 | 0.628 | **0.631** | 0.62 | 0.62 | 0.624 | 0.629 | 0.615 | 0.618 | 0.627 |
| Extraversion | 0.639 | 0.636 | 0.624 | 0.641 | **0.645** | 0.617 | 0.63 | 0.633 | 0.629 | 0.616 | 0.619 | 0.617 |
| Agreeableness | 0.6 | 0.598 | 0.59 | 0.607 | **0.61** | 0.582 | 0.594 | 0.599 | 0.591 | 0.582 | 0.584 | 0.581 |
| Emotional Stability | 0.603 | 0.602 | 0.596 | 0.611 | **0.612** | 0.589 | 0.6 | 0.603 | 0.598 | 0.589 | 0.593 | 0.588 |
| Book | **0.652** | 0.619 | 0.624 | 0.624 | 0.628 | 0.629 | 0.59 | 0.634 | 0.646 | 0.554 | 0.553 | 0.569 |
| Flickr | 0.809 | 0.784 | 0.786 | 0.775 | 0.8 | 0.814 | 0.79 | 0.812 | **0.815** | 0.757 | 0.757 | 0.752 |
| Movies | **0.808** | 0.79 | 0.769 | 0.785 | 0.795 | 0.741 | 0.788 | 0.783 | 0.768 | 0.695 | 0.693 | 0.693 |
| Ta-Feng | **0.704** | 0.683 | 0.677 | 0.69 | 0.694 | 0.654 | 0.676 | 0.679 | 0.677 | 0.616 | 0.616 | 0.608 |
| URL | **0.998** | 0.997 | 0.985 | 0.995 | 0.997 | 0.992 | 0.989 | 0.996 | 0.996 | 0.91 | 0.91 | 0.967 |
| Wins: | **14** | 0 | 0 | 1 | 6 | 0 | 0 | 1 | 1 | 0 | 2 | 1 |
| Mean AUC: | **0.754** | 0.73 | 0.721 | 0.738 | 0.747 | 0.716 | 0.732 | 0.74 | 0.737 | 0.709 | 0.714 | 0.718 |
| Mean Rank: | **2.27** | 6.42 | 8.5 | 4.62 | 2.88 | 8.62 | 7.04 | 4.96 | 5.5 | 10.0 | 8.69 | 8.5 |

data sets. The results not only enable us to make numerous practical conclusions related to DR's use in predictive modeling and predictive modeling in general, but also yield a valuable lesson in the design of data science systems.

When modeling using $L_2$-regularized logistic regression, utilizing default regularization settings and with a fixed value for $k$, DR tends to improve performance for harder predictive problems and diminish performance for easier ones. Choosing $k$ using nested cross-validation improves the modeling performance when using the DR features; however, the same dichotomy still exists: DR helps on harder problems and hurts for easier ones. Carefully selecting regularization parameters in modeling (again, through nested cross-validation) improves performance for both the full-feature and DR models; however, conducting careful regularization erases any advantage that modeling with the DR features has, even on harder problems. That is to say, utilizing DR to reduce variance can lead to diminished performance over the baseline of the full feature set with default regularization parameter (especially on easier tasks). On the other hand, utilizing nested cross-validation to carefully perform supervised regularization generally does not: on our test-bed, careful regularization did not diminish performance for any of the 21 predictive tasks. These results are not limited to SVD: results using NMF and LDA to accomplish DR are qualitatively similar. In fact, full-feature modeling performs statistically significantly better than modeling using any of the three DR methods when nested cross-validation is used to select both regularization parameters and $k$.

