Deterministic Feature Selection for K-means Clustering

Christos Boutsidis
Mathematical Sciences Department
IBM T.J. Watson Research Center
cboutsi@us.ibm.com

Malik Magdon-Ismail
Computer Science Department
Rensselaer Polytechnic Institute
magdon@cs.rpi.edu

June 25, 2013

Abstract

We study feature selection for k-means clustering. Although the literature contains many methods with good empirical performance, algorithms with provable theoretical behavior have only recently been developed. Unfortunately, these algorithms are randomized and fail with, say, a constant probability. We present the first *deterministic* feature selection algorithm for k-means clustering with relative error guarantees. At the heart of our algorithm lies a deterministic method for decompositions of the identity and a structural result which quantifies some of the tradeoffs in dimensionality reduction.

1 Introduction

This paper is about feature selection for k-means clustering, a topic that received considerable attention from scientists and engineers. Arguably, k-means is the most widely used clustering algorithm in practice [40]. Its simplicity and effectiveness are remarkable among all the available methods [34]. On the negative side, using k-means to cluster high dimensional data with, for example, billions of features is not simple and straightforward [19]; the curse of dimensionality makes the algorithm very slow. On top of that, noisy features often lead to overfitting, another undesirable effect. Therefore, reducing the dimensionality of the data by selecting a subset of the features, i.e. feature selection, and optimizing the k-means objective on the low dimensional representation of the high dimensional data is an attractive approach that not only will make k-means faster, but also more robust [16, 19].

The natural concern with throwing away potentially useful dimensions is that it could lead to a significantly higher clustering error. So, one has to select the features carefully to ensure that one can recover comparably good clusters just by using the dimension-reduced data. Practitioners have developed numerous feature selection methods that work well *empirically* [16, 19]. The main focus of this work is on algorithms for feature selection with provable guarantees. Recently, Boutsidis et al. described a feature selection algorithm that gives a theoretical guarantee on the quality of the clusters that are produced after reducing the dimension [6]. Their algorithm, which employs a technique of Rudelson and Vershynin [36], selects the features randomly with probabilities that are computed via the right singular vectors of the matrix containing the data ([8] describes a similar randomized algorithm with the same bound but faster running time). Although Boutsidis et al. give a strong theoretical bound for the quality of the resulting clusters (we will discuss this bound in detail later), the bound fails with some non negligible probability, due to the randomness in how they sample the features. This means that every time the feature selection is performed, the algorithm could (a) fail, and (b) return a different answer each time. To better address the applicability of such feature selection algorithms for k-means, there is a need for *deterministic*, provably accurate feature selection algorithms. We present the first deterministic algorithms of this type. Our contributions can roughly be summarized as follows.

- **Deterministic Supervised Feature Selection (Theorem 2).** Given *any* dataset and *any* input k-partition of this dataset, there is a small set of $O(k)$ feature dimensions in which any near optimal output k-partition of the data is no more than a constant factor worse in quality than the given input k-partition (the quality of the input and output k-partitions are compared in the original dimension). Moreover, this small set of feature dimensions can be computed in deterministic low-order polynomial time. This is the first deterministic algorithm of this type. Prior work [6, 8] gives a randomized algorithm that can only reduce to $\Omega(k \log k)$ dimensions and guarantee comparable clustering quality.

- **Existence of a small set of near optimal features (Corollary 3).** We prove existence of a small set of near optimal features. That is, given *any* dataset and the number of clusters k, there is a set of $O(k)$ feature dimensions such that any optimal k-partition in the reduced dimension is no more than a constant factor worse in quality than the optimal k-partition of the dataset. The existence of such a small set of features was not known before. Prior work [6, 8] only implies the existence of $\Omega(k \log k)$ features with comparable performance.

