Do Not Use Principal Components Analysis (PCA) for Prediction Problems, LOL is Typically Better

Joshua T. Vogelstein, Minh Tang, Da Zheng, Randal Burns, Mauro Maggioni

Abstract

It is increasingly common to acquire scientific data with hundreds, millions, or billions of features. The goal of these massive datasets is often to discover directions (or dimensions) that separate the data into two classes (for example, healthy and diseased). Unfortunately, when the dimensionality of the feature set is larger than the sample size, there are an infinite number of equally good options. Moreover, exactly finding a small set of dimensions that maximize classification accuracy is computationally intractable. Instead, the nearly ubiquitous approach is to utilize Principal Components Analysis (PCA) to find a low-dimensional representation of the features that maximize variance. However, in general there is no reason to suspect that the dimensions that maximize classification accuracy will be close to those that maximize variance. Indeed, we show that, under reasonable assumptions, those two sets of directions will be approximately independent. To mitigate this issue, we developed a closed-form method called "Linear Optimal Low-rank" embedding (LoL), which extends PCA by by jointly utilizing both the dimensions that maximize classification accuracy (ignoring the variance) and the dimensions that maximize variance (ignoring the classification task). We show via a combination of theoretical results and simulations that—as compared to PCA—LoL finds a better low-dimensional representation of the data for subsequent classification under linear Gaussian settings. In other words, no matter what the dimensionality of the original data, or the dimensionality of embedded data, or sample size, LoL finds a better representation of the data for subsequent classification, while adding negligible additional computational cost. Additional numerical experiments demonstrate that the same intuition may fruitfully be applied to hypothesis testing and regression. We demonstrate the performance of LoL on three different applications: (i) sparse genetics, (ii) dense images, and (iii) a connectomics dataset with over 500 million features comprising >400 gigabytes. In each case, LoL outperforms the other methods, while only requiring a few minutes on a single machine to run even on the huge dataset. Our open source implementation of LoL is easy to use, and computationally efficient, making it poised to tackle supervised dimensionality reduction challenges across disciplines.

Supervised learning—the art and science of estimating statistical relationships using labeled training data—is a crucial tool in scientific discovery. Supervised learning has enabled a wide variety of basic and applied findings, ranging from discovering biomarkers in omics data [1] to object recognition from images [2]. A special case is classification; a classifier predicts the "class" of a novel observation via partitioning the space of observations (for example, predicting male or female from MRI scans). One of the most foundational and important approaches to classification is called "Fisher's Linear Discriminant Analysis" (LDA) [3]. LDA has a number of highly desirable properties for a reference classifier. First, it is built based on very simple geometric reasoning: when the data are Gaussian, all the information is in the means and variances, so the optimal classifier uses both the means and the variances. Second, due to its simplicity, LDA can be applied to multiclass problems, and can easily be extended to other problems. Third, theorems guarantee that when sample size is large, and dimensionality is small, LDA converges to the optimal classifier under the Gaussian assumption. And finally, again because it is so simple, algorithms for implementing it are highly efficient.

Modern scientific datasets, however, present challenges for classification that were not addressed in Fisher's era. Specifically, the dimensionality of datasets is quickly ballooning. Currently, across many scientific disciplines, the raw data might consist of hundreds of millions of features or dimensions; for example, an entire genome or connectome. While the dimensionality of these data have increased precipitously, the sample sizes have not witnessed a concomitant increase. This "large p, small n" problem is disastrous for many classical statistical approaches because they were designed with "small p, large n" in mind. LDA in particular estimates a hyperplane in $p-1$ dimensions when the data are p dimensional. But there are an infinite number of $p-1$ dimensional hyperplanes that fit the data exactly when $p > n$. To visualize this, imagine fitting a line to a single point, or a plane to two points. In each case, one can choose any rotation, and still

fit the data perfectly. Therefore, without further constraints, algorithms will "over-fit", effectively randomly choosing one of many equally good fits. Supervised manifold learning is field devoted to combating this this over-fitting issue by searching for a small number of dimensions that maximize predictive accuracy. Two complementary strategies have been pursued.

Perhaps the most prominent strategy is to use principal components analysis (PCA) [4] to "pre-process" the data—that is, to reduce the dimensionality of the data prior to a subsequent classification using the much lower dimensional data. While hugely successful, PCA is a wholly *unsupervised* dimensionality reduction technique, meaning that PCA does not utilize the class labels while learning the low dimensional representation. This results in dimensions that have no statistical guarantee of being close to the best ones, and in fact, as we show below, with high probability will be completely wrong. Other unsupervised nonlinear dimensionality reduction techniques, called manifold learning, are ill-equipped to address this problem because they typically only learn a low-dimensional representation for a set of points; thus, they are unable to be applied to new test data. Moreover, they often require costly numerical methods that do not scale, and lack theoretical justification in this setting.

A different strategy that focuses on utilizing the class label information is called "sparsity". Sparse methods find a small subset of the original features to use for subsequent inference [5]. There are many such approaches (often called "feature selection" or "feature screening"); the advantage of these approaches is that the result is sometimes more easily interpretable. The disadvantage, however, is that exactly solving the problem is computationally intractable, requiring time that increases exponentially in the number of features. Various approximations enable efficient algorithms that have provable guarantees under certain limiting assumptions. Lasso is a particularly popular algorithm that has these properties. Unfortunately, Lasso cannot run on millions of dimensions, it has a hyper-parameter that requires careful tuning, and often produces spurious answers even when the unrealistically strict assumptions are met [6]. A more recent approach is called "regularized optimal affine discriminant" (ROAD). ROAD finds the optimal sparse dimensionality reduction under certain Gaussian assumptions. ROAD, however, can only be applied in two-class settings, and requires solving a computationally costly numerical optimization problem, and thus does not scale to large dimensionality.

There is therefore a gap: the field lacks the high-dimensional analog of Fisher's LDA, that is, a method based on simple geometric intuition, that can be applied to multiclass and more general problems, with theorems that guarantee good performance for arbitrarily large dimensionality, and an efficient implementation that scales to hundreds of millions of features. To address these concerns, we developed "Linear Optimal Low-rank" embedding (LOL). The key intuition behind LOL is that we can jointly utilize both directions that are informative with regard to the classification task (the means) and the directions that maximize the variance (the covariance), much like LDA. But by virtue of utilizing random matrix theory, we are able to prove that LOL finds a better low dimensional representation than PCA and other linear methods under the Gaussian assumption. This is true regardless of the dimensionality of the features, the number of samples, and the number of dimensions in which we project. Numerical experiments quantitatively demonstrate the improvement of LOL over reference methods on a wide range of simulated experiments that both satisfy our theoretical assumptions, and go beyond them. In fact, one can extend LOL outside of classification problems to regression and hypothesis testing and obtain qualitatively similar results. Computationally, LOL is always numerically stable, and requires no more computational space, and smaller computational time, than other linear methods. Moreover, we provide a highly scalable implementation that efficiently runs on datasets with hundreds of millions of features comprising hundreds of gigabytes. We then tested LOL against a set of standard methods on several benchmark datasets: there was not a single method that outperformed LOL in any dimension. We provide open source implementations of LOL in both MATLAB and R to support further applications and extensions.

This work therefore makes four complementary challenges to the machine learning literature. First, we provide geometric intuition for how supervised manifold learning methods can work, suggestive of potential algorithms. Second, we provide one example algorithm that satisfies the desiderata, specifically that is simple enough to admit theoretical investigations and efficient implementations. Third, we provide a theoretical framework for evaluating supervised manifold learning for classification that does not depend on the subsequent classifier. And fourth, we provide a scalable implementation that can run on half-terabyte

(A) Obtain Samples   **(B) Learn Projection**   **(C) Project Samples**   **(D) Learn Classifier**
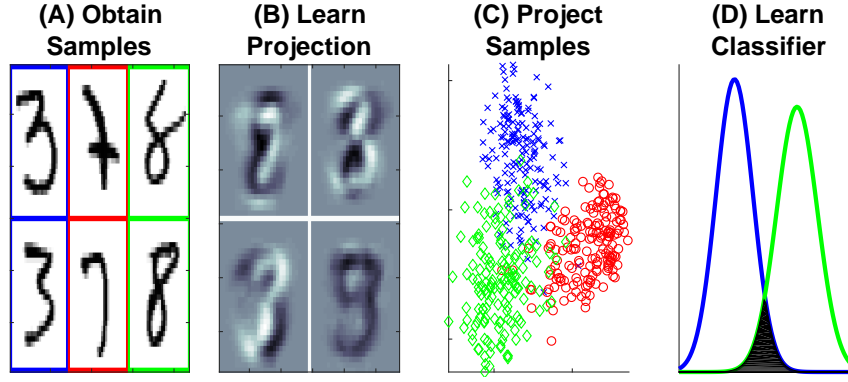
Figure 1: Schematic illustrating Linear Optimal Low-rank (LoL) as a supervised manifold learning technique. **(A)** 300 training samples of the numbers 3, 7, and 8 from the MNIST dataset (100 samples per digit); each sample is a $28 \times 28 = 784$ dimensional image (boundary colors are for visualization purposes). **(B)** The first four projection matrices learned by LoL. Each is a linear combination of the sample images. **(C)** Projecting 500 new (test) samples into the top two learned dimensions; digits color coded as in (A). LoL projected data form three distinct clusters. **(D)** Use the low-dimensional data to learn a classifier. The estimated distributions for 3 and 8 of test samples after projecting data into two dimensions and using LDA to classify demonstrate that 3 and 8 are easily separable by linear methods after LoL projections (color of line indicates the digit). The filled area is the estimated error rate; the goal of any classification algorithm is to minimize that area. LoL is performing well on this high-dimensional real data example.

datasets on single machines in a few minutes. The arguments and methodology developed herein provide a comprehensive framework for developing big supervised manifold learning algorithms to tackle other data science challenges.

Supervised Manifold Learning

A general strategy for supervised manifold learning is schematiized in Figure 1. Step **(A)**, obtain or select n training samples of high-dimensional data. For concreteness, we utilize one of the most popular benchmark datasets, the MNIST dataset [7]. This dataset consists of images of hand-written digits 0 through 9. Each such image is represented by a 28×28 matrix, which means that the observed dimensionality of the data is $p = 28^2 = 784$. Because we are motivated by the $n \ll p$ scenario, we subsample the data to select $n = 300$ examples of the numbers 3, 7, and 8 (100 of each). Step **(B)**, learn a "projection" that maps the high-dimensional data to a low dimension representation. One can do so in a way that ignores which images corresponds to which digit (that "class labels"), as PCA and most manifold learning techniques do, or try to use them (as sparse methods do). LoL uses the class labels to learn projections that are linear combinations of the original data samples. Step **(C)**, use the learned projections to map high-dimensional data into the learned lower dimensional space. This step requires having learned a projection that can be applied to new (test) data samples for which we do not know the true class labels. Nonlinear manifold learning methods typically are unable to be applied in this way (though see [8]). LoL, however, can project new samples in such a way as to separate the data into classes. Finally, step **(D)**, using the low-dimensional representation of the data, learn a classifier. A good classifier correctly identifies as many points as possible with the correct label. When using LDA on the low-dimensional data learned by LoL, the data points are mostly linearly separable, yielding a highly accurate classifier.

Linear Gaussian Intuition

To build intuition about when LoL performs well, and when it does not, we consider the simplest high-dimensional classification setting. We observe n samples $(\boldsymbol{x}_i, y_i)$, where $\boldsymbol{x}_i$ are p dimensional feature vectors, and y_i is the binary class label, that is y_i is either 0 or 1. We assume that both classes are distributed according to a multivariate Gaussian distribution, and the two classes have the same covariance matrix $\boldsymbol{\Sigma}$ and data from either class is equally likely, so that the only difference between the classes is their means,

3

$\boldsymbol{\mu}^1$ and $\boldsymbol{\mu}^2$. The optimal low-dimensional projection is analytically available in this scenario—commonly referred to as Fisher's Linear Discriminant Analysis (LDA)—it is the dot product of the difference of means and the inverse covariance matrix, $(\boldsymbol{\mu}^1 - \boldsymbol{\mu}^2)^\mathsf{T}\boldsymbol{\Sigma}^{-1}$ [9] (see Methods for derivation). When the distribution of the data are unavailable, as in all real data problems, machine learning methods estimate the parameters instead. Unfortunately, when $n < p$, the estimated covariance matrix will not be invertible, so something else is required. As mentioned above, PCA is commonly used to learn a low-dimensional representation. Formally, PCA utilizes the pooled sample mean, $\hat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i$ and the pooled sample covariance matrix, $\hat{\Sigma}$ with entries $\hat{\Sigma}_{kl} = \frac{1}{n}\sum_{i=1}^{n}(x_{ik} - \hat{\mu}_k)(x_{il} - \hat{\mu}_l)$. The PCA projection is the top d eigenvectors of the pooled sample covariance matrix, thus completely ignoring the class labels.

The key insight of our work is that we can combine the class means and the covariance matrix in a simple fashion, rather than just the covariance matrix, to find a low dimensional projection. This is motivated by Fisher's LDA, which utilizes both means and variance, and should therefore improve performance over PCA which only utilizes the variances. More specifically, for a two-class problem, LoL first computes the *sample mean of each class* (as opposed to the pooled mean ignoring class labels), $\hat{\boldsymbol{\mu}}^j = \frac{1}{n_j}\sum_{i:y_i=j} \boldsymbol{x}_i$, where n_j is the number of samples in class j. Second, LoL estimates the difference between means, $\hat{\boldsymbol{\delta}} = \hat{\boldsymbol{\mu}}^1 - \hat{\boldsymbol{\mu}}^2$, just like the optimal strategy. Third, LoL computes the class-conditional covariance matrix, $\widetilde{\Sigma}$ with entries $\widetilde{\Sigma}_{kl} = \sum_{j=1}^{J} \frac{1}{n}\sum_{i:y_i=j}(x_{ik} - \mu_k^j)bx_{il} - \mu_l^j)$. In other words, LoL centers each data point with respect to its own classes mean, rather than the overall pooled mean, and then computes the class-centered covariances. Fourth, LoL computes the eigenvectors of this class-conditionally centered covariance—these eigenvectors correspond to the directions that maximize the variance after subtracting the two class means. If one does not subtract these means, the directions that maximize variance will point towards the means, which are already accounted for by the mean computation. Finally, LoL simply concatenates the difference of the means with the top $d-1$ eigenvectors of $\widetilde{\Sigma}$. Note that the sample class-conditional covariance matrix estimates the population covariance, $\boldsymbol{\Sigma}$, whereas the sample pooled covariance matrix is distorted by the difference of the class means. All together, LoL estimates both the difference of the means and the covariance matrix, just like Fisher's LDA.

Figure 2 shows three different examples of data sampled from the Gaussian model to geometrically illustrate the intuition that motivated LoL (see Methods for details). In each, the top row shows, for $n = 100$ training samples, the first two dimensions of a $p = 1000$ dimensional space, so $n \ll p$. The next four rows each show the distribution of test data after using LDA on the low-dimensional representation.

Figure 2A shows an example we call "stacked cigars". In this example all dimensions are uncorrelated with one another. Moreover, the difference between the means and direction of maximum variance are both large along the same dimension. This is an idealized setting for PCA, because PCA finds the direction of maximal variance, which happens to correspond to the direction of maximal separation of the classes. We also compare this to a method we refer to as PCA'. which uses the top d eigenvectors of the sample class-conditional covariance matrix, $\widetilde{\Sigma}$. Composing this projection with LDA is equivalent to a method called "Reduced Rank LDA" [10] (see Appendix III.B for proof). PCA' performs well here too, for the same reason that PCA does. Because all dimensions are uncorrelated, and one dimension contains most of the information discriminating between the two classes, this is an ideal scenario for sparse methods. Indeed, ROAD, a sparse classifier designed for precisely this scenario, does an excellent job finding the most useful dimensions [11]. LoL, using both the difference of means and the directions of maximal variance also does well. To calibrate all of these methods, we also show the performance of the optimal classifier.

Figure 2B shows an example which is worse for PCA. In particular, the variance is getting larger for subsequent dimensions, $\sigma_1 < \sigma_2 < \cdots < \sigma_p$, while the magnitudes of the difference between the means are decreasing with dimension, $\delta_1 > \delta_2 < \cdots > \delta_p$. Because PCA operates on the pooled sample covariance matrix, the dimensions with the maximum difference are included in the estimate, and therefore, PCA finds some of them, while also finding some of the dimensions of maximum variance, therefore performing fairly well. PCA', however, by virtue of subtracting out the difference of the means, is now completely at chance performance. ROAD is not hampered by this problem, it is also able to find the directions of maximal discrimination, rather than those of maximal variance. Again, LoL, by using both the means and the covariance, does extremely well.