Methods that incorporate feature selection ($L_1$-LR and Random Forests) show different performance. For such models, the DR features perform better than the full feature sets (or at least, not worse; the results are not statistically significant for random forest models). This is not entirely due to an improvement in the SVD performance; the full-feature performance is substantially worse for the $L_1$-LR and RF models than it is for $L_2$-LR. These results hold for NMF and LDA as well. We therefore draw a few further interesting conclusions: feature selection (itself a form of DR, as we discuss in Section 2) works surprisingly poorly on large, sparse data. Conversely, $L_2$-regularized logistic regression using the full feature set is hard to beat, especially when the regularization hyperparameter is chosen carefully. Second, SVD consistently has the best performance out of the three methods that we used and so is a good default option for when DR is desirable or necessary.

The results in this paper are largely limited to one sort of data (large, sparse data mainly generated by human behavior), and one type of predictive modeling task (binary classification).

This particular combination of data type and task is quite common in the predictive modeling literature as well as in industry use, and our results are relatively conclusive; however, we do not mean to imply that DR will not work for *any* data type or modeling task. In Section 5.6, we show preliminary results for a few other predictive tasks where DR does seem to add substantial value: numerical regression and multi-class classification. However, the results from Section 5.6 should be taken as the basis for future work rather than a comprehensive study due to the more limited nature of the investigation and smaller number of data points.

Our results should be taken as guidelines for anyone doing predictive modeling, and anyone considering using DR in particular. We suggest a clear baseline to try when performing binary classification with large, sparse behavioral data: $L_2$-regularized logistic regression. When doing the modeling, careful regularization is key, and should be done using nested cross-validation. If it is absolutely necessary to utilize DR for predictive modeling (perhaps for reasons unrelated to performance, such as the ones we list in Section 3.1), SVD should be the default method of choice, and it is best to use a modeling algorithm that uses feature selection such as $L_1$-regularized logistic regression; however, be especially cautious for easier predictive tasks.

The core contribution of this paper can be summarized as a basic lesson in design science principles: exercise caution when adding complexity via a dimensionality reduction step to the predictive modeling process, even if one feels confident that DR will benefit the performance. We have both established that this principle is frequently violated in the predictive modeling literature, and that this violation is a mistake that can lead to less strong results than might otherwise be possible. However, it is important that our results don't cause other researchers to make the same mistake in reverse! We show a few classes of predictive tasks—numerical regression and multiclass classification—where DR does add value, further supporting our main message: don't rely on intuition or results in other contexts, but compare design alternatives carefully.

Predictive modeling is an important IS artifact. There are a multitude of major and minor variants and modifications that can be made to the process. Unless there is a very good reason, applied researchers and practitioners should keep the process as simple as possible so as not to inadvertently harm predictive performance or do unnecessary computations.

| Task | K's $n$ | Our $n$ | Our $d$ | Base Rate | K's AUC | Our AUC |
|---|---|---|---|---|---|---|
| Single vs. In Relationship | 46,027 | 162,980 | 179,605 | 47% | .67 | .63 |
| Parents together at 21 | 766 | 2,088 | 84,813 | 50% | .6 | .63 |
| Smokes Cigarettes | 1,211 | 3,690 | 118,643 | 25% | .73 | .77 |
| Drinks Alcohol | 1,196 | 3,667 | 118,604 | 51% | .7 | .74 |
| Uses drugs | 856 | 2,711 | 104,869 | 17% | .65 | .77 |
| Caucasian vs. African American | 7,000 | 2,645 | 100,506 | 95% | .95 | .83 |
| Christianity vs. Islam | 18,833 | 3,625 | 105,023 | 95% | .82 | .92 |
| Democrat vs. Republican | 9,752 | 12,936 | 147,759 | 59% | .85 | .86 |
| Gay | | 25,813 | 167,307 | 5% | .88 | .86 |
| Lesbian | | 30,087 | 173,375 | 3% | .75 | .73 |
| Gender | 57,505 | 210,004 | 179,605 | 61% | .93 | .87 |