- **Deterministic Unsupervised Feature Selection (Theorem 4).** Given *any* dataset and the number of clusters k, it is possible, in deterministic low-order polynomial time, to select r feature dimensions, for any $r > k$, such that the optimal k-partition of the dimension-reduced data is no more than $O(n/r)$ worse in quality than the optimal k-partition; here n is the number of features of the dataset. This is the first deterministic algorithm of this type. Prior work [6, 8] offers a randomized algorithm with error $(3 + O(k \log k/r))$, for $r = \Omega(k \log k)$.

- **Unsupervised Feature Selection with a small subset of features (Theorem 5).** Finally, given *any* dataset and the number of clusters k, in randomized low-order polynomial time it is possible to select a small number, r, of feature dimensions, with $k < r = o(k \log k)$, such that the optimal k-partition for the dimension-reduced data is no more than $O(k \log(k)/r)$ worse in quality than the optimal k-partition. In particular, this is the first (albeit randomized) algorithm of this type that can select a small subset of $O(k)$ feature dimensions and provide an $O(\log k)$-factor guarantee. Prior work [6, 8] is limited to selecting $r = \Omega(k \log k)$ features. The new algorithm combines ideas from this paper with the technique of [6, 8].

In order to prove our results we prove a structural result in Lemmas 10 and 11. This structural result quantifies the tradeoffs when selecting feature dimensions in terms of how the feature selection process preserves the matrix norms of certain crucial matrices. This is a general structural result and may be of independent interest.

In order to get deterministic algorithms and be able to select $O(k)$ feature dimensions, we need to use techniques that are completely different from those used in [6, 8]. Our approach is inspired by a recent deterministic result for decompositions of the identity which was introduced in [3] and subsequently extended in [4], while [6, 8] use the randomized technique of [36] to extract the features. This general approach might also be of interest to the Machine Learning and Pattern Recognition communities, with potential applications to other problems involving subsampling, for example sparse PCA and matrix approximation [37, 10].

1.1 Background

We first provide the basic background on k-means clustering that is needed to describe our results in Section 2. We postpone the more technical background needed for describing our main algorithms and for proving our main results to Section 3. We begin with the definition of the k-means clustering problem.[1] Consider a set $\mathcal{P}$ of m points in an n-dimensional Euclidian space,

$$\mathcal{P} = \{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_m\} \in \mathbb{R}^{m \times n},$$

and an integer k denoting the desired number of clusters. The objective of k-means is to find a k-partition of $\mathcal{P}$ such that points that are "close" to each other belong to the same cluster and points that are "far" from each other belong to different clusters. A k-partition of $\mathcal{P}$ is a collection

$$\mathcal{S} = \{\mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_k\},$$

of k non-empty pairwise disjoint sets which covers $\mathcal{P}$. Let $s_j = |\mathcal{S}_j|$, be the size of $\mathcal{S}_j$. For each set S_j, let $\boldsymbol{\mu}_j \in \mathbb{R}^n$ be its centroid (the mean point),

$$\boldsymbol{\mu}_j = \frac{1}{s_j} \sum_{\mathbf{p}_i \in S_j} \mathbf{p}_i.$$

The k-means objective function is

$$\mathcal{F}(\mathcal{P}, \mathcal{S}) = \sum_{i=1}^{m} \|\mathbf{p}_i - \boldsymbol{\mu}(\mathbf{p}_i)\|_2^2,$$

where $\boldsymbol{\mu}(\mathbf{p}_i)$ is the centroid of the cluster to which $\mathbf{p}_i$ belongs. The goal of k-means is to find a partition $\mathcal{S}$ which minimizes $\mathcal{F}$ for a given $\mathcal{P}$ and k. We will refer to any such optimal clustering as,

$$\mathcal{S}_{opt} = \underset{\mathcal{S}}{\operatorname{argmin}}\, \mathcal{F}(\mathcal{P}, \mathcal{S}).$$

[1] In Section 3, we provide an alternative definition using matrix notation, which will be useful in proving the main results of this work.