Figure 2: Lol achieves near optimal performance for a wide variety of Gaussian distributions. Each point is sampled from a multivariate Gaussian; the three columns correspond to different simulation parameters (see Methods for details). In each of 3 simulations, we sample $n = 100$ points in $p = 1000$ dimensions, so $n \ll p$. And for each approach, we embed into the top 20 dimensions. Note that we use the sample estimates, rather than the true population values of the parameters. The five columns show (in decreasing order): **Row 1**: A scatter plot of the first two dimensions of the sampled points, with class 0 and 1 as black and gray dots, respectively. the posteriors after projecting using different manifold learning techniques, including **Row 2** Pca. **Row 3** Pca', a method that projects onto the top d eigenvectors of sample class-conditional covariance, **Row 4** Road, a sparse method designed specifically for this model. **Row 5** Lol, our proposed method. **Row 6** the Bayes optimal classifier. In each case, the solid (dashed) line is the estimate posterior for class 0 (class 1), and the overlap of the distributions—which quantifies the magnitude of the error—is filled. The black vertical line shows the estimated threshold for each method. **(A)** The mean difference vector is aligned with the direction of maximal variance, and is mostly concentrated in a single dimension, maxing it ideal for Pca, Pca', and spare methods. In this setting, the results are similar for all methods, and essentially optimal. **(B)** The mean difference vector is orthogonal to the direction of maximal variance, making Pca perform worse, Pca' is at chance, but sparse methods and Lol can still recover the correct dimensions, achieving nearly optimal performance. **(C)** Same as B, but the data are rotated, in this case, only Lol performs well. Note that Lol is closest to Bayes optimal in all three settings.

Figure 2C is exactly the same as B, except the data have been randomly rotated in all 1000 dimensions. This means that none of the original features have much information, rather, linear combinations of them do. This is evidenced by observing the scatter plot, which shows that the first two dimensions fail to disambiguate the two classes. Pca, being rotationally invariant, performs approximately as well in this scenario as in B. Pca' is not helped by this random rotation, so still performs at chance levels. Because there is no small number of features that separate the data well, Road fails. Lol performs nearly as well here as it does in the other examples.

Collectively, these three examples demonstrate when, based purely on geometric intuition, that Lol performs as expected in a variety of Gaussian settings.

Statistical Theory

The above numerical experiments provide the intuition to guide our theoretical developments.

Theorem 1. LoL *is always better than or equal to* Pca' *(and nearly any other linear projection) under the Gaussian model, and better than or equal to* Pca *with additional (relatively weak) conditions. This is true for all possible observed dimensionality of the data, and number of dimensions into which we project, for sufficiently large sample sizes. Moreover, under relatively weak assumptions, these conditions hold almost surely as the number of dimensions increases.*

A formal statement of the theorem and proof are provided in Appendix D. The condition for LoL to be better than Pca is essentially that the dth eigenvector of the pooled sample covariance matrix has less information about classification than the difference of the means vector. The implication of the above theorem is that it is better to incorporate the mean difference vector into the projection matrix than not under basically the same assumptions that motivate Pca. The *degree* of improvement is a function of the projection dimension d, the dimensionality of the feature set p, the number of samples n, and the parameters (see Methods for details and proof), but the *existence* of an improvement—or at least no worse performance—is independent of those factors. It is worth specifying exactly what "better" means in this context. It is desirable to have a notion of better that is agnostic to the subsequent classifier, that is, a metric that quantifies how good an embedding in, no matter which classifier we will use. We utilized Chernoff Information to calculate the distance between the distributions after embedding. Chernoff information is fundamentally related to the expected classification error; specifically, it is the exponential convergence rate for the Bayes error.

## Numerical Experiments Extending Our Theoretical Results

Here we numerically investigate the performance of LoL versus Pca and other methods empirically using simulations, both under the model assumptions for which our theorems hold, as well as more general assumptions for which we currently lack theory. For each of four different scenarios, we sample $n = 100$ training samples each with $p = 100$ features; therefore, FIsher's Lda cannot solve the problem because there are infinitely many ways to overfit. For each setting we evaluate the misclassification rate on held out data for all number of dimensions to embed into. The comparison algorithms are Pca, Pca', and two sparse methods Lasso [12], and Road [11]. Road is a sparse approach specifically designed for Gaussian data, but only works for two-class problems, whereas Lasso was designed for finding sparse dimensions and can be applied to any number of classes.

Theoretical model We begin by investigating two scenarios that satisfy the Lda model assumptions required by our proofs. First, consider the rotated trunk example from Figure 2C as well as a "Toeplitz" example, as depicted in Figures 3A and B, respectively. In both scenarios, for all dimensions, LoL achieves a lower error rate than either of its competitors, often dramatically so.

Multiple Classes LoL can trivially be extended to > 2 class situations, unlike Road. In brief, LoL computes the mean of each class, and then selecets one mean to be the reference, and computes the difference between all the other means and the reference one. Under the linearity assumptiosn, this approach does not lose any information relative to computing the distance between all pairs of means (see Methods for details). We generated data again from the same Trunk example, but added a third class whose mean is the zero vector. We used Lasso as the sparse method approach, which utterly fails in this near sparse setting. As before, LoL outperforms the other methods for all dimensions.

Fat Tails Figure 3D shows a sparse example with "fat tails" to mirror real data settings better. More specifically, each class is the sum of multiple Gaussians, with the same mean, but different covariances (see Methods for details). The qualitative results are consistent with those of the previous numerical experiments, even though we have no theoretical guarantees here. More specifically, LoL outperforms all other methods for all dimensions.

QDA Sometimes, it makes more sense to model each class as having a unique covariance matrix, rather than a shared covariance matrix. Assuming everything is Gaussian, the optimal classifier in this scenario

6
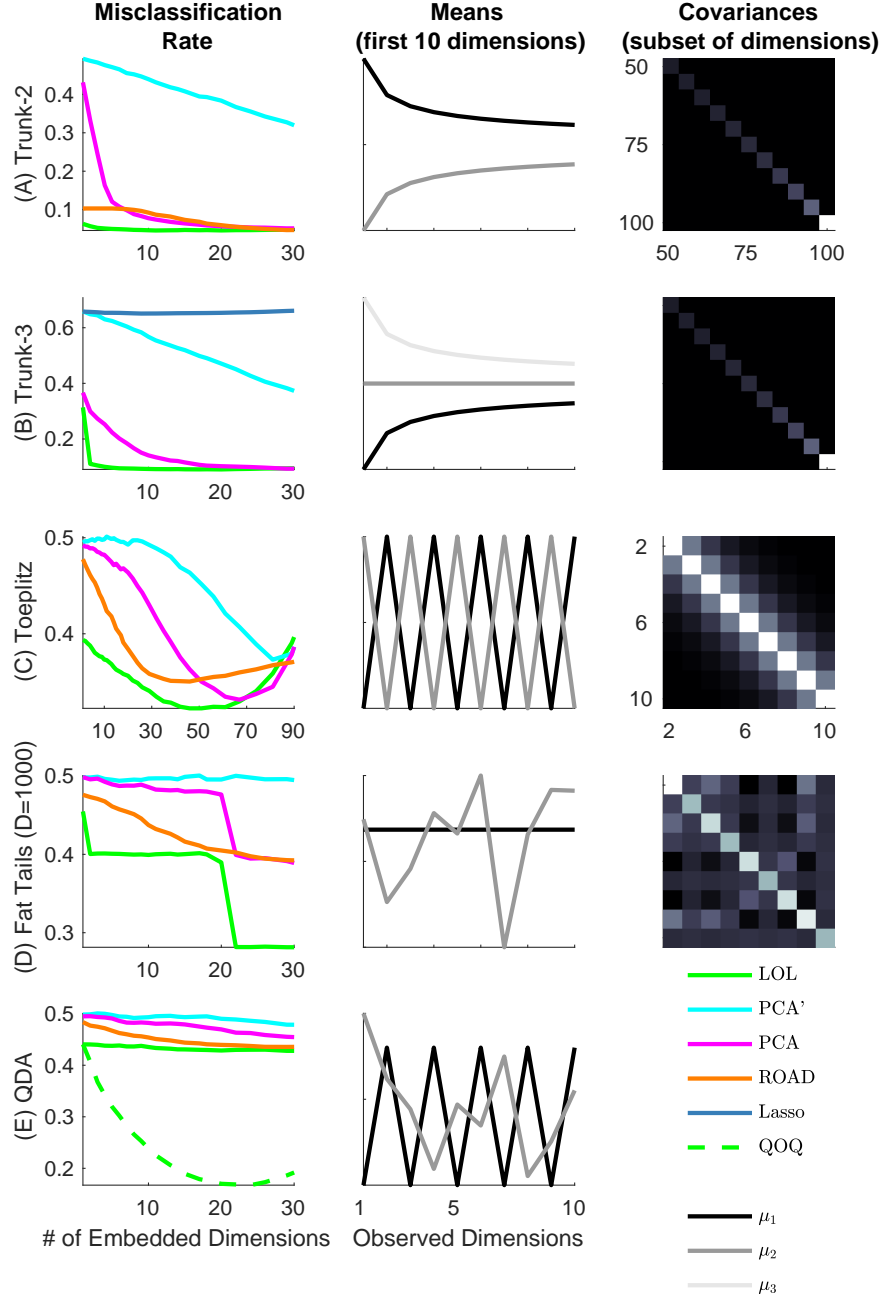
Figure 3: Seven simulations demonstrating `LoL` achieves superior finite sample performance over competitors both in settings for which we have asymptotic theoretical guarantees, and those for which we do not. For the first three, the top panels depict the means (top), the shared covariance matrix (middle). For the next three, the top panels depict a 2D scatter plot (left), mean and level set of one standard deviation of covariance matrix (right). For all seven simulations, the bottom panel shows misclassification rate as a function of the number of embedded dimensions, for several different classifiers. The simulations settings are as follows: **(A)** Rotated Trunk: same as Figure 2C. **(B)** Toeplitz: another setting where mean difference is not well correlated with any eigenvector, and no ambient coordinate is particularly useful on its own. **(C)** 3 Class variant of the rotated Trunk example to demonstrate that `LoL` naturally adapts, and excels in, multi-class problems. **(D)** Fat Tails: a common phenomenon in real data that is more general than our theory supports. **(E)** QDA: `QoQ`, a variant of `LoL` when each class has a unique covariance, outperforms `LoL`, as expected, when the true discriminant boundary is a quadratic, rather than linear, function. In all cases, `LoL`, or the appropriate generalization thereof, outperforms unsupervised or sparse methods for all dimensions. Moreover, the optimal embedding dimension is never the true discriminant dimension, but rather, a smaller number jointly determined by parameter settings and sample size.

209 is called Quadratic Discriminant Analysis (QDA) [13]. Intuitively then, we can modify LoL to compute the
210 eigenvectors separately for each class, and concatenate them (sorting them according to their singular
211 values). Moreover, rather than classifying the projected data with LDA, we can then classify the projected
212 data with QDA. Indeed, simulating data according to such a model (Figure 3E), LoL performs slightly better
213 than chance, regardless of the number of dimensions we use to project, whereas QOQ (for Quadratic Optimal
214 QDA) performs significantly better regardless of how many dimensions it keeps. This demonstrates a
215 straightforward generalization of LoL, available to us because of the simplicity and intuitiveness of LoL.
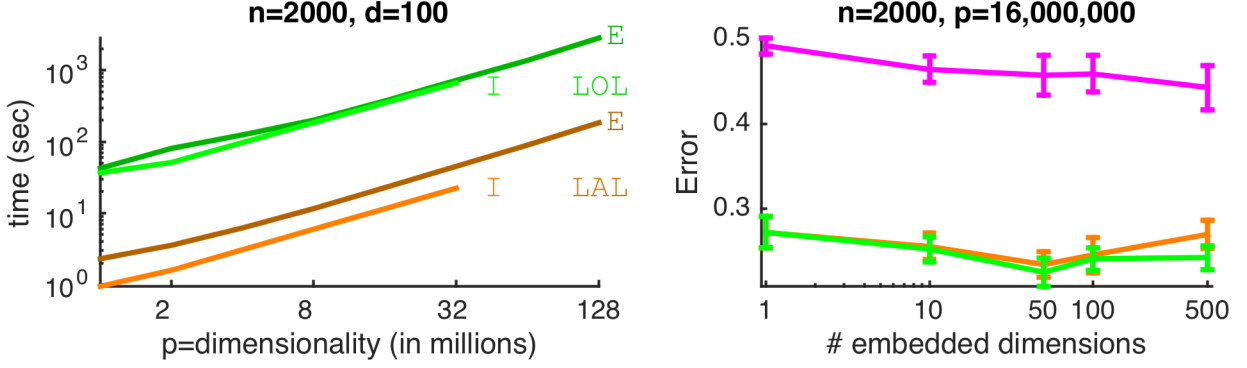
Computational Efficiency

217 When the dimensionality p or sample size n is large, the main bottleneck or concern is sometimes merely
218 the ability to run anything on the data, rather than its predictive accuracy. Fortunately, LoL not only ex-
219 hibits improved performance over standard methods, it also has several algorithmic and implementation
220 level designs that make it computationally efficient. First, LoL admits a closed form solution, enabling
221 it to leverage highly optimized linear algebraic routines rather than the costly iterative programming tech-
222 niques currently required for sparse or dictionary learning type problems. This means that when the dataset
223 is small enough, it will run very quickly. Specifically, the implementation is linear in both sample size and
224 dimensionality (Figure 4A; light green line). When the data size is larger than the memory of computer, how-
225 ever, other techniques are required. Second, LoL is designed to be parallelizable. Typical parallelization
226 strategies distribute work across compute nodes in a large cluster. However, doing so comes with a severe
227 communication cost between the nodes. Instead, we therefore leverage recent advances in computer ar-
228 chitecture, including multicore processors and solid-state drives. Building on FlashX [14–16], we developed
229 extremely efficient LoL implementations with an R interface for ease of use, including both an in memory
230 implementation when the data are small enough to be kept in RAM, and a semi-external memory imple-
231 mentation for larger data. Our implementations enables us to run LoL on essentially arbitrarily large data,
232 achieving in-memory speeds for small data, and enabling the same speeds for multi-terabyte data (Figure
233 4A, dark green line). Third, because LoL is so simple, we can use randomized approximate algorithms
234 to further accelerate its performance. In particular, random projections—for which the data are multiplied
235 by a lower-dimensional random matrix—have been shown to provide excellent approximation eigenvectors
236 [17]. Moreover, very sparse random projections, in which the elements of the matrix are mostly zero, with
237 ± 1 randomly distributed, have been shown to be effective, and have significant computational benefits [18].
238 We therefore further modified FlashX to incorporate very sparse random projections, which we denote by
239 Linear Approximate Low-rank (LAL) . LAL shows an order of magnitude improvement in both the in-memory
240 and semi-external memory implementations (Figure 4A; orange lines).

241 These empirical observations mirror the theoretical bounds of performance. In particular, given T threads
242 with sparsity c, our implementation achieves a computational complexity of $\mathcal{O}(npd/Tc)$, with an optimal
243 speed up and scale up (not shown). Moreover, the error for LoL and LAL are substantially smaller than
244 PCA for this setting for all dimensions (Figure 4B).