Table 3: Data details for binary target variables

| Target | $n$ | $d$ | Threshold | Base Rate |
|---|---|---|---|---|
| Satisfaction with Life | 6,512 | 141,815 | 5 | 37% |
| Intelligence | 6,129 | 137,558 | 115 | 49% |
| Emotional Stability | 173,109 | 179,443 | 2.75 | 54% |
| Agreeableness | 173,109 | 179,443 | 3.6 | 50% |
| Extraversion | 173,109 | 179,443 | 3.56 | 50% |
| Conscientiousness | 173,109 | 179,443 | 3.5 | 52% |
| Openness | 173,109 | 179,443 | 4 | 51% |
| Network Density | 46,265 | 178,914 | .05 | 25% |
| Number of Friends | 171,789 | 179,402 | 334 | 25% |
| Age | 185,692 | 179,605 | 30 | 24% |

Table 4: Data details for numeric target variables.

# A Tables

Table 3 shows the amount of data available at the time of publication of Kosinski et al. (2013) versus the number of users and Likes currently available for each binary target variable and the base rates for each of these variables. We also include the AUC obtained by Kosinski et al. versus the AUC we obtained when running our experiments with default regularization and $k$ settings.

Table 4 shows the number of users and Likes for which there is labeled data for each numerical target variable, the thresholds for denoting positive versus negative instances, and the base rates.

To ensure that the results are not just limited to the particular choice of the Facebook Likes dataset, we also replicate the experiments on an additional five large, sparse behavioral data sets with associated binary prediction tasks (Junqué de Fortuny et al. 2013). The tasks associated with

| | Num Features | Num Instances | % Nonzero Entries | Base Rate |
|---|---|---|---|---|
| Book | 62,107 | 285,089 | .0047% | .46 |
| Flickr | 100,000 | 88,787 | .0035% | .27 |
| Movies | 7,619 | 11,916 | .23% | .29 |
| Ta-Feng | 32,266 | 23,812 | .10% | .54 |
| URL | 100,000 | 358,073 | .03% | .33 |

Table 5: Characteristics of new "Big Data" datasets.

each data set are:

1. Book-Crossing: predicting users' ages based on books rated (Ziegler et al. 2005).
2. Yahoo Movies: Predicting users' genders based on movies rated.[14]
3. Ta-Feng: Predicting users' ages based on products purchased.[15]
4. URL: Predicting whether a URL is malicious based on website features (Ma et al. 2009).
5. Flickr: Predicting volume of comments on pictures based on which users have "favorited" them (Cha et al. 2009).

A listing of the sizes of these data sets, the number of nonzero elements, and the base rates can be found in Table 5.

# B Supporting Graphs

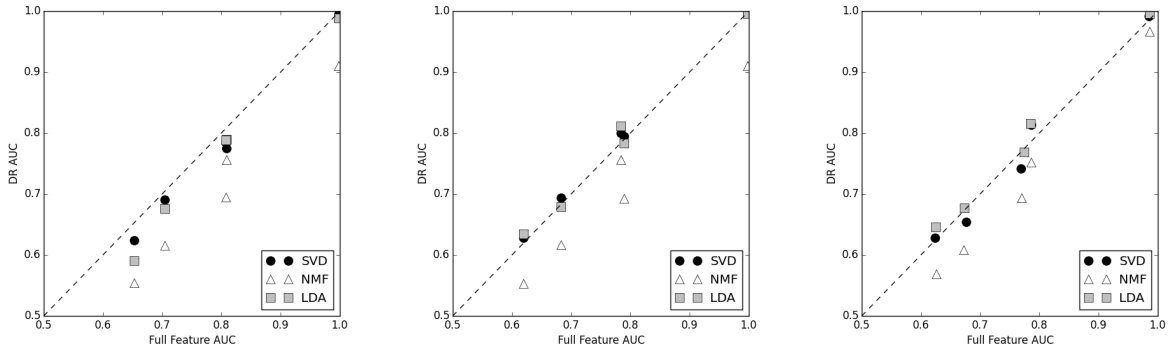
Figure 15 shows the performance for the additional datasets: Book-Crossing, Movies, Ta-Feng, URL, and Flickr.