The corresponding objective value is

$$\mathcal{F}_{opt} = \mathcal{F}(\mathcal{P}, \mathcal{S}_{opt}).$$

The goal of feature selection is to construct points

$$\hat{\mathcal{P}} = \{\hat{\mathbf{p}}_1, \hat{\mathbf{p}}_2, \ldots, \hat{\mathbf{p}}_m\} \in \mathbb{R}^{m \times r},$$

(for some $r < n$ specified in advance) by projecting each $\mathbf{p}_i$ onto r of the coordinate dimensions. Consider the optimum k-means partition of the points in $\hat{\mathcal{P}}$,

$$\hat{\mathcal{S}}_{opt} = \operatorname*{argmin}_{\mathcal{S}} \mathcal{F}(\hat{\mathcal{P}}, \mathcal{S}).$$

The goal of feature selection is to construct a new set of points $\hat{\mathcal{P}}$ such that,

$$\mathcal{F}(\mathcal{P}, \hat{\mathcal{S}}_{opt}) \leq \alpha \cdot \mathcal{F}(\mathcal{P}, \mathcal{S}_{opt}).$$

Here, α is the approximation factor and might depend on m, n, k and r. In words, computing a partition $\hat{\mathcal{S}}_{opt}$ by using the low-dimensional data and plugging it back into the clustering metric $\mathcal{F}(\mathcal{P}, \cdot)$ for the high dimensional data, gives an α-approximation to the optimal value of the k-means objective function. Notice that we measure the quality of $\hat{\mathcal{S}}_{opt}$ by evaluating the k-means objective function in the original space, an approach which is standard [33, 27, 2]. Comparing $\hat{\mathcal{S}}_{opt}$ directly to $\mathcal{S}_{opt}$, i.e. the identity of the clusters, not just the clustering error, would be much more interesting but at the same time a much harder (combinatorial) problem. A feature selection algorithm is called *unsupervised* if it computes $\hat{\mathcal{P}}$ by only looking at $\mathcal{P}$ and k. *Supervised* algorithms construct $\hat{\mathcal{P}}$ with respect to a given partition $\mathcal{S}_{in}$ of the data. Finally, an algorithm will be a γ-approximation for k-means ($\gamma \geq 1$) if it finds a clustering $\mathcal{S}_\gamma$ with corresponding value $\mathcal{F}_\gamma \leq \gamma \mathcal{F}_{opt}$. Such algorithms will be used to state our results in a more general way.

Definition 1. [K-MEANS APPROXIMATION ALGORITHM] *An algorithm is a "γ-approximation" for k-means clustering ($\gamma \geq 1$) if it takes as input the dataset $\mathcal{P} = \{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_m\} \in \mathbb{R}^{m \times n}$ and the number of clusters k, and returns a clustering $\mathcal{S}_\gamma$ such that,*

$$\mathcal{F}_\gamma = \mathcal{F}(\mathcal{P}, \mathcal{S}_\gamma) \leq \gamma \mathcal{F}_{opt}.$$

The simplest algorithm with $\gamma = 1$, but exponential running time, would try all possible k-partitions and return the best. Another example of such an algorithm is in [27] with $\gamma = 1 + \epsilon$ ($0 < \epsilon < 1$). The corresponding running time is $O(mn \cdot 2^{(k/\epsilon)^{O(1)}})$. Also, the work in [2] describes a method with $\gamma = O(\log k)$ and running time $O(mnk)$. The latter two algorithms are randomized. For other γ-approximation algorithms, see [33] as well as the discussion and the references in [33, 27, 2].

2 Statement of our main results

We first discuss guarantees for supervised feature selection when the user has a candidate input k-partition. We next present results for the unsupervised case. We end this section with a brief discussion of the results.

2.1 Supervised Feature Selection

Our first result is within the context of supervised feature selection. Suppose that we are given points $\mathcal{P}$ and some k-partition $\mathcal{S}_{in}$. The goal is to find the features of the points in $\mathcal{P}$ from which we can find a partition $\mathcal{S}_{out}$ that is not much worse than the given partition $\mathcal{S}_{in}$. Notice that if $\mathcal{S}_{in}$ is arbitrarily bad, then $\mathcal{S}_{out}$ might be much better than $\mathcal{S}_{in}$; however, our bound does not capture how much better the resulting partition would be. It only guarantees that it will not be much worse than the given partition.