Benchmark Real Data Applications

246 Although LoL both statistically and computationally outperforms its natural competitors both in theory and
247 in a variety of simulation settings, real data often tell a different story. We have therefore selected three
248 high-dimensional datasets to compare LoL to several state-of-the-art algorithms (see Methods for details).
249 For each dataset, we compare LoL to (i) PCA, (ii) PCA', (iii) LASSO, and (iv) ROAD for the two-class problems.
250 For LoL, PCA, and PCA', we used Fisher's LDA to classify the points after projecting into a lower dimensional
251 space. Figure 5 shows the results for all three datasets. Panel A shows performance on a commonly used
252 dataset for sparse methods. Even still, LoL achieves a better (lower) misclassification rate than both sparse
253 approaches, as well as PCA and PCA'. Panel B shows performance on the MNIST dataset, one of the most
254 widely used datasets in machine learning. To challenge the methods we only provided 100 samples total
255 from the 10 classes, where each point is a 784 dimensional vector obtained from a 28×28 dimensional
256 image. PCA' does nearly as well as LoL on this dataset, whereas the other methods perform substantially
257 worse. Finally, panel C shows the performance of LoL on a dataset derived from diffussion magnetic

8

Figure 4: Computational efficiency of various low-dimensional projection methods. In all cases, $n = 2000$, and we used spherically symmetric simulation parameters (see Methods for details). We compare PCA with the projection step of LoL (light green for in memory, dark green for semi-external memory) and LAL (light orange for in-memory, dark orange for semi-external memory) for different observed dimensions (p). **(A)** LoL exhibits optimal (linear) scale up and scale out, requiring only 46 minutes to find the embedding on a 2TB dataset, and only 3 minutes using LAL(the sparse constant of sparse random projection $c = \frac{1}{\sqrt{p}}$). **(B)** Error for LAL is the same as LoL in this setting, and both are significantly better than FLDoPCA for all choices of embedding dimension.
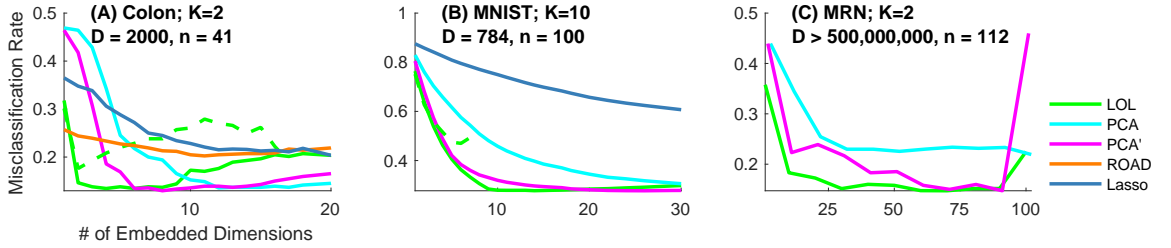
Figure 5: For three datasets, we benchmark LoL (green) versus standard classification methods, PCA' (cyan), PCA (magenta), ROAD (orange), and LASSO (dark blue). For LoL, PCA, and PCA', we compose the embedding with Fisher's LDA. The three panels show the misclassification rate (vertical axis) and the number of projection dimensions (horizontal axis). **(A)** A standard sparse colon cancer genetics dataset. **(B)** A dense image dataset from Figure 1, but this time using all 10 digits. **(C)** A magnetic resonance imaging dataset with over five hundred million features. In all cases, for all number of projection dimensions, LoL does as well or better than all other methods.

resonance imaging. Specifically, we registered the raw data to a standard template (MNI152), and did no further pre-processing. This is in stark contrast to the standard approaches to dealing with this data, which includes a deep processing chain including many steps of parametric modeling and downsampling. At the native resolution each brain volume is over 500 million dimensions, and we have only 112 samples. We learned classifiers on the basis of sex, and evaluated using a leave-one-out procedure. As in the other two applications, LoL achieve a lower misclassification rate for all number of projection dimensions than both PCA and PCA'. Neither ROAD nor LASSO can run on these big data, whereas running this experiment only took a few minutes using our implementation of LoL. Moreover, the minimum misclassification rate was about 15%, which is the same performance we obtain using the severely processed and downsampled data that is typically required. Note that processing these data remains hugely controversial in the future, so we side-step those issues by not processing and simply apply state of the art machine learning on them.

Extensions to Other Supervised Learning Problems

The utility of incorporating the mean difference vector into supervised machine learning for big and wide data extends beyond merely classification. In particular, hypothesis testing can be considered as a special case of classification, with a particular loss function. We therefore apply the same idea to a hypothesis testing

9
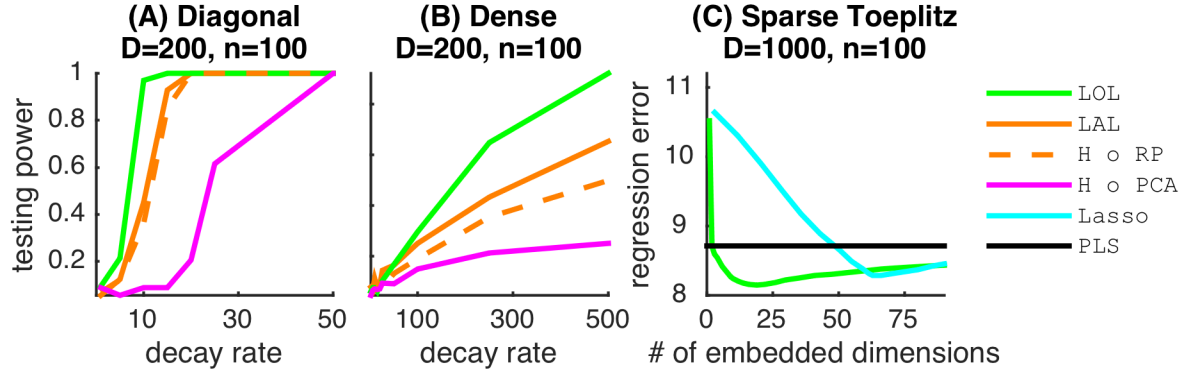
Figure 6: The intuition of including the mean difference vector is equally useful for other supervised manifold learning problems, including testing and regression. **(A)** and **(B)** show two different high-dimensional testing settings, as described in Methods. Power is plotted against the decay rate of the spectrum, which approximates the effective number of dimensions. LoL composed with Hotelling's test outperforms the random projections variants described in [19], as well as several other variants. **(C)** A sparse high-dimensional regression setting, as described in Methods, designed for sparse methods to perform well. Log_{10} mean squared error is plotted against the number of embedded dimensions. LoL composed with linear regression outperforms LASSO (cyan), the classic sparse regression method, as well as partial least squares (PLS; black). These three simulation settings therefore demonstrate the generality of this technique.

scenario. The multivariate generalization of the t-test, called Hotelling's Test, suffers from the same problem as does the classification problem; namely, it requires inverting an estimate of the covariance matrix whose estimate would be low-rank and therefore singular, in the high-dimensional setting. To mitigate this issue in the hypothesis testing scenario, prior art applied similar tricks as they have done in the classification setting. One particularly nice and related example is that of Lopes et al. [19], who addresses this dilemma by using random projections to obtain a low-dimensional representation, following by applying Hotelling's Test in the lower dimensional subspace. Figure 6A and B shows the power of their test (labeled RP) alongside the power of PCA, LoL and LAL for two different conditions. In each case we use some method to project to low dimensions, followed by using Hotelling's test on the projected data. In the first example the true covariance matrix is diagonal, and in the second, the true covariance matrix is dense. The horizontal axis on both panels characterizes the decay rate of the eigenvalues, so larger numbers imply the data is closer to low-rank. (see Methods for details). The results make it clear that the LoL test has higher power for essentially all scenarios. Moreover, it is not merely the replacing random projections with PCA (solid magenta line), nor simply incorporating the mean difference vector (dashed green line), but rather, it appears that LoL for testing uses both modifications to improve performance.

High-dimensional regression is another supervised learning method that can utilize the LoL idea. Linear regression, like classification and Hotelling's Test, requires inverting a matrix as well. By projecting the data only a lower dimensional subspace first, followed by linear regression on the low-dimensional data, we can mitigate the curse of high-dimensions. To choose the projection matrix, we partition the data into K partitions, based on the percentile of the target variable, we obtain a K class classification problem. Then, we can apply LoL to learn the embedding. Figure 6C shows an example of this approach, contrasted with LASSO and partial least squares, in a sparse simulation setting (see Methods for details). LoL is able to find a better low-dimensional projection than LASSO, and performs significantly better than partial least squares, for essentially all choices of number of dimensions to project into.

Discussion

We have introduced a very simple methodology to improve performance on supervised learning problems with wide data (that is, big data where dimensionality is as large or much larger than sample size). In particular, LoL utilizes both the difference of the means, and the covariance matrices, and we proved that it performs better than PCA under reasonable assumptions. This is in stark contrast to most previous

10

approaches, which only utilize the covariance matrices (or kernel variants thereof), or try to solve a difficult optimization theoretic problem.

Related Work One of the first publications to compose Fisher's LDA with an unsupervised learning method was the celebrated Fisherfaces paper [20]. The authors showed via a sequence of numerical experiments the utility of projecting the data using PCA prior to classifying with LDA. We extend this work by adding a supervised component to the initial embedding. Moreover, we provide the geometric intuition for why and when this is advantageous, as well as show numerous examples demonstrating its superiority. Most manifold learning methods, while exhibiting both strong theoretical [21–23] and empirical performance, are fully unsupervised. Thus, in classification problems, they discover a low-dimensional representation of the data, ignoring the labels. This can be highly problematic when the discriminant dimensions and the directions of maximal variance in the learned manifold are not aligned (see Figure 1 for an example). Moreover, nonlinear manifold learning techniques tend to learn a mapping from the original samples to a low-dimensional space, but do not learn a projection, meaning that new samples cannot easily be mapped onto the low-dimensional space, a requirement for supervised learning. Another recent set of methods comes from the statistics community: "sufficient dimensionality reduction" (SIR) or "first two moments" (F2M) methods [24–28]. These methods are theoretically elegant, but typically require the sample size to be larger than the number of observed dimensions (although see [29] for some promising work). Other approaches formulate an optimization problem, such as projection pursuit [30], empirical risk minimization [31], or supervised dictionary learning [32]. These methods are limited because they are prone to fall into local minima, they require costly iterative algorithms, and lack any theoretical guarantees [31]. Other approaches, such as higher criticism thresholding [33] effectively filter the dimensions, possibly prior to performing PCA on the remaining features [34]. These approaches could be combined with LoL in ultrahigh-dimensional problems. Similarly, another recently proposed supervised PCA builds on the elegant Hilbert-Schmidt independence criterion [35] to learn an embedding [36]. Our theory demonstrates that under the Gaussian model, composing this linear projection with the difference of the means will improve subsequent performance under general settings, implying that this will be a fertile avenue to pursue. Finally, several distributed machine learning libraries have become available, including Apache Spark's mllib, H20, Dato, and Vowpal Wabbit [37]. These focus almost entirely on large sample size and low dimensionality regime, whereas the motivating problems of interest for this work are small sample size and high-dimensionality. Moreover, they utilize a distributed platform that is susceptible to low bandwidth communication between nodes, meaning that as one adds resources, the computations cannot scale out optimally, as our LoL implementation does.

In conclusion, the LoL idea, appending the mean difference vector to convert unsupervised manifold learning to supervised manifold learning, has many potential applications and extensions. We have presented the first few. Incorporating additional nonlinearities via kernel methods [38], ensemble methods such as random forests [39], and multiscale methods [23] are all of immediate interest. MATLAB, R, and FlashR code for the experiments performed in this manuscript is available from http://neurodata.io/tools/LOL/.

A Simulations

For most simulation settings, each class is Guassian: $f_{x|y} = \mathcal{N}(\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y)$, $f_y = \mathcal{B}(\pi)$. We typically assume that both classes are equally like, $\pi = 0.5$, and the covariance matrices are the same, $\boldsymbol{\Sigma}_0 = \boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}$. Under such assumptions, we merely specify $\boldsymbol{\theta} = \{\boldsymbol{\mu}_0, \boldsymbol{\mu}_1, \boldsymbol{\Sigma}\}$.

Stacked Cigars

- $\boldsymbol{\mu}_0 = \mathbf{0}$,
- $\boldsymbol{\mu}_1 = (a, b, a, \ldots, a)$,
- $\boldsymbol{\Sigma}$ is a diagonal matrix, with diagonal vector, $\boldsymbol{d} = (1, b, 1 \ldots, 1)$,

where $a = 0.15$ and $b = 4$.

Trunk

- $\boldsymbol{\mu}_0 = b/\sqrt{(1, 3, 5, \ldots, 2p)}$,
- $\boldsymbol{\mu}_1 = -\boldsymbol{\mu}_0$,
- $\boldsymbol{\Sigma}$ is a diagonal matrix, with diagonal vector, $\boldsymbol{d} = 100/\sqrt{(p, p-1, p-2, \ldots, 1)}$,

where $b = 4$.

Rotated Trunk

Same as Trunk, but the data are randomly rotated, that is, we sample Q uniformly from the set of p-dimensional rotation matrices, and then set:

- $\boldsymbol{\mu}_0 \leftarrow Q\boldsymbol{\mu}_0$,
- $\boldsymbol{\mu}_1 \leftarrow Q\boldsymbol{\mu}_1$,
- $\boldsymbol{\Sigma} \leftarrow Q\boldsymbol{\Sigma}Q^{\mathsf{T}}$.

Toeplitz

- $\boldsymbol{\mu}_0 = b \times (1, -1, 1, -1, \ldots, 1)$,
- $\boldsymbol{\mu}_1 = -\boldsymbol{\mu}_0$,
- $\boldsymbol{\Sigma}$ is a Toeplitz matrix, where the top row is $\rho^{(0,1,2,\ldots,p-1)}$,

where b is a function of the Toeplitz matrix such that the noise stays constant as dimensionality increases, and $\rho = 0.5$.

3 Classes

Same as Trunk, but with a third mean equal to the zero vector, $\mu_2 = \mathbf{0}$.

Fat Tails

For this setting, each class is actually a mixture of two Gaussians with the same mean (the two classes have the same covariances):

- $\boldsymbol{\mu}_0 = \mathbf{0}$,
- $\boldsymbol{\mu}_1 = (0, \ldots, 0, 1, \ldots, 1)$, where the first $s = 10$ elements are zero,
- $\boldsymbol{\Sigma}_0$ is a matrix with one's on the diagonal, and 0.2 on the off diagonal,
- $\boldsymbol{\Sigma}_1 = 15 \times \boldsymbol{\Sigma}_0$,

and then we randomly rotated as in the rotated Trunk example.

QDA

A generalization of the Toeplitz setting, where the two classes have two different covariance matrices, meaning that the optimal discriminant boundary is quadratic.

- $\boldsymbol{\mu}_0 = b \times (1, -1, 1, -1, \ldots, 1)$,
- $\boldsymbol{\mu}_1 = -\boldsymbol{Q} \times (\boldsymbol{\mu}_0 + 0.1)$,
- $\boldsymbol{\Sigma}_0$ is the same Toeplitz matrix as described above, and
- $\boldsymbol{\Sigma}_1 = \boldsymbol{Q}\boldsymbol{\Sigma}_0\boldsymbol{Q}^\mathsf{T}$.

Outliers

In this dataset, we generate $n/2$ samples from an inlier model, and the remaining $n/2$ samples from an outlier model. For the inlier model, we first generate a random $d \times p$ dimensional orthonormal matrix, V, where $d = p/10$. Then, the first half of the inlier points are generated by $f_{x|0}$, the next half by $f_{x|1}$, and the remaining points generated by $f_{x|\emptyset}$. For the outliers, we sampled their class randomly from a fair Bernoulli distribution:

- $f_{x|0} = \mathcal{N}_d(0, \sigma^2) \times V^\mathsf{T}$,
- $f_{x|1} = \mathcal{N}_d(0, \sigma^2) \times V^\mathsf{T} + b$,
- $f_{x|\emptyset} = \mathcal{N}_p(\mathbf{0}, \sigma^2\boldsymbol{I})$,
- $f_\emptyset = \mathcal{B}(0.5)$.

where we set $\sigma = 0.1$ and $b = 0.5$.

13

ALGORITHMS	p < n	2-Class LDA MODELS, p > n			GENERALIZED MODELS, p > n					OTHER TASKS, p > n		ALGORITHM PROPERTIES							ALGORITHMS	REF	NOTES
	LDA model	delta & cov aligned	& cov misaligned	delta & cov misaligned rotated	Fat Tails	>2 Class	QDA Model	Outlier Model	Non-linear	Testing	Regress	Fast	Simple	Theory	Open Source	Dense	2-step/ DL	Super-vised			
kNN	X	X	X	X	X	X		X	X		X		X	X	X	X		X	kNN	1	not considered
HCT	X	X	X									X	X	X			X	X	HCT	14	
LDA o Trace-Ratio	X	X	X	X								X	X	X		X	X	X	LDA o Trace-Ratio	4	
F2M / SDR	X			X		X							X	X	X	X		X	F2M / SDR	10	needs n>p
Deep Learning	X				X	X	X	X	X		X			X	X		X	X	Deep Learning	11	best when n>>p
Vowpal Wabbit	X				X	X		X			X	X		X	X		X	X	Vowpal Wabbit	5	best when n>>p
Ridge Regression	X											X	X	X	X	X		X	Ridge Regression	1	regression
Partial Least Squares											X	X						X	Partial Least Squares	1	regression
Naive Bayes	X	X	X			X						X	X	X	X	X		X	Naive Bayes	1,12	
HDDA	X	X	X	X		X									X	X		X	HDDA	6	
ROAD	X	X											X	X	X			X	ROAD	7,8,9	
LASSO	X	X				X					X	X	X	X	X		X	X	LASSO	3	
kSVM	X	X	X	X	X	X	X	X	X				X	X	X			X	kSVM	1	
Random Forest	X												X	X	X			X	Random Forest	1, 15	
LDA	X											X	X	X	X			X	LDA	1	
LDA o PCA	X	X				X					X	X	X	X	X	X	X	1/2	LDA o PCA	2	
LOL	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	LOL	*	
FIGURE		2A	2B, 3A	2C, 3B	3C	3D	3E	3F	3G	5A, 5D	5C, 5D	6									

Figure 7: Table of algorithms and their properties for high-dimensional data. Gray elements indicate that results are demonstrated in the Figure labeled in the bottom row. 'X' denotes relatively good performance for a given setting, or has the particular property.