# References

Dimitris Achlioptas. Database-friendly random projections. In *Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 274–281. ACM, 2001.

Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *Knowledge and Data Engineering, IEEE Transactions on*, 17(6):734–749, 2005.

[14] `http://webscope.sandbox.yahoo.com/`
[15] `http://recsyswiki.com/wiki/Grocery_shopping_datasets`




(a) $L_2$ logistic regression (b) $L_1$ logistic regression (c) Random Forest

Figure 15: DR vs. full feature performance, for various modeling algorithms, across five other data sets.

Hyunchul Ahn, Eunsup Choi, and Ingoo Han. Extracting underlying meaningful features and canceling noise using independent component analysis for direct marketing. *Expert Systems with Applications*, 33(1): 181–191, 2007.

OT Arulogun, EO Omidiora, MA Waheed, OA Fakolujo, OM Olaniyi, et al. On the classification of gasoline-fuelled engine exhaust fume related faults using electronic nose and principal component analysis. *Computing, Information Systems, Development Informatics and Allied Research Journal*, 3(2), 2012.

Richard Bellman. *Adaptive control processes: a guided tour*, volume 4. Princeton university press Princeton, 1961.

István Bíró, Jácint Szabó, and András A Benczúr. Latent dirichlet allocation in web spam filtering. In *Proceedings of the 4th international workshop on Adversarial information retrieval on the web*, pages 29–32. ACM, 2008.

Christopher M Bishop et al. *Pattern recognition and machine learning*, volume 1. springer New York, 2006.

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

Avrim L Blum and Pat Langley. Selection of relevant features and examples in machine learning. *Artificial intelligence*, 97(1):245–271, 1997.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

MC Burl, UM Fayyad, P Perona, P Smyth, and MP Burl. Automating the hunt for volcanoes on venus. In *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*, pages 302–309. IEEE, 1994.

Jinye Cai, Pingping Xu, Huiyu Tang, and Lin Sun. An improved selective ensemble method for spam filtering.

In *Communication Technology (ICCT), 2013 15th IEEE International Conference on*, pages 743–747. IEEE, 2013.

Meeyoung Cha, Alan Mislove, and Krishna P Gummadi. A measurement-driven analysis of information propagation in the flickr social network. In *Proceedings of the 18th international conference on World wide web*, pages 721–730. ACM, 2009.

Kristof Coussement and Dirk Van den Poel. Integrating the voice of customers through call center emails into a decision support system for churn prediction. *Information & Management*, 45(3):164–174, 2008.

Sampath Deegalla and Henrik Bostrom. Reducing high-dimensional data by principal component analysis vs. random projection for nearest neighbor classification. In *Machine Learning and Applications, 2006. ICMLA'06. 5th International Conference on*, pages 245–250. IEEE, 2006.

Scott C. Deerwester, Susan T Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407, 1990.

Matthew F Der, Lawrence K Saul, Stefan Savage, and Geoffrey M Voelker. Knock it off: profiling the online storefronts of counterfeit merchandise. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1759–1768. ACM, 2014.

Inderjit S Dhillon, Subramanyam Mallela, and Rahul Kumar. A divisive information theoretic feature clustering algorithm for text classification. *The Journal of Machine Learning Research*, 3:1265–1287, 2003.

Pedro Domingos. A few useful things to know about machine learning. *Communications of the ACM*, 55 (10):78–87, 2012.

George Forman. An extensive empirical study of feature selection metrics for text classification. *The Journal of machine learning research*, 3:1289–1305, 2003.

Jerome H Friedman. On bias, variance, 0/1loss, and the curse-of-dimensionality. *Data mining and knowledge discovery*, 1(1):55–77, 1997.