Theorem 2. *There is an $O(mn\min\{m,n\} + rk^2n)$ time deterministic feature selection algorithm which takes as input any set of points $\mathcal{P} = \{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_m\} \in \mathbb{R}^{m\times n}$, a number of clusters k, a number of features to be selected $k < r < n$, and a k-partition of the points $\mathcal{S}_{in}$. The algorithm constructs $\hat{\mathcal{P}} = \{\hat{\mathbf{p}}_1, \hat{\mathbf{p}}_2, ..., \hat{\mathbf{p}}_m\} \in \mathbb{R}^{m\times r}$ such that $\mathcal{S}_{out}$, an arbitrary γ-approximation on $\hat{\mathcal{P}}$, satisfies*

$$\mathcal{F}(\mathcal{P}, \mathcal{S}_{out}) \quad \leq \quad \left(1 + \frac{4\gamma}{(1-\sqrt{k/r})^2}\right)\mathcal{F}(\mathcal{P}, \mathcal{S}_{in}).$$

Asymptotically, as $r \to \infty$, the approximation factor is $\gamma \cdot O\left(1+\sqrt{k/r}\right)$. Essentially the clustering $\mathcal{S}_{out}$ is at most a constant factor worse than the original clustering $\mathcal{S}_{in}$; but, $\mathcal{S}_{out}$ was computed using the lower dimensional data. This means we can compress the number of the feature dimensions without destroying a given specific clustering in the data. This is useful, for example, in privacy preserving applications where one seeks to release minimal information of the data without destroying much of the encoded information [38]. Notice that the feature selection part of the theorem (i.e. the construction of $\hat{\mathcal{P}}$) is deterministic. The γ-approximation algorithm, which can be randomized, is only used to describe the clustering that can be obtained with the features returned by our deterministic feature selection algorithm (same comment applies to Theorem 4). The corresponding algorithm is presented as Algorithm 4 along with the proof of the theorem in Section 5. Prior to this result, the best, and in fact the only method with theoretical guarantees for this supervised setting [6, 8] is randomized[2] and gives

$$\mathcal{F}(\mathcal{P}, \mathcal{S}_{out}) \leq \gamma \cdot (3 + O(\sqrt{k\log(k)/r})) \cdot \mathcal{F}(\mathcal{P}, \mathcal{S}_{in}).$$

Further, [6, 8] requires $r = \Omega(k\log k)$, otherwise the analysis breaks. We improve this bound by a factor of $O(\log k)$; also, we allow the user to select as few as $r = O(k)$ features.

[2]We should note that [6] describes the result for the unsupervised setting but it's easy to verify that the same algorithm and bound apply to the supervised setting as well.

A surprising existential result is a direct corollary of the above theorem by setting $\mathcal{S}_{in} = \mathcal{S}_{opt}$ and using an optimal clustering algorithm on the reduced dimension data ($\gamma = 1$).

Corollary 3. *For any set of points $\mathcal{P} = \{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_m\} \in \mathbb{R}^{m \times n}$, integer k, and $\epsilon > 0$, there is a set of $r = O(k/\epsilon^2)$ features $\hat{\mathcal{P}} = \{\hat{\mathbf{p}}_1, \hat{\mathbf{p}}_2, ..., \hat{\mathbf{p}}_m\} \in \mathbb{R}^{m \times r}$ such that, if*

$$\mathcal{S}_{opt} = \underset{\mathcal{S}}{\operatorname{argmin}}\, \mathcal{F}(\mathcal{P}, \mathcal{S}) \qquad \text{and} \qquad \hat{\mathcal{S}}_{opt} = \underset{\mathcal{S}}{\operatorname{argmin}}\, \mathcal{F}(\hat{\mathcal{P}}, \mathcal{S}),$$

then,

$$\mathcal{F}(\mathcal{P}, \hat{\mathcal{S}}_{opt}) \leq (5 + \epsilon)\, \mathcal{F}(\mathcal{P}, \mathcal{S}_{opt}).$$

In words, for any dataset there exist a small subset of $O(k/\epsilon^2)$ features such that the optimal clustering on these features is at most a $(5+\epsilon)$-factor worse than the optimal clustering on the original features. Unfortunately, finding these features without the knowledge of $\mathcal{S}_{opt}$ is not obvious.