₃₉₂ # B   Theoretical Background

₃₉₃ ## II.A   The Classification Problem

Let $(\boldsymbol{X}, Y)$ be a pair of random variables, jointly sampled from $F := F_{\boldsymbol{X}, Y} = F_{\boldsymbol{X}|Y} F_Y$. Let $\boldsymbol{X}$ be a multivariate vector-valued random variable, such that its realizations live in p dimensional Euclidean space, $\boldsymbol{x} \in \mathbb{R}^p$. Let Y be a categorical random variable, whose realizations are discrete, $y \in \{0, 1, \ldots C\}$. The goal of a classification problem is to find a function $g(\boldsymbol{x})$ such that its output tends to be the true class label y:

$$g^*(\boldsymbol{x}) := \underset{g \in \mathcal{G}}{\operatorname{argmax}} \, \mathbb{P}[g(\boldsymbol{x}) = y].$$

When the joint distribution of the data is known, then the Bayes optimal solution is:

$$g^*(\boldsymbol{x}) := \underset{y}{\operatorname{argmax}} \, f_{y|\boldsymbol{x}} = \underset{y}{\operatorname{argmax}} \, f_{\boldsymbol{x}|y} f_y = \underset{y}{\operatorname{argmax}} \{\log f_{\boldsymbol{x}|y} + \log f_y\} \tag{1}$$

Denote expected misclassification rate of classifier g for a given joint distribution F,

$$L_g^F := \mathbb{E}[g(\boldsymbol{x}) \neq y] := \int \mathbb{P}[g(\boldsymbol{x}) \neq y] f_{\boldsymbol{x}, y} d\boldsymbol{x} dy,$$

₃₉₄ where $\mathbb{E}$ is the expectation, which in this case, is with respect to F_{XY}. For brevity, we often simply write L_g,
₃₉₅ and we define $L_* := L_{g^*}$.

₃₉₆ ## II.B   Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis (LDA) is an approach to classification that uses a linear function of the first two moments of the distribution of the data. More specifically, let $\mu_j = \mathbb{E}[F_{X|Y=j}]$ denote the class conditional mean, and let $\boldsymbol{\Sigma} = \mathbb{E}[F_X^2]$ denote the joint covariance matrix, and $\pi_j = \mathbb{P}[Y = j]$. Using this notation, we can define the LDA classifier:

$$g_{\text{Lda}}(\boldsymbol{x}) := \underset{y}{\operatorname{argmin}} \, \frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_0)^\mathsf{T} \boldsymbol{\Sigma}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_0) + \mathbb{I}\{Y = y\} \log \pi_y,$$

where $\mathbb{I}\{\cdot\}$ is one when its argument is true, and zero otherwise. Let L_{Lda}^F be the misclassification rate of the above classifier for distribution F. Assuming equal class prior and centered means, $\pi_0 = \pi_1$ and $(\boldsymbol{\mu}_0 + \boldsymbol{\mu}1)/2 = \boldsymbol{0}$, re-arranging a bit, we obtain

$$g_{\text{Lda}}(\boldsymbol{x}) := \underset{y}{\operatorname{argmin}} \, \boldsymbol{x}^\mathsf{T} \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_y.$$

In words, the LDA classifier chooses the class for whom the projection of an input vector $\boldsymbol{x}$, onto $\boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_y$, is maximized. When there are only two classes, this further simplies to

$$g_{2-\text{Lda}}(\boldsymbol{x}) := \mathbb{I}\{\boldsymbol{x}^\mathsf{T} \boldsymbol{\Sigma}^{-1} \boldsymbol{\delta} > 0\},$$

₃₉₇ where $\boldsymbol{\delta} = \boldsymbol{\mu}_0 - \boldsymbol{\mu}_1$. Note that the equal class prior and centered means assumptions merely changes the
₃₉₈ threshold constant from 0 to something else.

₃₉₉ ## II.C   LDA Model

₄₀₀ A statistical model is a family of distributions indexed by a parameter $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, $\mathcal{F}_{\boldsymbol{\theta}} = \{F_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \boldsymbol{\Theta}\}$. Consider
₄₀₁ the special case of the above where $F_{\boldsymbol{X}|Y=y}$ is a multivariate Gaussian distribution, $\mathcal{N}(\boldsymbol{\mu}_y, \boldsymbol{\Sigma})$, where each
₄₀₂ class has its own mean, but all classes have the same covariance. We refer to this model as the LDA model.
₄₀₃ Let $\boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$, and let $\boldsymbol{\Theta}_{C-\text{Lda}} = (\triangle_C, \mathbb{R}^{p \times C}, \mathbb{R}_{\succ 0}^{p \times p})$, where $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_C)$, $\triangle_C$ is the C dimensional
₄₀₄ simplex, that is $\triangle_C = \{\boldsymbol{x} : x_i \geq 0 \forall i, \sum_i x_i = 1\}$, and $\mathbb{R}_{\succ 0}^{p \times p}$ is the set of positive definite $p \times p$ matrices.
₄₀₅ Denote $\mathcal{F}_{\text{Lda}} = \{F_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \boldsymbol{\Theta}_{\text{Lda}}\}$, dropping the superscript C for brevity where appropriate. The following
₄₀₆ lemma is well known:

407 **Lemma 1.** $L_{Lda}^F = L_*^F$ *for any* $F \in \mathcal{F}_{Lda}$.

408 *Proof.* Under the LDA model, the Bayes optimal classifier is available by plugging the explicit distributions
409 into Eq. (1). $\qquad\square$

C Projection Based Classifiers

410

Let $\boldsymbol{A} \in \mathbb{R}^{d \times p}$ be an orthonormal matrix, that is, a matrix that projects p dimensional data into a d dimen-
sional subspace, where $\boldsymbol{A}\boldsymbol{A}^\mathsf{T}$ is the $d \times d$ identity matrix, and $\boldsymbol{A}^\mathsf{T}\boldsymbol{A}$ skis symmetric $p \times p$ matrix with rank d.
The question that motivated this work is: what is the best projection matrix that we can estimate, to use to
"pre-process" the data prior to applying LDA. Projecting the data $\boldsymbol{x}$ onto a low-dimensional subspace, and
the classifying via LDA in that subspace is equivalent to redefining the parameters in the low-dimensional
subspace, $\boldsymbol{\Sigma}_A = \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}^\mathsf{T} \in \mathbb{R}^{d \times d}$ and $\boldsymbol{\delta}_A = \boldsymbol{A}\boldsymbol{\delta} \in \mathbb{R}^d$, and then using g_{Lda}. When $C = 2$, $\pi_0 = \pi_1$, and
$(\mu_0 + \mu_1)/2 = \boldsymbol{0}$, this amounts to:

$$g_A^d(x) := \mathbb{I}\{(\boldsymbol{A}\boldsymbol{x})^\mathsf{T}\boldsymbol{\Sigma}_A^{-1}\boldsymbol{\delta}_A > 0\}, \text{ where } \boldsymbol{A} \in \mathbb{R}^{d \times p}. \tag{2}$$

411 Let $L_A^d := \int \mathbb{P}[g_A(\boldsymbol{x}) = y] f_{\boldsymbol{x},y} d\boldsymbol{x} dy$. Our goal therefore is to be able to choose A for a given parameter
412 setting $\boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\delta}, \boldsymbol{\Sigma})$, such that L_A is as small as possible (note that L_A will never be smaller than L_*).

413 Formally, we seek to solve the following optimization problem:

$$\begin{aligned} \underset{\boldsymbol{A}}{\text{minimize}} \quad & \mathbb{E}[\mathbb{I}\{\boldsymbol{x}^\mathsf{T}\boldsymbol{A}^\mathsf{T}\boldsymbol{\Sigma}_A^{-1}\boldsymbol{\delta}_A > 0\} \neq y] \\ \text{subject to} \quad & \boldsymbol{A} \in \mathbb{R}^{p \times d}, \quad \boldsymbol{A}\boldsymbol{A}^\mathsf{T} = \boldsymbol{I}_{d \times d}, \end{aligned} \tag{3}$$

414 where $\boldsymbol{I}_{u \times v}$ is the $u \times v$ identity matrix identity, that is, $\boldsymbol{I}(i,j) = 1$ for all $i = j \leq \min(u,v)$, and zero otherwise.
415 Let $\mathcal{A}^d = \{\boldsymbol{A} : \boldsymbol{A} \in \mathbb{R}^{d \times p}, \boldsymbol{A}\boldsymbol{A}^\mathsf{T} = \boldsymbol{I}_{d \times d}\}$, and let $\mathcal{A}_* \subset \mathcal{A}$ be the set of $\boldsymbol{A}$ that minimize Eq. (3), and let
416 $\boldsymbol{A}_* \in \mathcal{A}_*$ (where we dropped the superscript d for brevity). Let $L_{\boldsymbol{A}}^* = L_{\boldsymbol{A}_*}$ be the misclassification rate for
417 any $\boldsymbol{A} \in \mathcal{A}_*$, that is, $L_{\boldsymbol{A}}^*$ is the Bayes optimal misclassification rate for the classifier that composes $\boldsymbol{A}$ with
418 LDA.

419 In our opinion, Eq. (3) is the simplest supervised manifold learning problem there is: a two-class classifi-
420 cation problem, where the data are multivariate Gaussians with shared covariances, the manifold is linear,
421 and the classification is done via LDA. Nonetheless, solving Eq. (3) is difficult, because we do not know
422 how to evaluate the integral analytically, and we do not know any algorithms that are guaranteed to find the
423 global optimum in finite time. We proceed by studying a few natural choices for $\boldsymbol{A}$.

III.A Bayes Optimal Projection

424

425 **Lemma 2.** $\boldsymbol{\delta}^\mathsf{T}\boldsymbol{\Sigma}^{-1} \in \mathcal{A}_*$

Proof. Let $\boldsymbol{B} = (\boldsymbol{\Sigma}^{-1}\boldsymbol{\delta})^\mathsf{T} = \boldsymbol{\delta}^\mathsf{T}(\boldsymbol{\Sigma}^{-1})^\mathsf{T} = \boldsymbol{\delta}^\mathsf{T}\boldsymbol{\Sigma}^{-1}$, so that $\boldsymbol{B}^\mathsf{T} = \boldsymbol{\Sigma}^{-1}\boldsymbol{\delta}$, and plugging this in to Eq. (2), we
obtain

$$\begin{aligned} g_B(x) &= \mathbb{I}\{\boldsymbol{x}\boldsymbol{B}^\mathsf{T}\boldsymbol{\Sigma}_B^{-1}\boldsymbol{\delta}_B > 0\} \\ &= \mathbb{I}\{\boldsymbol{x}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\delta} \times (\boldsymbol{\Sigma}_B^{-1}\boldsymbol{\delta}_B) > 0\} && \text{plugging in } \boldsymbol{B} \\ &= \mathbb{I}\{\boldsymbol{x}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\delta}k > 0\} && \text{because } \boldsymbol{\Sigma}_B^{-1}\boldsymbol{\delta}_B > 0. \end{aligned}$$

426 In other words, letting $\boldsymbol{B}$ be the Bayes optimal projection recovers the Bayes classifier, as it should. Or,
427 more formally, for any $F \in \mathcal{F}_{\mathrm{Lda}}$, $L_{\boldsymbol{\delta}^\mathsf{T}\boldsymbol{\Sigma}^{-1}} = L_*$ $\qquad\square$

428 III.B Principle Components Analysis (Pca) Projection

429 Principle Components Analysis (Pca) finds the directions of maximal variance in a dataset. Pca is closely
430 related to eigendecompositions and singular value decompositions (Svd). In particular, the top principle
431 component of a matrix $\boldsymbol{X} \in \mathbb{R}^{p \times n}$, whose columns are centered, is the eigenvector with the largest corre-
432 sponding eigenvalue of the centered covariance matrix $\boldsymbol{X}\boldsymbol{X}^\top$. Svd enables one to estimate this eigenvector
433 without ever forming the outer product matrix, because Svd factorizes a matrix $\boldsymbol{X}$ into $\boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^\top$, where $\boldsymbol{U}$
434 and $\boldsymbol{V}$ are orthonormal $p \times n$ matrices, and $\boldsymbol{S}$ is a diagonal matrix, whose diagonal values are decreasing,
435 $s_1 \geq s_2 \geq \cdots > s_n$. Defining $\boldsymbol{U} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_n]$, where each $\boldsymbol{u}_i \in \mathbb{R}^p$, then $\boldsymbol{u}_i$ is the ith eigenvector, and s_i
436 is the square root of the ith eigenvalue of $\boldsymbol{X}\boldsymbol{X}^\top$. Let $\boldsymbol{A}_d^{\mathrm{Pca}} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_d]$ be the truncated Pca orthonormal
437 matrix.

The Pca matrix is perhaps the most obvious choice of a orthonormal matrix for several reasons. First,
truncated Pca minimizes the squared error loss between the original data matrix and all possible rank d
representations:

$$\underset{A \in \mathbb{R}^{d \times p}: \boldsymbol{A}\boldsymbol{A}^\top = \boldsymbol{I}_{d \times d}}{\operatorname{argmin}} \left\| \boldsymbol{X} - \boldsymbol{A}^T\boldsymbol{A} \right\|_F^2.$$

438 Second, the ubiquity of Pca has led to a large number of highly optimized numerical libraries for computing
439 Pca (for example, LAPACK [40]).

440 Moreover, let $\boldsymbol{U}_d = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_d] \in \mathbb{R}^{p \times d}$, and note that $\boldsymbol{U}_d^\top \boldsymbol{U}_d = \boldsymbol{I}_{d \times p}$ and $\boldsymbol{U}_d^\top \boldsymbol{U}_d = \boldsymbol{I}_{p \times d}$. Similarly, let
441 $\boldsymbol{U}\boldsymbol{S}\boldsymbol{U}^\top = \boldsymbol{\Sigma}$, and $\boldsymbol{U}\boldsymbol{S}^{-1}\boldsymbol{U}^\top = \boldsymbol{\Sigma}^{-1}$. Let $\boldsymbol{S}_d$ be the matrix whose diagonal entries are the eigenvalues, up to
442 the dth one, that is $\boldsymbol{S}_d(i, j) = s_i$ for $i = j \leq d$ and zero otherwise. Similarly, $\boldsymbol{\Sigma}_d = \boldsymbol{U}\boldsymbol{S}_d\boldsymbol{U}^\top = \boldsymbol{U}_d\boldsymbol{S}_d\boldsymbol{U}_d^\top$.

443 Let $g_{\mathrm{Pca}}^d := g_{A_{\mathrm{Pca}}^d}$, and let $L_{\mathrm{Pca}}^d := L_{A_{\mathrm{Pca}}^d}$. And let $g_{\mathrm{Lda}}^d := \mathbb{I}\{x\boldsymbol{\Sigma}_d^{-1}\boldsymbol{\delta} > 0\}$ be the regularized Lda classifier, that
444 is, the Lda classifier, but sets the bottom $p - d$ eigenvalues to zero.

445 **Lemma 3.** $L_{Pca}^d = L_{RR-Lda}^d$.

Proof. Plugging $\boldsymbol{U}_d$ into Eq. (2) for $\boldsymbol{A}$, and considering only the left side of the operand, we have

$$\begin{aligned}
(\boldsymbol{A}\boldsymbol{x})^\top \boldsymbol{\Sigma}_A^{-1} \boldsymbol{\delta}_A &= \boldsymbol{x}^\top \boldsymbol{A}^\top \boldsymbol{A} \boldsymbol{\Sigma}^{-1} \boldsymbol{A}^\top \boldsymbol{A} \boldsymbol{\delta}, \\
&= \boldsymbol{x}^\top \boldsymbol{U}_d \boldsymbol{U}_d^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{U}_d \boldsymbol{U}_d^\top \boldsymbol{\delta}, \\
&= \boldsymbol{x}^\top \boldsymbol{U}_d \boldsymbol{U}_d^\top \boldsymbol{U} \boldsymbol{S}^{-1} \boldsymbol{U} \boldsymbol{U}_d \boldsymbol{U}_d^\top \boldsymbol{\delta}, \\
&= \boldsymbol{x}^\top \boldsymbol{U}_d \boldsymbol{I}_{d \times p} \boldsymbol{S}^{-1} \boldsymbol{I}_{p \times d} \boldsymbol{U}_d^\top \boldsymbol{\delta}, \\
&= \boldsymbol{x}^\top \boldsymbol{U}_d \boldsymbol{S}_d^{-1} \boldsymbol{U}_d^\top \boldsymbol{\delta}, \\
&= \boldsymbol{x}^\top \boldsymbol{\Sigma}_d^{-1} \boldsymbol{\delta}.
\end{aligned}$$

446   □

447 The implication of this lemma is that if one desires to implement RR-Lda, rather than first learning the
448 eigenvectors and then learning Lda, one can instead directly implement regularized Lda by setting the
449 bottom $p - d$ eigenvalues to zero.