Bjarne Ørum Fruergaard, Toke Jansen Hansen, and Lars Kai Hansen. Dimensionality reduction for click-through rate prediction: Dense versus sparse representation. *arXiv preprint arXiv:1311.6976*, 2013.

Aristides Gionis, Piotr Indyk, Rajeev Motwani, et al. Similarity search in high dimensions via hashing. In *VLDB*, volume 99, pages 518–529, 1999.

Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

Isabelle Guyon, Vincent Lemaire, Marc Boullé, Gideon Dror, and David Vogel. Analysis of the kdd cup 2009: Fast scoring on a large orange customer database. In *KDD Cup*, pages 1–22, 2009.

Alan R Hevner, Salvatore T March, Jinsoo Park, and Sudha Ram. Design science in information systems research. *MIS quarterly*, 28(1):75–105, 2004.

Peter D Hoff. Model averaging and dimension selection for the singular value decomposition. *Journal of the American Statistical Association*, 102(478), 2007.

Jian Hu, Hua-Jun Zeng, Hua Li, Cheng Niu, and Zheng Chen. Demographic prediction based on user's browsing behavior. In *Proceedings of the 16th international conference on World Wide Web*, pages 151–160. ACM, 2007.

David D Jensen and Paul R Cohen. Multiple comparisons in induction algorithms. *Machine Learning*, 38 (3):309–338, 2000.

Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. URL `http://www.scipy.org/`. [Online; accessed 2014-10-26].

Enric Junqué de Fortuny, David Martens, and Foster Provost. Predictive modeling with big data: is bigger really better? *Big Data*, 1(4):215–226, 2013.

George Karypis and Eui-Hong Sam Han. Fast supervised dimensionality reduction algorithm with applications to document categorization & retrieval. In *Proceedings of the ninth international conference on Information and knowledge management*, pages 12–19. ACM, 2000.

Rehan M Khan, Chung-Hay Luk, Adeen Flinker, Amit Aggarwal, Hadas Lapid, Rafi Haddad, and Noam Sobel. Predicting odor pleasantness from odorant structure: pleasantness as a reflection of the physical world. *The Journal of Neuroscience*, 27(37):10015–10023, 2007.

YongSeog Kim, W Nick Street, Gary J Russell, and Filippo Menczer. Customer targeting: A neural network approach guided by genetic algorithms. *Management Science*, 51(2):264–276, 2005.

Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

Michal Kosinski, David Stillwell, and Thore Graepel. Private traits and attributes are predictable from digital records of human behavior. *Proceedings of the National Academy of Sciences*, 110(15):5802–5805, 2013.

Pat Langley. Crafting papers on machine learning. In *ICML*, pages 1207–1216, 2000.

Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

Yury Lifshits and Dirk Nowotka. Estimation of the click volume by large scale regression analysis. In *Computer Science–Theory and Applications*, pages 216–226. Springer, 2007.

Huan Liu and Hiroshi Motoda. *Feature extraction, construction and selection: A data mining perspective*. Springer Science & Business Media, 1998.

Míriam López, Javier Ramírez, Juan Manuel Górriz, Ignacio Álvarez, Diego Salas-Gonzalez, Fermín Segovia, Rosa Chaves, Pablo Padilla, and Manuel Gómez-Río. Principal component analysis-based techniques and supervised classification schemes for the early detection of alzheimer's disease. *Neurocomputing*, 74(8):1260–1271, 2011.

Justin Ma, Lawrence K Saul, Stefan Savage, and Geoffrey M Voelker. Identifying suspicious urls: an application of large-scale online learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 681–688. ACM, 2009.

Andrew Y Ng. Feature selection, l 1 vs. l 2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, page 78. ACM, 2004.

Art B Owen and Patrick O Perry. Bi-cross-validation of the svd and the nonnegative matrix factorization. *The Annals of Applied Statistics*, pages 564–594, 2009.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Foster Provost and Tom Fawcett. *Data Science for Business: What you need to know about data mining and data-analytic thinking.* " O'Reilly Media, Inc.", 2013.