2.2 Unsupervised Feature Selection

Our second result is within the context of unsupervised feature selection. In unsupervised feature selection, the goal is to obtain a k-partition that is as close as possible to the optimal k-partition of the high dimensional data. The theorem below shows that it is possible to reduce the dimension of any high-dimensional dataset by using a deterministic method and obtain some theoretical guarantees on the quality of the clusters.

Theorem 4. *There is an $O(mn \min\{m, n\} + rk^2 n)$ time deterministic algorithm which takes as input any set of points $\mathcal{P} = \{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_m\} \in \mathbb{R}^{m \times n}$, a number of clusters k, and a number of features to be selected $k < r < n$. The algorithm constructs $\hat{\mathcal{P}} = \{\hat{\mathbf{p}}_1, \hat{\mathbf{p}}_2, ..., \hat{\mathbf{p}}_m\} \in \mathbb{R}^{m \times r}$ such that $\mathcal{S}_{out}$, an arbitrary γ-approximation on $\hat{\mathcal{P}}$, satisfies*

$$\mathcal{F}(\mathcal{P}, \mathcal{S}_{out}) \quad \leq \quad \left(1 + 4\gamma \frac{(1 + \sqrt{n/r})^2}{(1 - \sqrt{k/r})^2}\right) \mathcal{F}_{opt}.$$

Asymptotically, as $r \to \infty$ and $n/r \to \infty$, the approximation factor is $\gamma \cdot O\left(n/r\right)$. The corresponding algorithm is presented as Algorithm 5 along with the proof of the theorem in Section 5. Prior to this result, the best method for this task was given in [6, 8], which replaces $O\left(n/r\right)$ with $\left(3 + O\left(\sqrt{k \log(k)/r}\right)\right)$ and it is *randomized*, i.e. this bound is achieved only with a constant probability. Further, [6, 8] requires $r = \Omega(k \log k)$, so one *cannot* select $o\left(k \log k\right)$ features. Clearly, our bound is worse but we achieve it deterministically, and it applies to any $r > k$, even $r = k + 1$.

Finally, it is possible to combine the algorithm of Theorem 4 with the randomized algorithm of [6, 8] and obtain the following result.

Theorem 5. *There is an $O\left(mnk + rk^3 \log(k) + r \log r\right)$ time randomized algorithm which takes as input any set of points $\mathcal{P} = \{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_m\} \in \mathbb{R}^{m \times n}$, a number of clusters k, and a number of features to be selected $k < r < 4k \log k$. The algorithm constructs $\hat{\mathcal{P}} = \{\hat{\mathbf{p}}_1, \hat{\mathbf{p}}_2, ..., \hat{\mathbf{p}}_m\} \in \mathbb{R}^{m \times r}$ such* *that $\mathcal{S}_{out}$, an arbitrary γ-approximation on $\hat{\mathcal{P}}$, satisfies with probability* 0.4,

$$F(\mathcal{P}, \mathcal{S}_{out}) \quad \leq \quad \left(15 + 320\gamma \left(\frac{1 + \sqrt{16k\log(20k)/r}}{1 - \sqrt{k/r}}\right)^2\right) \mathcal{F}_{opt}.$$