450 III.C Linear Optimal Low-Rank (Lol) Projection

451 The basic idea of Lol is to use both $\boldsymbol{\delta}$ and the top d eigenvectors. Most naïvely, we could simply concatenate
452 the two, $\boldsymbol{A}_{\mathrm{Lol}}^d = [\boldsymbol{\delta}, \boldsymbol{A}_{\mathrm{Pca}}^{d-1}]$. Recall that eigenvectors are orthonormal. To maintain orthonormality, we could
453 easily apply Gram-Schmidt, $\boldsymbol{A}_{\mathrm{Lol}}^d = \text{ORTH}([\boldsymbol{\delta}, \boldsymbol{A}_{\mathrm{Pca}}^{d-1}])$. Both in practice and in theory (as will be shown
454 below), this orthogonalization step does not matter much.

455 to ensure that they are balanced appropriately, we normalize $\boldsymbol{\delta}$

456 each vector in $\boldsymbol{\delta}$ to have norm unity. Formally, let $\widetilde{\boldsymbol{\delta}}_j = \boldsymbol{\delta}_j / \|\boldsymbol{\delta}_j\|$, where $\boldsymbol{\delta}_j$ is the jth difference of the
457 mean vector (remember, the number of vectors is equal to $C - 1$, where C is the total number of classes),

⁴⁵⁸ and let $A_{\mathtt{Lol}}^d = [\widetilde{\boldsymbol{\delta}}, A_{\mathtt{Pca}}^{d-(C-1)}]$. The eigenvectors are all normalized and orthogonal to one another; to
⁴⁵⁹ impose orthogonality between $\widetilde{\boldsymbol{\delta}}$ and the eigenvectors, we could use any number of numerically optimized
⁴⁶⁰ algorithms. However, in practice, orthogonalizing does not matter very much, so we do not bother. We
⁴⁶¹ formally demonstrate this below.

D   Theoretical Properties of LDA based Classifiers

IV.A   `LDA` is rotationally invariant

⁴⁶⁴ For certain classification tasks, the ambient coordinates have intrinsic value, for example, when simple
⁴⁶⁵ interpretability is desired. However, in many other contexts, interpretability is less important [41]. When
⁴⁶⁶ the exploitation task at hand is invariant to rotations, then we have no reason to restrict our search space
⁴⁶⁷ to be sparse in the ambient coordinates, rather, for example, we can consider sparsity in the eigenvector
⁴⁶⁸ basis. Fisherfaces is one example of a rotationally invariant classifier, under certain model assumptions.
⁴⁶⁹ Let $\boldsymbol{W}$ be a rotation matrix, that is $\boldsymbol{W} \in \mathcal{W} = \{\boldsymbol{W} : \boldsymbol{W}^\top = \boldsymbol{W}^{-1}$ and $\det(\boldsymbol{W}) = 1\}$. Moreover, let
⁴⁷⁰ $\boldsymbol{W} \circ F$ denote the distribution F after transformation by an operator $\boldsymbol{W}$. For example, if $F = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ then
⁴⁷¹ $\boldsymbol{W} \circ F = \mathcal{N}(\boldsymbol{W}\boldsymbol{\mu}, \boldsymbol{W}\boldsymbol{\Sigma}\boldsymbol{W}^\top)$.

Definition 1. *A rotationally invariant classifier has the following property:*

$$L_g^F = L_g^{W \circ F}, \qquad F \in \mathcal{F}.$$

⁴⁷² *In words, the Bayes risk of using classifier g on distribution F is unchanged if F is first rotated, for any*
⁴⁷³ $F \in \mathcal{F}$.

⁴⁷⁴ Now, we can state the main lemma of this subsection: `LDA` is rotationally invariant.

⁴⁷⁵ **Lemma 4.** $L_{Lda}^F = L_{Lda}^{W \circ F}$, *for any* $F \in \mathcal{F}$.

⁴⁷⁶ *Proof.* `LDA` simply becomes thresholding $\boldsymbol{x}^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{\delta}$. Thus, we can demonstrate rotational invariance by
⁴⁷⁷ demonstrating that $\boldsymbol{x}^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{\delta}$ is rotationally invariant.

$$
\begin{aligned}
(\boldsymbol{W}\boldsymbol{x})^\top (\boldsymbol{W}\boldsymbol{\Sigma}\boldsymbol{W}^\top)^{-1} \boldsymbol{W}\boldsymbol{\delta} &= \boldsymbol{x}^\top \boldsymbol{W}^\top (\boldsymbol{W}\boldsymbol{U}\boldsymbol{S}\boldsymbol{U}^\top \boldsymbol{W}^\top)^{-1} \boldsymbol{W}\boldsymbol{\delta} && \text{by substituting } \boldsymbol{U}\boldsymbol{S}\boldsymbol{U}^\top \text{ for } \boldsymbol{\Sigma} \\
&= \boldsymbol{x}^\top \boldsymbol{W}^\top (\widetilde{\boldsymbol{U}}\boldsymbol{S}\widetilde{\boldsymbol{U}}^\top)^{-1} \boldsymbol{W}\boldsymbol{\delta} && \text{by letting } \widetilde{\boldsymbol{U}} = \boldsymbol{W}\boldsymbol{U} \\
&= \boldsymbol{x}^\top \boldsymbol{W}^\top (\widetilde{\boldsymbol{U}}\boldsymbol{S}^{-1}\widetilde{\boldsymbol{U}}^\top) \boldsymbol{W}\boldsymbol{\delta} && \text{by the laws of matrix inverse} \\
&= \boldsymbol{x}^\top \boldsymbol{W}^\top \boldsymbol{W}\boldsymbol{U}\boldsymbol{S}^{-1}\boldsymbol{U}^\top \boldsymbol{W}^\top \boldsymbol{W}\boldsymbol{\delta} && \text{by un-substituting } \boldsymbol{W}\boldsymbol{U} = \widetilde{\boldsymbol{U}} \\
&= \boldsymbol{x}^\top \boldsymbol{U}\boldsymbol{S}^{-1}\boldsymbol{U}^\top \boldsymbol{\delta} && \text{because } \boldsymbol{W}^\top \boldsymbol{W} = \boldsymbol{I} \\
&= \boldsymbol{x}^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{\delta} && \text{by un-substituting } \boldsymbol{U}\boldsymbol{S}^{-1}\boldsymbol{U}^\top = \boldsymbol{\Sigma}
\end{aligned}
$$

⁴⁷⁸ $\qquad$ $\square$

⁴⁷⁹ One implication of this lemma is that we can reparameterize without loss of generality. Specifically, defining
⁴⁸⁰ $\boldsymbol{W} := \boldsymbol{U}^\top$ yields a change of variables: $\boldsymbol{\Sigma} \mapsto \boldsymbol{S}$ and $\boldsymbol{\delta} \mapsto \boldsymbol{U}^\top \boldsymbol{\delta} := \boldsymbol{\delta}''$, where $\boldsymbol{S}$ is a diagonal covariance
⁴⁸¹ matrix. Moreover, let $\boldsymbol{d} = (\sigma_1, \ldots, \sigma_D)^\top$ be the vector of eignevalues, then $\boldsymbol{S}^{-1}\boldsymbol{\delta}' = \boldsymbol{d}^{-1} \odot \widetilde{\boldsymbol{\delta}}$, where
⁴⁸² $\odot$ is the Hadamard (entrywise) product. The `LDA` classifier may therefore be encoded by a unit vector,
⁴⁸³ $\widetilde{\boldsymbol{d}} := \frac{1}{m} \boldsymbol{d}^{-1} \odot \widetilde{\boldsymbol{\delta}}'$, and its magnitude, $m := \left\| \boldsymbol{d}^{-1} \odot \widetilde{\boldsymbol{\delta}} \right\|$. This will be useful later.

IV.B   Rotation of Projection Based Linear Classifiers g_A

⁴⁸⁵ By a similar argument as above, one can easily show that:

$$(\boldsymbol{AWx})^{\mathsf{T}}(\boldsymbol{AW\Sigma W}^{\mathsf{T}}\boldsymbol{A}^{\mathsf{T}})^{-1}\boldsymbol{AW\delta} = \boldsymbol{x}^{\mathsf{T}}(\boldsymbol{W}^{\mathsf{T}}\boldsymbol{A}^{\mathsf{T}})(\boldsymbol{AW})\boldsymbol{\Sigma}^{-1}(\boldsymbol{W}^{\mathsf{T}}\boldsymbol{A}^{\mathsf{T}})(\boldsymbol{AW})\boldsymbol{\delta}$$
$$= \boldsymbol{x}^{\mathsf{T}}\boldsymbol{Y}^{\mathsf{T}}\boldsymbol{Y}\boldsymbol{\Sigma}^{-1}\boldsymbol{Y}^{\mathsf{T}}\boldsymbol{Y}\boldsymbol{\delta}$$
$$= \boldsymbol{x}^{\mathsf{T}}\boldsymbol{Z}\boldsymbol{\Sigma}^{-1}\boldsymbol{Z}^{\mathsf{T}}\boldsymbol{\delta}$$
$$= \boldsymbol{x}^{\mathsf{T}}(\boldsymbol{Z}\boldsymbol{\Sigma}\boldsymbol{Z}^{\mathsf{T}})^{-1}\boldsymbol{\delta} = \boldsymbol{x}^{\mathsf{T}}\widetilde{\boldsymbol{\Sigma}}_d^{-1}\boldsymbol{\delta},$$

where $\boldsymbol{Y} = \boldsymbol{AW} \in \mathbb{R}^{d \times p}$ so that $\boldsymbol{Z} = \boldsymbol{Y}^{\mathsf{T}}\boldsymbol{Y}$ is a symmetric $p \times p$ matrix of rank d. In other words, rotating and then projecting is equivalent to a change of basis. The implications of the above is:

Lemma 5. g_A *is rotationally invariant if and only if span(A)=span($\boldsymbol{\Sigma}_d$). In other words,* PCA *is the only rotationally invariant projection.*

IV.C Chernoff information

We now introduce the notion of the Chernoff information, which serves as our surrogate measure for the Bayes error of any classification procedure given the *projected* data – in the context of this paper the projection is via LOL or PCA. Our discussion of the Chernoff information is under the context of decision rules for hypothesis testing, nevertheless, as evidenced by the fact that the Maximum A Posterior decision rule – equivalently the Bayes classifier – achieves the Chernoff information rate, this distinction between hypothesis testing and classification is mainly for ease of exposition.

Let F_0 and F_1 be two absolutely continuous multivariate distribution in $\Omega \subset \mathbb{R}^d$ with density function f_0 and f_1, respectively. Suppose that $Y_1, Y_2, \ldots, Y_m$ are independent and identically distributed random variables, with Y_i distributed either F_0 or F_1. We are interested in testing the simple null hypothesis $\mathbb{H}_0 \colon F = F_0$ against the simple alternative hypothesis $\mathbb{H}_1 \colon F = F_1$. A test T is a sequence of mapping $T_m \colon \Omega^m \mapsto \{0, 1\}$ such that given $Y_1 = y_1, Y_2 = y_2, \ldots, Y_m = y_m$, the test rejects $\mathbb{H}_0$ in favor of $\mathbb{H}_1$ if $T_m(y_1, y_2, \ldots, y_m) = 1$; similarly, the test rejects $\mathbb{H}_1$ in favor of $\mathbb{H}_0$ if $T_m(y_1, y_2, \ldots, y_m) = 0$. The Neyman-Pearson lemma states that, given $Y_1 = y_1, Y_2 = y_2, \ldots, Y_m = y_m$ and a threshold $\eta_m \in \mathbb{R}$, the likelihood ratio test which rejects $\mathbb{H}_0$ in favor of $\mathbb{H}_1$ whenever

$$\left(\sum_{i=1}^m \log f_0(y_i) - \sum_{i=1}^m \log f_1(y_i)\right) \leq \eta_m$$

is the most powerful test at significance level $\alpha_m = \alpha(\eta_m)$, i.e., the likelihood ratio test minimizes the type-II error β_m subject to the contraint that the type-I error is at most α_m.

Assuming that $\pi \in (0, 1)$ is a prior probability that $\mathbb{H}_0$ is true. Then, for a given $\alpha_m^* \in (0, 1)$, let $\beta_m^* = \beta_m^*(\alpha_m^*)$ be the type-II error associated with the likelihood ratio test when the type-I error is at most α_m^*. The quantity $\inf_{\alpha_m^* \in (0,1)} \pi\alpha_m^* + (1-\pi)\beta_m^*$ is then the Bayes risk in deciding between $\mathbb{H}_0$ and $\mathbb{H}_1$ given the m independent random variables $Y_1, Y_2, \ldots, Y_m$. A classical result of Chernoff [42] states that the Bayes risk is intrinsically linked to a quantity known as the *Chernoff information*. More specifically, let $C(F_0, F_1)$ be the quantity

$$C(F_0, F_1) = -\log\left[\inf_{t \in (0,1)} \int_{\mathbb{R}^d} f_0^t(\boldsymbol{x})f_1^{1-t}(\boldsymbol{x})\mathrm{d}\boldsymbol{x}\right]$$
$$= \sup_{t \in (0,1)}\left[-\log \int_{\mathbb{R}^d} f_0^t(\boldsymbol{x})f_1^{1-t}(\boldsymbol{x})\mathrm{d}\boldsymbol{x}\right] \tag{4}$$

Then we have

$$\lim_{m \to \infty}\frac{1}{m}\inf_{\alpha_m^* \in (0,1)} \log(\pi\alpha_m^* + (1-\pi)\beta_m^*) = -C(F_0, F_1). \tag{5}$$

Thus $C(F_0, F_1)$ is the *exponential* rate at which the Bayes error $\inf_{\alpha_m^* \in (0,1)} \pi\alpha_m^* + (1-\pi)\beta_m^*$ decreases as $m \to \infty$; we also note that the $C(F_0, F_1)$ is independent of π. We also define, for a given $t \in (0, 1)$ the Chernoff divergence $C_t(F_0, F_1)$ between F_0 and F_1 by

$$C_t(F_0, F_1) = -\log \int_{\mathbb{R}^d} f_0^t(\boldsymbol{x})f_1^{1-t}(\boldsymbol{x})\mathrm{d}\boldsymbol{x}.$$

505 The Chernoff divergence is an example of a f-divergence as defined in [43]. When $t = 1/2$, $C_t(F_0, F_1)$ is
506 the Bhattacharyya distance between F_0 and F_1.

507 The result of Eq. (5) can be extended to $K + 1 \geq 2$ hypothesis, with the exponential rate being the min-
508 imum of the Chernoff information between any pair of hypothesis. More specifically, let $F_0, F_1, \ldots, F_K$ be
509 distributions on $\mathbb{R}^d$ and let $Y_1, Y_2, \ldots, Y_m$ be independent and identically distributed random variables with
510 distribution $F \in \{F_0, F_1, \ldots, F_K\}$. Our inference task is in determining the distribution of the Y_i among the
511 $K + 1$ hypothesis $\mathbb{H}_0 \colon F = F_0, \ldots, \mathbb{H}_K \colon F = F_K$. Suppose also that hypothesis $\mathbb{H}_k$ has *a priori* probability
512 π_k. For any decision rule δ, the risk of δ is $r(\delta) = \sum_k \pi_k \sum_{l \neq k} \alpha_{lk}(\delta)$ where $\alpha_{lk}(\delta)$ is the probability of
513 accepting hypothesis $\mathbb{H}_l$ when hypothesis $\mathbb{H}_k$ is true. Then we have [44]

$$\inf_{\delta} \lim_{m \to \infty} \frac{r(\delta)}{m} = -\min_{k \neq l} C(F_k, F_l). \tag{6}$$

514 where the infimum is over all decision rule δ, i.e., for any δ, $r(\delta)$ decreases to 0 as $m \to \infty$ at a rate no
515 faster than $\exp(-m \min_{k \neq l} C(F_k, F_l))$.