Troy Raeder, Claudia Perlich, Brian Dalessandro, Ori Stitelman, and Foster 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.

Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

Yvan Saeys, Iñaki Inza, and Pedro Larrañaga. A review of feature selection techniques in bioinformatics. *bioinformatics*, 23(19):2507–2517, 2007.

Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Analysis of recommendation algorithms for e-commerce. In *Proceedings of the 2nd ACM conference on Electronic commerce*, pages 158–167. ACM, 2000.

Louis L Scharf. The svd and reduced rank signal processing. *Signal processing*, 25(2):113–133, 1991.

Farial Shahnaz, Michael W Berry, V Paul Pauca, and Robert J Plemmons. Document clustering using nonnegative matrix factorization. *Information Processing & Management*, 42(2):373–386, 2006.

Qinfeng Shi, James Petterson, Gideon Dror, John Langford, Alex Smola, and SVN Vishwanathan. Hash kernels for structured data. *The Journal of Machine Learning Research*, 10:2615–2637, 2009.

Galit Shmueli and Otto Koppius. Predictive analytics in information systems research. *Robert H. Smith School Research Paper No. RHS*, pages 06–138, 2010.

Galit Shmueli and Otto R Koppius. Predictive analytics in information systems research. *Mis Quarterly*, 35 (3):553–572, 2011.

Abdulhamit Subasi and M Ismail Gursoy. Eeg signal classification using pca, ica, lda and support vector machines. *Expert Systems with Applications*, 37(12):8659–8666, 2010.

Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

Dirk Thorleuchter, Dirk Van den Poel, and Anita Prinzie. Analyzing existing customers websites to improve the customer acquisition process as well as the profitability prediction in b-to-b marketing. *Expert systems with applications*, 39(3):2597–2605, 2012.

Monica Chiarini Tremblay, Donald J Berndt, Stephen L Luther, Philip R Foulis, and Dustin D French. Identifying fall-related injuries: Text mining the electronic medical record. *Information Technology and Management*, 10(4):253–265, 2009.

Matthew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1):71–86, 1991.

LJP Van der Maaten, EO Postma, and HJ Van Den Herik. Dimensionality reduction: A comparative review. *Journal of Machine Learning Research*, 10:1–41, 2009.

Mike West, Carrie Blanchette, Holly Dressman, Erich Huang, Seiichi Ishida, Rainer Spang, Harry Zuzan, John A Olson, Jeffrey R Marks, and Joseph R Nevins. Predicting the clinical status of human breast cancer by using gene expression profiles. *Proceedings of the National Academy of Sciences*, 98(20): 11462–11467, 2001.

Frank Westad, Margrethe Hersletha, Per Lea, and Harald Martens. Variable selection in pca in sensory descriptive and consumer data. *Food Quality and Preference*, 14(5):463–472, 2003.

Brian Whitman. Semantic rank reduction of music audio. In *Applications of Signal Processing to Audio and Acoustics, 2003 IEEE Workshop on.*, pages 135–138. IEEE, 2003.

Frank Wilcoxon, SK Katti, and Roberta A Wilcox. *Critical values and probability levels for the Wilcoxon rank sum test and the Wilcoxon signed rank test.* American Cyanamid Comp., 1963.

Dongshan Xing and Mark Girolami. Employing latent dirichlet allocation for fraud detection in telecommunications. *Pattern Recognition Letters*, 28(13):1727–1734, 2007.

Xin Xu and Xuening Wang. An adaptive network intrusion detection method based on pca and support vector machines. In *Advanced Data Mining and Applications*, pages 696–703. Springer, 2005.

Yiming Yang. Noise reduction in a statistical approach to text categorization. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 256–263. ACM, 1995.

Cai-Nicolas Ziegler, Sean M McNee, Joseph A Konstan, and Georg Lausen. Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web*, pages 22–32. ACM, 2005.