Since $r < 4k\log k$, the approximation factor is $\gamma \cdot O\left(k\log k/r\right)$; there is no result in the literature which provides an approximation guarantee for selecting $o(k\log k)$ features. Note that the theorem is stated with the assumption $r < 4k\log k$, even though the corresponding algorithm (Algorithm 6) runs with any $r > k$. This is because the algorithm of Theorem 5 is essentially the main algorithm of [6, 8] when $r = \Omega(k\log k)$ (that is, we do not provide an improved result for $r = \Omega(k\log k)$). What we achieve is to break the barrier of having to select $r = \Omega\left(k\log k\right)$ features. So, Theorem 5, to the best of our knowledge, is the best available algorithm in the literature providing theoretical guarantees for unsupervised feature selection in k-means clustering using $O\left(k\right)$ features. The proof of Theorem 5 is given in Section 5. Comparing Theorem 5 with Theorem 4, we obtain a much better approximation bound at the cost of introducing randomization. We note here that the main algorithm of [6] can be obtained as Algorithm 6 in Section 5 after ignoring the 4th and 5th steps and replacing the approximate SVD with the exact SVD in the first step (the algorithm in [8] uses approximate SVD as we do in Algorithm 6).

2.3 Discussion

Supervised feature selection for k-means is not prevalent in the literature, precisely because most of the time one is interested in obtaining the input partition $\mathcal{S}_{in}$ to begin with. The practical implication of our result is that it is possible to identify (efficiently) a small set of important feature dimensions that are sufficient for essentially reproducing a given clustering. We also point out that there are randomized algorithms which can quickly give a good input clustering $\mathbf{X}_{in}$ from which to obtain the features; for example the k-means++ algorithm [2] provides a randomized input clustering that is an $O(\log k)$-factor approximation. Using this clustering will yield randomized feature dimensions such that clustering in the reduced dimension would also give an $O(\log k)$ factor approximation.

For unsupervised feature selection, the theoretical guarantee from the deterministic algorithm is weak, but we suspect that it may perform very well in practice on typical input data matrices. The empirical investigation of this algorithm would also be an interesting future direction.

All our algorithms require the top k singular vectors of the data matrix (or an approximation to the top k singular vectors). Then, the selection of the features is done by looking at the structure of these singular vectors and using the deterministic techniques of [3, 4]. We should note that [6, 8] takes a similar approach as far as computing the (approximate) singular vectors of the dataset; then [6, 8] employ the randomized technique of [36] to extract the features.

The high level description of our results for unsupervised feature selection assumed that the output partition, obtained in the reduced-dimension space, is the optimal one: $\mathcal{S}_{out} = \hat{\mathcal{S}}_{opt}$. This is the case if we assume a γ-approximation algorithm with $\gamma = 1$. Notice, though, that our theorems are actually more general and can accomodate any $\gamma \geq 1$.

3 Preliminaries

We now provide the necessary technical background that we will use in presenting our algorithms and proving our main theorems in Section 5.

3.1 Singular Value Decomposition

The Singular Value Decomposition (SVD) of a matrix plays an important role in the description of our algorithms. The SVD of a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ with $\text{rank}(\mathbf{A}) = \rho \le \min\{m, n\}$ is

$$\mathbf{A} = \underbrace{\begin{pmatrix} \mathbf{U}_k & \mathbf{U}_{\rho-k} \end{pmatrix}}_{\mathbf{U_A}\in\mathbb{R}^{m\times\rho}} \underbrace{\begin{pmatrix} \mathbf{\Sigma}_k & \mathbf{0} \\ \mathbf{0} & \mathbf{\Sigma}_{\rho-k} \end{pmatrix}}_{\mathbf{\Sigma_A}\in\mathbb{R}^{\rho\times\rho}} \underbrace{\begin{pmatrix} \mathbf{V}_k^{\mathrm{T}} \\ \mathbf{V}_{\rho-k}^{\mathrm{T}} \end{pmatrix}}_{\mathbf{V}_{\mathbf{A}}^{\mathrm{T}}\in\mathbb{R}^{\rho\times n}},$$