When the distributions F_0 and F_1 are multivariate normal, that is, $F_0 = \mathcal{N}(\mu_0, \Sigma_0)$ and $F_1 = \mathcal{N}(\mu_1, \Sigma_1)$; then, denoting by $\Sigma_t = t\Sigma_0 + (1-t)\Sigma_1$, we have

$$C(F_0, F_1) = \sup_{t \in (0,1)} \left(\frac{t(1-t)}{2}(\mu_1 - \mu_2)^\top \Sigma_t^{-1}(\mu_1 - \mu_2) + \frac{1}{2} \log \frac{|\Sigma_t|}{|\Sigma_0|^t |\Sigma_1|^{1-t}} \right).$$

516 IV.D Projecting data and Chernoff information

517 We now discuss how the Chernoff information characterizes the effect a linear transformation A of the data
518 has on classification accuracy. We start with the following simple result whose proof follows directly from
519 Eq. (6).

520 **Lemma 6.** *Let $F_0 = \mathcal{N}(\mu_0, \Sigma)$ and $F_1 \sim \mathcal{N}(\mu_1, \Sigma)$ be two multivariate normals with equal covariance*
521 *matrices. For any linear transformation A, let $F_0^{(A)}$ and $F_1^{(A)}$ denotes the distribution of AX when $X \sim F_0$*
522 *and $X \sim F_1$, respectively. We then have*

$$
\begin{aligned}
C(F_0^{(A)}, F_1^{(A)}) &= \frac{1}{8}(\mu_1 - \mu_0)^\top A^\top (A\Sigma A^\top)^{-1} A(\mu_1 - \mu_0) \\
&= \frac{1}{8}(\mu_1 - \mu_0)^\top \Sigma^{-1/2} \Sigma^{1/2} A^\top (A\Sigma A^\top)^{-1} A\Sigma^{1/2} \Sigma^{-1/2}(\mu_1 - \mu_0) \\
&= \frac{1}{8}\|P_{\Sigma^{1/2}A^\top} \Sigma^{-1/2}(\mu_1 - \mu_0)\|_F^2
\end{aligned}
\tag{7}
$$

523 *where $P_Z = Z(Z^\top Z)^{-1} Z^\top$ denotes the matrix corresponding to the orthogonal projection onto the columns*
524 *of Z.*

Thus for a classification problem where $X|Y = 0$ and $X|Y = 1$ are distributed multivariate normals with mean μ_0 and μ_1 and the same covariance matrix Σ, Lemma 6 then states that for any two linear transformations A and B, the transformed data AX is to be preferred over the transformed data BX if

$$(\mu_1 - \mu_0)^\top A^\top (A\Sigma A^\top)^{-1} A(\mu_1 - \mu_0) > (\mu_1 - \mu_0)^\top B^\top (B\Sigma B^\top)^{-1} B(\mu_1 - \mu_0).$$

As an example, suppose Σ is diagonal with distinct eigenvalues where the diagonal entries of Σ are in non-increasing order. Denote by $\delta = \mu_1 - \mu_0$ and let $A = \delta^\top$ and $B = e_1^\top = (1, 0, 0, \ldots, 0)$ be the linear transformations for LoL and PCA of X into $\mathbb{R}$. We then have

$$C(F_0^{(A)}, F_1^{(A)}) = \frac{(\delta^\top \delta)^2}{\delta^\top \Sigma \delta} = \frac{(\sum_i \delta_i^2)^2}{\sum_i \delta_i^2 \lambda_i}; \quad C(F_0^{(B)}, F_1^{(B)}) = \frac{\delta_1^2}{\lambda_1}$$

where λ_1 is the largest eigenvalue of Σ. Suppose furthermore that $\delta_1 \leq \delta_2 \leq \cdots \leq \delta_p$ and $\lambda_1 > \lambda_2 > \cdots > \lambda_p$. Then $C(F_0^{(A)}, F_1^{(A)})$ can be lower-bounded as

$$C(F_0^{(A)}, F_1^{(A)}) = \frac{(\sum_i \delta_i^2)^2}{\sum_i \delta_i^2 \lambda_i} \geq \frac{(p\delta_1^2)^2}{p\delta_p^2 \lambda_1}$$

525 and hence $C(F_0^{(A)}, F_1^{(A)}) > C(F_0^{(B)}, F_1^{(B)})$ provided $p\delta_1^2 \geq \delta_p^2$.

When $A = [\delta \mid e_1 \mid e_2 \cdots \mid e_{d-1}]^\top \in \mathbb{R}^{d \times p}$ and $B = [e_1 \mid e_2 \mid \cdots \mid e_d]^\top \in \mathbb{R}^{d \times p}$ are the linear transformation for LoL and PCA of X into $\mathbb{R}^d$, we have

$$C(F_0^{(A)}, F_1^{(A)}) = \frac{(\sum_{i=d}^p \delta_i^2)^2}{\sum_{i=d}^p \delta_i^2 \lambda_i} + \sum_{i=1}^{d-1} \frac{\delta_i^2}{\lambda_i}; \quad C(F_0^{(B)}, F_1^{(B)}) = \sum_{i=1}^d \frac{\delta_i^2}{\lambda_i}.$$

526 This can be seen as follows. Let $\xi_{d-1} = [e_1 \mid e_2 \mid \cdots \mid e_{d-1}] \in \mathbb{R}^{p \times (d-1)}$ and $\zeta_{d-1} = (\lambda_1 \delta_1, \lambda_2 \delta_2, \ldots, \lambda_{d-1} \delta_{d-1})^\top \in$
527 $\mathbb{R}^{d-1}$. Then

$$A\Sigma A^\top = [\delta \mid \xi_{d-1}]^\top \Sigma [\delta \mid \xi_{d-1}] = \begin{bmatrix} \delta^\top \Sigma \delta & \delta^\top \Sigma \xi_{d-1} \\ \xi_{d-1}^\top \Sigma \delta & \xi_{d-1}^\top \Sigma \xi_{d-1} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^p \delta_i^2 \lambda_i & \zeta_{d-1}^\top \\ \zeta_{d-1} & \Sigma_{d-1} \end{bmatrix} \tag{8}$$

528 where $\Sigma_{d-1} = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_{d-1})$ is the submatrix of Σ corresponding to the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_{d-1}$.
529 Using a formula for the inverse of a partitioned matrix, we have

$$(A\Sigma A^\top)^{-1} = \begin{bmatrix} \sum_{i=1}^p \delta_i^2 \lambda_i & \zeta_{d-1}^\top \\ \zeta_{d-1} & \Sigma_{d-1} \end{bmatrix}^{-1}$$

$$= \begin{bmatrix} \left(\sum_{i=1}^p \delta_i^2 \lambda_i - \zeta_{d-1}^\top \Sigma_{d-1}^{-1} \zeta_{d-1}\right)^{-1} & -\left(\sum_i \delta_i^2 \lambda_i - \zeta_{d-1}^\top \Sigma_{d-1}^{-1} \zeta_{d-1}\right)^{-1} \zeta_{d-1}^\top \Sigma_{d-1}^{-1} \\ -\Sigma_{d-1}^{-1} \zeta_{d-1} \left(\sum_i \delta_i^2 \lambda_i - \zeta_{d-1}^\top \Sigma_{d-1}^{-1} \zeta_{d-1}\right)^{-1} & \Sigma_{d-1}^{-1} + \Sigma_{d-1}^{-1} \zeta_{d-1} \left(\sum_i \delta_i^2 \lambda_i - \zeta_{d-1}^\top \Sigma_{d-1}^{-1} \zeta_{d-1}\right)^{-1} \zeta_{d-1}^\top \Sigma_{d-1}^{-1} \end{bmatrix} \tag{9}$$

530 Now, $\sum_{i=1}^p \delta_i^2 \lambda_i - \zeta_{d-1}^\top \Sigma_{d-1}^{-1} \zeta_{d-1} = \sum_{i=1}^p \delta_i^2 - \sum_{i=1}^{d-1} \delta_i^2 \lambda_i = \sum_{i=d}^p \delta_i^2 \lambda_i$. Therefore,

$$(A\Sigma A^\top)^{-1} = \begin{bmatrix} \left(\sum_{i=d}^p \delta_i^2 \lambda_i\right)^{-1} & -\frac{\zeta_{d-1}^\top \Sigma_{d-1}^{-1}}{\sum_{i=d}^p \delta_i^2 \lambda_i} \\ -\frac{\Sigma_{d-1}^{-1} \zeta_{d-1}}{\sum_{i=d}^p \delta_i^2 \lambda_i} & \Sigma_{d-1}^{-1} + \frac{\Sigma_{d-1}^{-1} \zeta_{d-1} \zeta_{d-1}^\top \Sigma_{d-1}^{-1}}{\sum_{i=d}^p \delta_i^2 \lambda_i} \end{bmatrix}. \tag{10}$$

In addition, $A(\mu_1 - \mu_0) = (\delta^\top \delta, \delta_1, \delta_2, \ldots, \delta_{d-1})^\top = (\delta^\top \delta, \zeta_{d-1}^\top \Sigma_{d-1}^{-1})^\top \in \mathbb{R}^d$. Hence

$$(\mu_1 - \mu_0)^\top A^\top (A\Sigma A^\top)^{-1} A(\mu_1 - \mu_0) = [\delta^\top \delta \mid \zeta_{d-1}^\top \Sigma_{d-1}^{-1}] \begin{bmatrix} \left(\sum_{i=d}^p \delta_i^2 \lambda_i\right)^{-1} & -\frac{\zeta_{d-1}^\top \Sigma_{d-1}^{-1}}{\sum_{i=d}^p \delta_i^2 \lambda_i} \\ -\frac{\Sigma_{d-1}^{-1} \zeta_{d-1}}{\sum_{i=d}^p \delta_i^2 \lambda_i} & \Sigma_{d-1}^{-1} + \frac{\Sigma_{d-1}^{-1} \zeta_{d-1} \zeta_{d-1}^\top \Sigma_{d-1}^{-1}}{\sum_{i=d}^p \delta_i^2 \lambda_i} \end{bmatrix} \begin{bmatrix} \delta^\top \delta \\ \Sigma_{d-1}^{-1} \zeta_{d-1} \end{bmatrix}$$

$$= \frac{(\delta^\top \delta)^2}{\sum_{i=d}^p \delta_i^2 \lambda_i} - 2\delta^\top \delta \frac{\zeta_{d-1}^\top \Sigma_{d-1}^{-2} \zeta_{d-1}}{\sum_{i=d}^p \delta_i^2 \lambda_i} + \left(\zeta_{d-1}^\top \Sigma_{d-1}^{-3} \zeta_{d-1} + \frac{(\zeta_{d-1}^\top \Sigma_{d-1}^{-2} \zeta_{d-1})^2}{\sum_{i=d}^p \delta_i^2 \lambda_i}\right)$$

$$= \frac{(\delta^\top \delta - \zeta_{d-1}^\top \Sigma_{d-1}^{-2} \zeta_{d-1})^2}{\sum_{i=d}^p \delta_i^2 \lambda_i} + \zeta_{d-1}^\top \Sigma_{d-1}^{-3} \zeta_{d-1}$$

$$= \frac{\left(\sum_{i=1}^p \delta_i^2 - \sum_{i=1}^{d-1} \delta_i^2\right)^2}{\sum_{i=d}^p \delta_i^2 \lambda_i} + \sum_{i=1}^{d-1} \frac{\delta_i^2}{\lambda_i} = \frac{\left(\sum_{i=d}^p \delta_i^2\right)^2}{\sum_{i=d}^p \delta_i^2 \lambda_i} + \sum_{i=1}^{d-1} \frac{\delta_i^2}{\lambda_i}.$$

The derivation of $C(F_0^{(B)}, F_1^{(B)})$ is straightforward and will be omitted. We therefore have

$$C(F_0^{(A)}, F_1^{(A)}) - C(F_0^{(B)}, F_1^{(B)}) = \frac{\left(\sum_{i=d}^p \delta_i^2\right)^2}{\sum_{i=d}^p \delta_i^2 \lambda_i} - \frac{\delta_d^2}{\lambda_d}.$$

D. LDA

From our assumption that $\lambda_d \geq \lambda_{d+1} \geq \cdots \geq \lambda_p$, we have $\sum_{i=d}^{p} \delta_i^2 \lambda_i \leq \lambda_d \sum_{i=d}^{p} \delta_i^2$ and hence

$$\frac{\left(\sum_{i=d}^{p} \delta_i^2\right)^2}{\sum_{i=d}^{p} \delta_i^2 \lambda_i} \geq \frac{\left(\sum_{i=d}^{p} \delta_i^2\right)^2}{\lambda_d \sum_{i=d}^{p} \delta_i^2} = \frac{1}{\lambda_d} \sum_{i=d}^{p} \delta_i^2 \geq \frac{\delta_d^2}{\lambda_d}$$

and hence $C(F_0^{(A)}, F_1^{(A)}) \geq C(F_0^{(B)}, F_1^{(B)})$ always, and the inequality is strict provided that $\sum_{d+1}^{p} \delta_d^2 > 0$.

Finally we consider the case where Σ is an arbitrary covariance matrix. Let $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p$ be the eigenvalues of Σ and let $u_1, u_2, \ldots, u_p$ be the corresponding eigenvectors. For $d \leq p$, let $U_{d-1} = \begin{bmatrix} u_1 \mid u_2 \mid \cdots \mid u_{d-1} \end{bmatrix} \in \mathbb{R}^{p \times (d-1)}$ be the matrix whose columns are the eigenvectors $u_1, u_2, \ldots, u_{d-1}$ of Σ. Then the LOL projection matrix into $\mathbb{R}^d$ is given by $A = \begin{bmatrix} \delta | U_{d-1} \end{bmatrix}^\top$. We first have

$$A\Sigma A^\top = \begin{bmatrix} \delta | U_{d-1} \end{bmatrix}^\top \Sigma \begin{bmatrix} \delta | U_{d-1} \end{bmatrix} = \begin{bmatrix} \delta^\top \Sigma & \delta^\top \Sigma U_{d-1} \\ U_{d-1}^\top \Sigma \delta & U_{d-1}^\top \Sigma U_{d-1} \end{bmatrix} = \begin{bmatrix} \delta^\top \Sigma & \delta^\top \Sigma U_{d-1} \\ U_{d-1}^\top \Sigma \delta & \Lambda_{d-1} \end{bmatrix}$$

where $\Lambda_{d-1} = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_{d-1})$ is the $(d-1) \times (d-1)$ diagonal matrix formed by the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_{d-1}$. Therefore, letting $\gamma = \delta^\top \Sigma \delta - \delta^\top \Sigma U_{d-1} \Lambda_{d-1}^{-1} U_{d-1}^\top \Sigma \delta$, we have

$$(A\Sigma A^\top)^{-1} = \begin{bmatrix} \delta^\top \Sigma \delta & \delta^\top \Sigma U_{d-1} \\ U_{d-1}^\top \Sigma \delta & U_{d-1}^\top \Sigma U_{d-1} \end{bmatrix}^{-1}$$

$$= \begin{bmatrix} \gamma^{-1} & -\delta^\top \Sigma U_{d-1} \Lambda_{d-1}^{-1} \gamma^{-1} \\ -\Lambda_{d-1}^{-1} U_{d-1}^\top \Sigma \delta \gamma^{-1} & \left(\Lambda_{d-1} - \frac{U_{d-1}^\top \Sigma \delta \delta^\top \Sigma U_{d-1}}{\delta^\top \Sigma \delta}\right)^{-1} \end{bmatrix}.$$

The Sherman-Morrison-Woodbury formula then implies

$$\left(\Lambda_{d-1} - \frac{U_{d-1}^\top \Sigma \delta \delta^\top \Sigma U_{d-1}}{\delta^\top \Sigma \delta}\right)^{-1} = \Lambda_{d-1}^{-1} + \frac{\Lambda_{d-1}^{-1} U_{d-1}^\top \Sigma \delta \delta^\top \Sigma U_{d-1} \Lambda_{d-1}^{-1} / (\delta^\top \Sigma \delta)}{1 - \delta^\top \Sigma U_{d-1} \Lambda_{d-1}^{-1} U_{d-1}^\top \Sigma \delta / (\delta^\top \Sigma \delta)}$$

$$= \Lambda_{d-1}^{-1} + \frac{\Lambda_{d-1}^{-1} U_{d-1}^\top \Sigma \delta \delta^\top \Sigma U_{d-1} \Lambda_{d-1}^{-1}}{\delta^\top \Sigma \delta - \delta^\top \Sigma U_{d-1} \Lambda_{d-1}^{-1} U_{d-1}^\top \Sigma \delta}$$

$$= \Lambda_{d-1}^{-1} + \gamma^{-1} \Lambda_{d-1}^{-1} U_{d-1}^\top \Sigma \delta \delta^\top \Sigma U_{d-1} \Lambda_{d-1}^{-1}$$

We note that $\Sigma U_{d-1} = U_{d-1} \Lambda_{d-1}$ and $U_{d-1}^\top \Sigma = \Lambda_{d-1} U_{d-1}^\top$ and hence

$$\gamma = \delta^\top \Sigma \delta - \delta^\top \Sigma U_{d-1} \Lambda_{d-1}^{-1} U_{d-1}^\top \Sigma \delta = \delta^\top \Sigma \delta - \delta^\top U_{d-1} \Lambda_{d-1} \Lambda_{d-1}^{-1} \Lambda_{d-1} U_{d-1}^\top \delta$$

$$= \delta^\top \Sigma \delta - \delta^\top U_{d-1} \Lambda_{d-1} U_{d-1}^\top \delta = \delta^\top (\Sigma - \Sigma_{d-1}) \delta$$

where $\Sigma_{d-1} = U_{d-1} \Lambda_{d-1} U_{d-1}^\top$ is the best rank $d-1$ approximation to Σ with respect to any unitarily invariant norm. In addition,

$$\Lambda_{d-1}^{-1} U_{d-1}^\top \Sigma \delta \delta^\top \Sigma U_{d-1} \Lambda_{d-1}^{-1} = \Lambda_{d-1}^{-1} \Lambda_{d-1} U_{d-1}^\top \delta \delta^\top U_{d-1} \Lambda_{d-1} \Lambda_{d-1}^{-1} = U_{d-1}^\top \delta \delta^\top U_{d-1}.$$

We thus have

$$(A\Sigma A^\top)^{-1} = \begin{bmatrix} \gamma^{-1} & -\delta^\top \Sigma U_{d-1} \Lambda_{d-1}^{-1} \gamma^{-1} \\ -\Lambda_{d-1}^{-1} U_{d-1}^\top \Sigma \delta \gamma^{-1} & \left(\Lambda_{d-1} - \frac{U_{d-1}^\top \Sigma \delta \delta^\top \Sigma U_{d-1}}{\delta^\top \Sigma \delta}\right)^{-1} \end{bmatrix} = \begin{bmatrix} \gamma^{-1} & -\gamma^{-1} \delta^\top U_{d-1} \\ -\gamma^{-1} U_{d-1}^\top \delta & \Lambda_{d-1}^{-1} + \gamma^{-1} U_{d-1}^\top \delta \delta^\top U_{d-1} \end{bmatrix}.$$

Therefore,

$$\delta^\top A^\top (A\Sigma A^\top)^{-1} A\delta = \delta^\top \begin{bmatrix} \delta \mid U_{d-1} \end{bmatrix} \begin{bmatrix} \gamma^{-1} & -\gamma^{-1} \delta^\top U_{d-1} \\ -\gamma^{-1} U_{d-1}^\top \delta & \Lambda_{d-1}^{-1} + \gamma^{-1} U_{d-1}^\top \delta \delta^\top U_{d-1} \end{bmatrix} \begin{bmatrix} \delta | U_{d-1} \end{bmatrix}^\top \delta$$

$$= \begin{bmatrix} \delta^\top \delta \mid \delta^\top U_{d-1} \end{bmatrix} \begin{bmatrix} \gamma^{-1} & -\gamma^{-1} \delta^\top U_{d-1} \\ -\gamma^{-1} U_{d-1}^\top \delta & \Lambda_{d-1}^{-1} + \gamma^{-1} U_{d-1}^\top \delta \delta^\top U_{d-1} \end{bmatrix} \begin{bmatrix} \delta^\top \delta \\ U_{d-1}^\top \delta \end{bmatrix}$$

$$= \gamma^{-1} (\delta^\top \delta)^2 - 2\gamma^{-1} \delta^\top \delta \delta^\top U_{d-1} U_{d-1}^\top \delta + \delta^\top U_{d-1} (\Lambda_{d-1}^{-1} + \gamma^{-1} U_{d-1}^\top \delta \delta^\top U_{d-1}) U_{d-1}^\top \delta$$

$$= \gamma^{-1} (\delta^\top \delta - \delta^\top U_{d-1} U_{d-1}^\top \delta)^2 + \delta^\top U_{d-1} \Lambda_{d-1}^{-1} U_{d-1}^\top \delta$$

$$= \gamma^{-1} (\delta^\top (I - U_{d-1} U_{d-1}^\top) \delta)^2 + \delta^\top \Sigma_{d-1}^\dagger \delta$$

22

where $\Sigma_{d-1}^{\dagger}$ is the Moore-Penrose pseudo-inverse of Σ_{d-1}. The PCA projection matrix into $\mathbb{R}^d$ is given by $B = U_d^{\top}$ and hence

$$\delta^{\top} B^{\top} (B\Sigma B^{\top})^{-1} B\delta = \delta^{\top} U_d \Lambda_d^{-1} U_d^{\top} \delta = \delta^{\top} \Sigma_d^{\dagger} \delta. \tag{11}$$

We thus have

$$
\begin{aligned}
C(F_0^{(A)}, F_1^{(A)}) - C(F_0^{(B)}, F_1^{(B)}) &= \gamma^{-1}(\delta^{\top}(I - U_{d-1}U_{d-1}^{\top})\delta)^2 - \delta^{\top}(\Sigma_d^{\dagger} - \Sigma_{d-1}^{\dagger})\delta \\
&= \frac{(\delta^{\top}(I - U_{d-1}U_{d-1}^{\top})\delta)^2}{\delta^{\top}(\Sigma - \Sigma_{d-1})\delta} - \delta^{\top}(\Sigma_d^{\dagger} - \Sigma_{d-1}^{\dagger})\delta \\
&\geq \frac{(\delta^{\top}(I - U_{d-1}U_{d-1}^{\top})\delta)^2}{\lambda_d \delta^{\top}(I - U_{d-1}U_{d-1}^{\top})\delta} - \frac{1}{\lambda_d}\delta^{\top} u_d u_d^{\top}\delta \\
&= \frac{1}{\lambda_d}\delta^{\top}(I - U_{d-1}U_{d-1})\delta - \frac{1}{\lambda_d}\delta^{\top}(U_d U_d^{\top} - U_{d-1}U_{d-1}^{\top})\delta \geq 0
\end{aligned}
$$

where we recall that u_d is the d-th column of U_d. Thus $C(F_0^{(A)}, F_1^{(A)}) \geq C(F_0^{(B)}, F_1^{(B)})$ always, and the inequality is strict whenever $\delta^{\top}(I - U_d U_d^{\top})\delta > 0$.

Next we consider the case when $A = [\delta | U_{d-1}]^{\top}$ and $B = U_d$ where U_d is an arbitrary $p \times d$ matrix with $U_d^{\top} U_d = I$, i.e., U_d has d orthonornal columns, and U_{d-1} is the first $d-1$ columns of U_d. A similar derivation to the above yields

$$C(F_0^{(A)}, F_1^{(A)}) = \frac{(\delta^{\top}\Sigma^{-1/2}(I - V_{d-1}V_{d-1}^{\top})\Sigma^{1/2}\delta)^2}{\delta^{\top}\Sigma^{1/2}(I - V_{d-1}V_{d-1}^{\top})\Sigma^{1/2}\delta} + \delta^{\top}\Sigma^{-1/2}V_{d-1}V_{d-1}^{\top}\Sigma^{-1/2}\delta \tag{12}$$

$$C(F_0^{(B)}, F_1^{(B)}) = \delta^{\top}\Sigma^{-1/2}V_d V_d^{\top}\Sigma^{-1/2}\delta \tag{13}$$

where $V_d V_{d-1}^{\top} = \Sigma^{1/2}U_d(U_d^{\top}\Sigma U_d)^{-1}U_d^{\top}\Sigma^{1/2}$ is the orthogonal projection onto the column space of $\Sigma^{1/2}U_d$. Hence $C(F_0^{(A)}, F_1^{(A)}) > C(F_0^{(B)}, F_1^{(B)})$ if and only if

$$\frac{(\delta^{\top}\Sigma^{-1/2}(I - V_{d-1}V_{d-1}^{\top})\Sigma^{1/2}\delta)^2}{\delta^{\top}\Sigma^{1/2}(I - V_{d-1}V_{d-1}^{\top})\Sigma^{1/2}\delta} > \delta^{\top}\Sigma^{-1/2}(V_d V_d^{\top} - V_{d-1}V_{d-1}^{\top})\Sigma^{-1/2}\delta.$$

Let $C(F_0^{(A)}, F_1^{(A)})$ and $C(F_0^{(B)}, F_1^{(B)})$ be the Chernoff informations when $A = [\delta \mid U_{d-1}]^{\top}$ and $B = [\delta \mid U_d]^{\top}$ where U_{d-1} and U_d contain the eignevectors of the population covariance matrices Σ; similarly, let $\hat{C}(F_0^{(A)}, F_1^{(A)})$ and $\hat{C}(F_0^{(B)}, F_1^{(B)})$ be the Chernoff informations when $A = [\delta \mid \hat{U}_{d-1}]$ and $B = [\delta \mid \hat{U}_d]$ where $\hat{U}_{d-1}$ and $\hat{U}_d$ contain the eigenvectors of the *sample* covariance matrices $\hat{\Sigma}$. Suppose furthermore that $C(F_0^{(A)}, F_1^{(A)}) > C(F_0^{(B)}, F_1^{(B)})$. Then for sufficiently large n, with high probability, $\hat{C}(F_0^{(A)}, F_1^{(A)}) > \hat{C}(F_0^{(B)}, F_1^{(B)})$.

Finally we consider the case when $B = \tilde{U}_d^{\top}$ where $\tilde{U}_d$ is the $p \times d$ matrix whose columns are the d largest eigenvectors of the *pooled* covariance matrix $\tilde{\Sigma} = \mathbb{E}[(X - \frac{\mu_0 + \mu_1}{2})(X - \frac{\mu_0 + \mu_1}{2})^{\top}]$. Assume, without loss of generality, that $\mu_1 = -\mu_0 = \mu$. We then have

$$\tilde{\Sigma} = \mathbb{E}[XX^{\top}] = \pi\Sigma + \pi\mu_0\mu_0^{\top} + (1-\pi)\Sigma + (1-\pi)\mu_1\mu_1^{\top} = \Sigma + \mu\mu^{\top} = \Sigma + \tfrac{1}{4}\delta\delta^{\top}.$$

Therefore

$$(B\Sigma B^{\top})^{-1} = (\tilde{U}_d^{\top}\Sigma\tilde{U}_d)^{-1} = (\tilde{U}_d^{\top}(\tilde{\Sigma} - \tfrac{1}{4}\delta\delta^{\top})\tilde{U}_d)^{-1} = (\tilde{S}_d - \tfrac{1}{4}\tilde{U}_d^{\top}\delta\delta^{\top}\tilde{U}_d)^{-1} = \tilde{S}_d^{-1} + \frac{\tilde{S}_d^{-1}\tilde{U}_d^{\top}\delta\delta^{\top}\tilde{U}_d\tilde{S}_d^{-1}}{4 - \delta^{\top}\tilde{U}_d\tilde{S}_d^{-1}\tilde{U}_d^{\top}\delta}$$

548 where $\tilde{S}_d$ is the diagonal matrix containing the d largest eigenvalues of $\tilde{\Sigma}$. Hence

$$C(F_0^{(B)}, F_1^{(B)}) = \delta^\top B^\top (B\Sigma B^\top)^{-1} B\delta = \delta^\top \tilde{U}_d \Big(\tilde{S}_d^{-1} + \frac{\tilde{S}_d^{-1}\tilde{U}_d^\top \delta\delta^\top \tilde{U}_d \tilde{S}_d^{-1}}{4 - \delta^\top \tilde{U}_d \tilde{S}_d^{-1}\tilde{U}_d^\top \delta} \Big) \tilde{U}_d^\top \delta$$

$$= \delta^\top \tilde{U}_d \tilde{S}_d^{-1}\tilde{U}_d^\top \delta + \frac{(\delta^\top \tilde{U}_d \tilde{S}_d^{-1}\tilde{U}_d^\top \delta)^2}{4 - \delta^\top \tilde{U}_d \tilde{S}_d^{-1}\tilde{U}_d^\top \delta} \qquad (14)$$

$$= \delta^\top \tilde{\Sigma}_d^\dagger \delta + \frac{(\delta^\top \tilde{\Sigma}_d^\dagger \delta)^2}{4 - \delta^\top \tilde{\Sigma}_d^\dagger \delta} = \frac{4\delta^\top \tilde{\Sigma}_d^\dagger \delta}{4 - \delta^\top \tilde{\Sigma}_d^\dagger \delta}.$$

549 where $\tilde{\Sigma}_d = \tilde{U}_d \tilde{S}_d \tilde{U}_d^\top$ is the best rank d approximation to $\tilde{\Sigma} = \Sigma + \frac{1}{4}\delta\delta^\top$.

We recall that the LoL projection $A = [\delta \mid U_{d-1}]^\top$ yields

$$C(F_0^{(A)}, F_1^{(A)}) = \frac{(\delta^\top (I - U_{d-1}U_{d-1}^\top)\delta)^2}{\delta^\top (\Sigma - \Sigma_{d-1})\delta} + \delta^\top \Sigma_{d-1}^\dagger \delta.$$

550 To illustrate the difference between the LoL projection and that based on the eigenvectors of the *pooled*
551 covariance matrix, consider the following simple example. Let $\Sigma = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_p)$ be a diagonal matrix
552 with $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p$. Also let $\delta = (0, 0, \ldots, 0, s)$. Suppose furthermore that $\lambda_p + s^2/4 < \lambda_d$. Then we
553 have $\tilde{\Sigma}_d = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_d, 0, 0, \ldots, 0)$. Thus $\tilde{\Sigma}_d^\dagger = \text{diag}(1/\lambda_1, 1/\lambda_2, \ldots, 1/\lambda_d, 0, 0, \ldots, 0)$ and $\delta^\dagger \tilde{\Sigma}_d^\dagger \delta = 0$.
554 Therefore, $C(F_0^{(B)}, F_1^{(B)}) = 0$.

On the other hand, we have

$$C(F_0^{(A)}, F_1^{(A)}) = \frac{(\delta^\top (I - U_{d-1}U_{d-1}^\top)\delta)^2}{\delta^\top (\Sigma - \Sigma_{d-1})\delta} + \delta^\top \Sigma_{d-1}^\dagger \delta = \frac{s^4}{s^2 \lambda_p} + 0 = s^2/\lambda_p.$$

We can generalize the previous example as follows. Let Σ be a $p \times p$ covariance matrix of the form

$$\Sigma = \begin{bmatrix} \Sigma_d & 0 \\ 0 & \Sigma_d^\perp \end{bmatrix}$$

where Σ_d is a $d \times d$ matrix. Let $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p$ be the eigenvalues of Σ and suppose furthermore that the
eigenvalues of Σ_d are $\lambda_1, \lambda_2, \ldots, \lambda_d$. Let $\gamma = \lambda_d - \lambda_{d+1}$. We now assume that δ are generated randomly as
follows. The entries of δ are i.i.d. random variable sampled according to the following distribution. Given an
index i, with probability ϵ, $\delta_i = 0$ and with probability $1 - \epsilon$, δ_i is distributed according to a normal distribution
with mean $\tau > 0$ and variance σ^2. Then with probability at least ϵ^d, the covariance matrix for $\tilde{\Sigma}$ is of the form

$$\tilde{\Sigma} = \begin{bmatrix} \Sigma_d & 0 \\ 0 & \Sigma_d^\perp + \frac{1}{4}(\tilde{\delta}\tilde{\delta}^\top) \end{bmatrix}$$

where $\tilde{\delta} \in \mathbb{R}^{p-d}$ is formed by excluding the first d elements of δ. Now, if $\lambda_{d+1} + \frac{1}{4}\|\tilde{\delta}\|^2 < \lambda_d$, then the d largest
eigenvalues of $\tilde{\Sigma}$ is still $\lambda_1, \lambda_2, \ldots, \lambda_d$, and thus the eigenvectors corresponding to the d largest eigenvalues
of $\tilde{\Sigma}$ is the same as that for the d largest eigenvalues of Σ. That is to say,

$$\lambda_{d+1} + \tfrac{1}{4}\|\tilde{\delta}\|^2 < \lambda_d \implies \tilde{\Sigma}_d^\dagger = \Sigma_d^\dagger \implies \delta^\top \tilde{\Sigma}_d^\dagger \delta = 0 \implies C(F_0^{(B)}, F_1^{(B)}) = 0.$$

We now compute the probability that $\lambda_{d+1} + \frac{1}{4}\|\tilde{\delta}\|^2 < \lambda_d$. Suppose for the moment that $\epsilon > 0$ is fixed and
do not varies with p. We then have

$$\frac{\sum_{i=d+1}^p \delta_i^2 - (p-d)(1-\epsilon)\tau^2}{\sqrt{(p-d)(2(1-\epsilon)(2\tau^2\sigma^2 + \sigma^4) + \epsilon(1-\epsilon)(\tau^4 + 2\tau^2\sigma^2 + \sigma^4))}} \xrightarrow{\text{d}} N(0,1).$$

Thus, as $p \to \infty$, the probability that $\lambda_{d+1} + \frac{1}{4}\|\tilde{\delta}\|^2 < \lambda_d$ converges to that of

$$\Phi\Big(\frac{4(\lambda_d - \lambda_{d+1}) - (p-d)(1-\epsilon)\tau^2}{\sqrt{(p-d)(2(1-\epsilon)(2\tau^2\sigma^2 + \sigma^4) + \epsilon(1-\epsilon)(\tau^4 + 2\tau^2\sigma^2 + \sigma^4))}} \Big).$$

This probability can be made arbitrarily close to 1 provided that $\lambda_d - \lambda_{d+1} \geq Cp(1-\epsilon)\tau^2$ for all sufficiently large p and for some constant $C > 1/4$. Since the probability that $\delta_1 = \delta_2 = \cdots = \delta_d$ is at least ϵ^d, we thus conclude that for sufficiently large p, with probability at least ϵ^d,

$$C(F_0^{(B)}, F_1^{(B)}) = 0 < C(F_0^{(A)}, F_1^{(A)}).$$

In the case where $\epsilon = \epsilon(p) \to 1$ as $p \to \infty$ such that $p(1-\epsilon) \to \theta$ for some constant K, then the probability that $\lambda_{d+1} + \frac{1}{4}\|\tilde{\delta}\|^2 < \lambda_d$ converges to the probability that

$$\frac{1}{4}\sum_{i=1}^{K}\sigma^2\chi_1^2(\tau) \geq \lambda_d - \lambda_{d+1}$$

where K is Poisson distributed with mean parameter θ and $\chi_i^2(\tau)$ is the non-central chi-square distribution with one degree of freedom and non-centrality parameter τ. Thus if $\lambda_d - \lambda_{d+1} \geq C\theta\tau^2 \log p$ for sufficiently large p and for some constant C, then this probability can also be made arbitrarily close to 1.

IV.E Finite Sample Performance

We now consider the finite sample performance of LoL and PCA-based classifiers in the high-dimensional setting with small or moderate sample sizes, e.g., when p is comparable to n or when $p \gg n$. Once again we assume that $X|Y = i \sim \mathcal{N}(\mu_i, \Sigma)$ for $i = 0, 1$. Furthermore, we also assume that Σ belongs to the class $\Theta(p, r, k, \tau, \lambda)$ as defined below.

Definition Let $\lambda > 0$, $\tau \geq 1$ and $k \leq p$ be given. Denote by $\Theta(p, r, k, \tau, \lambda, \sigma^2)$ the collection of matrices Σ such that

$$\Sigma = V\Lambda V^\top + \sigma^2 I$$

where V is a $p \times r$ matrix with orthonormal columns and Λ is a $r \times r$ diagonal matrix whose diagonal entries $\lambda_1, \lambda_2, \ldots, \lambda_r$ satisfy $\lambda \geq \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_r \geq \lambda/\tau$. In addition, assume also that $|\mathrm{supp}(V)| \leq k$ where $\mathrm{supp}(V)$ denote the non-zero rows of V, i.e., $\mathrm{supp}(V)$ is the subset of $\{1, 2, \ldots, p\}$ such that $V_j \neq 0$ if and only if $j \in \mathrm{supp}(V)$.

We note that in general $r \leq k \ll p$ and $\lambda/\tau \gg \sigma^2$. We then have the following result.

Theorem [45] Suppose there exists constants M_0 and M_1 such that $M_1 \log p \geq \log n \geq M_0 \log \lambda$. Then there exists a constant $c_0 = c_0(M_0, M_1)$ depending on M_0 and M_1 such that for all n and p for which

$$\frac{\tau k}{n} \log \frac{ep}{k} \leq c_0,$$

there exists an estimate $\hat{V}$ of V such that

$$\sup_{\Sigma \in \Theta(p, r, k, \tau, \lambda, \sigma^2)} \mathbb{E}\|\hat{V}\hat{V}^\top - VV^\top\|^2 \leq \frac{Ck(\sigma\lambda + \sigma^2)}{n\lambda^2} \log \frac{ep}{k}$$

where C is a universal constant not depending on p, r, k, τ, λ and σ^2.

We can therefore show that provided that M_0 and M_1 is large enough, and n and p satisfies the condition in the preceding theorem, then provided that the Chernoff information of the population version of LoL is larger than the Chernoff information of the population version of PCA, the expected Chernoff information for the sample version of LoL is also larger than the expected Chernoff information of the sample version of PCA. We emphasize that it is necessary that the LoL and the PCA version both projected into the top $d \leq r$ dimension of the sample covariance matrices.

E The R implementation of LOL

Figure 8 shows the R implementation of $\mathrm{L\scriptstyle OL}$ for binary classification using FlashMatrix [16]. The implementation takes a $D \times I$ matrix, where each column is a training instance and each instance has D features, and outputs a $D \times k$ projection matrix.

```
\Lol~<- function (m, labels , k) {
        counts <- fm.table(labels)
        num.labels <- length(counts$val)
        num.features <- dim(m)[1]
        nv <- k - (num.labels - 1)
        gr.sum <- fm.groupby(m, 1, fm.as.factor(labels, 2), fm.bo.add)
        gr.mean <- fm.mapply.row(gr.sum, counts$Freq, fm.bo.div, FALSE)
        diff <- fm.get.cols(gr.mean, 1) - fm.get.cols(gr.mean, 2)
        svd <- fm.svd(m, nv=0, nu=nv)
        fm.cbind(diff, svd$u)
}
```

Figure 8: The R implementation of LOL.

F Bibliography

[1] J. T. Vogelstein, Y. Park, T. Ohyama, R. Kerr, J. Truman, C. E. Priebe, and M. Zlatic, "Discovery of brainwide neural-behavioral maps via multiscale unsupervised structure learning," *Science*, vol. 344, no. 6182, pp. 386–392, 2014. 1

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-w 1

[3] R. A. Fisher, "Theory of Statistical Estimation," *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 22, no. 05, pp. 700–725, Oct. 1925. [Online]. Available: http://journals.cambridge.org/abstract_S0305004100009580 1

[4] I. T. Jolliffe, "Principal component analysis and factor analysis," in *Principal Component Analysis*, ser. Springer Series in Statistics. Springer, New York, NY, 1986, pp. 115–128. 2

[5] T. Hastie, R. Tibshirani, and M. Wainwright, *Statistical Learning with Sparsity: The Lasso and Generalizations (Chapman & Hall/CRC Monographs on Statistics & Applied Probability)*, 1st ed. Chapman and Hall/CRC, 7 May 2015. 2

[6] W. Su, M. Bogdan, and E. Candes, "False discoveries occur early on the lasso path," 5 Nov. 2015. 2

[7] Y. LeCun, C. Cortes, and C. Burges, "MNIST handwritten digit database." [Online]. Available: http://yann.lecun.com/exdb/mnist/ 3

[8] Y. Bengio, J.-F. Paiement, P. Vincent, O. Delalleau, N. L. Roux, and M. Ouimet, "Out-of-Sample extensions for LLE, isomap, MDS, eigenmaps, and spectral clustering," in *Advances in Neural Information Processing Systems 16*, S. Thrun, L. K. Saul, and P. B. Schölkopf, Eds. MIT Press, 2004, pp. 177–184. 3

[9] P. J. Bickel and E. Levina, "Some theory for Fisher's linear discriminant function, 'naive Bayes', and some alternatives when there are many more variables than observations," *Bernoulli*, vol. 10, no. 6, pp. 989–1010, Dec. 2004. [Online]. Available: http://projecteuclid.org/euclid.bj/1106314847 4

[10] T. Hastie and R. Tibshirani, "Discriminant analysis by gaussian mixtures," *J. R. Stat. Soc. Series B Stat. Methodol.*, vol. 58, no. 1, pp. 155–176, 1996. 4

[11] J. Fan, Y. Feng, and X. Tong, "A road to classification in high dimensional space: the regularized optimal affine discriminant," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 74, no. 4, pp. 745–771, Sep. 2012. [Online]. Available: http://doi.wiley.com/10.1111/j.1467-9868.2012.01029.x 4, 6

[12] H. Zou, "The Adaptive Lasso and Its Oracle Properties," pp. 1418–1429, 2006. 6

[13] T. Hastie, R. Tibshirani, and J. H. Friedman, "The Elements of Statistical Learning: Data Mining, Inference, and Prediction," *BeiJing: Publishing House of Electronics Industry*, 2004. 8

[14] D. Zheng, D. Mhembere, R. Burns, J. Vogelstein, C. E. Priebe, and A. S. Szalay, "FlashGraph: Processing billion-node graphs on an array of commodity SSDs," in *13th USENIX Conference on File and Storage Technologies (FAST 15)*, Santa Clara, CA, 2015. 8

[15] D. Zheng, R. Burns, J. Vogelstein, C. E. Priebe, and A. S. Szalay, "An ssd-based eigensolver for spectral analysis on billion-node graphs," *CoRR*, vol. abs/1602.01421, 2016.

[16] D. Zheng, D. Mhembere, J. T. Vogelstein, C. E. Priebe, and R. Burns, "Flashmatrix: Parallel, scalable data analysis with generalized matrix operations using commodity ssds," *arXiv preprint arXiv:1604.06414*, 2016. 8, 26

[17] E. J. Candès and T. Tao, "Near-Optimal Signal Recovery From Random Projections: Universal Encoding Strategies?" *IEEE Transactions on Information Theory*, vol. 52, no. 12, pp. 5406–5425, dec 2006. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4016283 8

[18] T. Hastie, K. W. Church, P. Li, and K. C. Kdd, "Very sparse random projections," *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '06*, p. 287, 2006. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1150402.1150436 8

[19] M. Lopes, L. Jacob, and M. J. Wainwright, "A More Powerful Two-Sample Test in High Dimensions using Random Projection," in *Neural Information Processing Systems*, 2011, pp. 1206–1214. [Online]. Available: http://papers.nips.cc/paper/4260-a-more-powerful-two-sample-test-in-high-dimensions-using-random-projection 10

[20] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. fisherfaces: Recognition using class specific linear projection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 711–720, 1997. 11

[21] C. Eckart and G. Young, "The approximation of one matrix by another of lower rank," *Psychometrika*, vol. 1, no. 3, pp. 211–218, Sep. 1936. [Online]. Available: http://www.springerlink.com/content/9v4274h33h75lq24/ 11

[22] V. de Silva and J. B. Tenenbaum, "Global Versus Local Methods in Nonlinear Dimensionality Reduction," in *Neural Information Processing Systems*, 2003, pp. 721–728.

[23] W. K. Allard, G. Chen, and M. Maggioni, "Multi-scale geometric methods for data sets II: Geometric Multi-Resolution Analysis," *Applied and Computational Harmonic Analysis*, vol. 32, no. 3, pp. 435–462, May 2012. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1063520311000868 11

[24] K.-C. Li, "Sliced Inverse Regression for Dimension Reduction," *Journal of the American Statistical Association*, vol. 86, no. 414, pp. 316–327, Jun. 1991. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/01621459.1991.10475035 11

[25] N. Tishby, F. C. Pereira, and W. Bialek, "The information bottleneck method arXiv : physics / 0004057v1 [physics . data-an] 24 Apr 2000," *Neural Computation*, pp. 1–16, 1999.

[26] A. Globerson and N. Tishby, "Sufficient Dimensionality Reduction," *Journal of Machine Learning Research*, vol. 3, no. 7-8, pp. 1307–1331, Oct. 2003. [Online]. Available: http://www.crossref.org/jmlr_DOI.html

[27] R. D. Cook and L. Ni, "Sufficient Dimension Reduction via Inverse Regression," *Journal of the American Statistical Association*, vol. 100, no. 470, pp. 410–428, Jun. 2005. [Online]. Available: http://amstat.tandfonline.com/doi/abs/10.1198/016214504000001501#.U6tH3Y1dUts

[28] K. Fukumizu, F. R. Bach, and M. I. Jordan, "Dimensionality Reduction for Supervised Learning with Reproducing Kernel Hilbert Spaces," *Journal of Machine Learning Research*, vol. 5, pp. 73–99, 2004. 11

[29] R. D. Cook, L. Forzani, and A. J. Rothman, "Prediction in abundant high-dimensional linear regression," *Electronic Journal of Statistics*, vol. 7, pp. 3059–3088, 2013. [Online]. Available: https://projecteuclid.org/euclid.ejs/1387207935 11

[30] P. J. Huber, "Projection Pursuit," *The Annals of Statistics*, vol. 13, no. 2, pp. 435–475, Jun. 1985. [Online]. Available: http://projecteuclid.org/euclid.aos/1176349519 11

[31] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold Regularization: A Geometric Framework for Learning from Labeled and Unlabeled Examples," *The Journal of Machine Learning Research*, vol. 7, pp. 2399–2434, Dec. 2006. [Online]. Available: http://dl.acm.org/citation.cfm?id=1248547.1248632http://dl.acm.org/citation.cfm?id=1248632 11

[32] J. Mairal, J. Ponce, G. Sapiro, A. Zisserman, and F. R. Bach, "Supervised Dictionary Learning," in *Advances in Neural Information Processing Systems*, 2009, pp. 1033–1040. [Online]. Available: http://papers.nips.cc/paper/3448-supervised 11

[33] D. L. Donoho and J. Jin, "Higher criticism thresholding: Optimal feature selection when useful features are rare and weak." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 105, no. 39, pp. 14 790–5, Sep. 2008. [Online]. Available: http://www.pnas.org/content/105/39/14790http://www.pnas.org/content/105/39/14790.short 11

[34] E. Bair, T. Hastie, D. Paul, and R. Tibshirani, "Prediction by supervised principal components," *J. Am. Stat. Assoc.*, vol. 101, no. 473, pp. 119–137, 1 Mar. 2006. 11

[35] A. Gretton, R. Herbrich, A. Smola, O. Bousquet, and B. Scholkopf, "Kernel methods for measuring independence," *Journal of Machine Learning Research*, vol. 6, pp. 2075–2129, 2005. 11

[36] E. Barshan, A. Ghodsi, Z. Azimifar, and M. Zolghadri Jahromi, "Supervised principal component analysis: Visualization, classification and regression on subspaces and submanifolds," *Pattern Recognit.*, vol. 44, no. 7, pp. 1357–1371, 1 Jul. 2011. 11

[37] A. Agarwal, O. Chapelle, M. Dudík, and J. Langford, "A reliable effective terascale linear learning system," *J. Mach. Learn. Res.*, vol. 15, pp. 1111–1133, 2014. 11

[38] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K. Mullers, "Fisher discriminant analysis with kernels," in *Neural Networks for Signal Processing IX: Proceedings of the 1999 IEEE Signal Processing Society Workshop (Cat. No.98TH8468)*. IEEE, 1999, pp. 41–48. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=788121 11

[39] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001. 11

[40] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. D. Croz, A. Greenbaum, S. Hammerling, A. McKenney, and D. Sorensen, *LAPACK Users' Guide: Third Edition*. SIAM, 1999. [Online]. Available: https://books.google.com/books?hl=en&lr=&id=AZlvEnr9gCgC&pgis=1 17

[41] L. Breiman, "Statistical modeling: The two cultures," *Statistical Science*, vol. 16, no. 3, pp. 199–231, 2001. 18

[42] H. Chernoff, "A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations," *Annals of Mathematical Statistics*, vol. 23, pp. 493–507, 1952. 19

[43] I. Csizár, "Information-type measures of difference of probability distributions and indirect observations," *Studia Scientiarum Mathematicarum Hungarica*, vol. 2, pp. 229–318, 1967. 20

[44] C. C. Leang and D. H. Johnson, "On the asymptotics of M-hypothesis bayesian detection," *IEEE Transactions on Information Theory*, vol. 43, pp. 280–282, 1997. 20

[45] T. T. Cai, Z. Ma, and Y. Wu, "Optimal estimation and rank detection for sparse spiked covariance matrices," 2013, arxiv preprint at http://arxiv.org/abs/1305.3235. 25