with singular values $\sigma_1 \ge \sigma_2 \ge \cdots \ge \sigma_\rho > 0$ contained in $\mathbf{\Sigma}_k \in \mathbb{R}^{k\times k}$ and $\mathbf{\Sigma}_{\rho-k} \in \mathbb{R}^{(\rho-k)\times(\rho-k)}$. $\mathbf{U}_k \in \mathbb{R}^{m\times k}$ and $\mathbf{U}_{\rho-k} \in \mathbb{R}^{m\times(\rho-k)}$ contain the left singular vectors of $\mathbf{A}$. Similarly, $\mathbf{V}_k \in R^{n\times k}$ and $\mathbf{V}_{\rho-k} \in \mathbb{R}^{n\times(\rho-k)}$ contain the right singular vectors. In our description of the SVD, we have explicitly separated the singular vectors into the top k singular vectors and the remaining $\rho - k$ sincular vectors (where k is the input number of clusters); this is because there is a relationship in our algorithms between the input number of clusters k and the top-k singular vectors. We use $\mathbf{A}^+ = \mathbf{V_A}\mathbf{\Sigma}_{\mathbf{A}}^{-1}\mathbf{U}_{\mathbf{A}}^{\mathrm{T}} \in \mathbb{R}^{n\times m}$ to denote the Moore-Penrose pseudo-inverse of $\mathbf{A}$ with $\mathbf{\Sigma}_{\mathbf{A}}^{-1}$ denoting the inverse of $\mathbf{\Sigma_A}$. We use the Frobenius and the spectral matrix norms: $\|\mathbf{A}\|_{\mathrm{F}}^2 = \sum_{i,j} \mathbf{A}_{ij}^2 = \sum_{i=1}^{\rho} \sigma_i^2$; and $\|\mathbf{A}\|_2^2 = \sigma_1^2$, respectively. Given $\mathbf{A}$ and $\mathbf{B}$ of appropriate dimensions, $\|\mathbf{AB}\|_{\mathrm{F}} \le \|\mathbf{A}\|_{\mathrm{F}}\|\mathbf{B}\|_2$; we call this property spectral submultiplicativity because it is a stronger version of the standard matrix-norm submultiplicativity property $\|\mathbf{AB}\|_{\mathrm{F}} \le \|\mathbf{A}\|_{\mathrm{F}}\|\mathbf{B}\|_{\mathrm{F}}$. Let $\mathbf{A}_k = \mathbf{U}_k\mathbf{\Sigma}_k\mathbf{V}_k^{\mathrm{T}} = \mathbf{A}\mathbf{V}_k\mathbf{V}_k^{\mathrm{T}}$ and $\mathbf{A}_{\rho-k} = \mathbf{A} - \mathbf{A}_k = \mathbf{U}_{\rho-k}\mathbf{\Sigma}_{\rho-k}\mathbf{V}_{\rho-k}^{\mathrm{T}}$. The SVD gives the best rank-k approximation to $\mathbf{A}$ in both the spectral and Frobenius norms: if $\text{rank}(\tilde{\mathbf{A}}) \le k$ then (for $\xi = 2, \mathrm{F}$) $\|\mathbf{A} - \mathbf{A}_k\|_\xi \le \|\mathbf{A} - \tilde{\mathbf{A}}\|_\xi$; and, $\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}} = \|\mathbf{\Sigma}_{\rho-k}\|_{\mathrm{F}}$.

3.2 Approximate Singular Value Decomposition

The exact SVD of $\mathbf{A}$, though a deterministic algorithm, takes cubic time. More specifically, for any $k \ge 1$, the running time to compute the top k left and/or right singular vectors of $\mathbf{A} \in \mathbb{R}^{m\times n}$ is $O(mn\min\{m, n\})$ (See Section 8.6 in [18]). We will use the exact SVD in our deterministic feature selection algorithms in Theorems 2 and 4. To speed up our randomized algorithm in Theorem 5, we will use a factorization, which can be computed fast and approximates the SVD in some well defined sense. We quote a recent result from [4] for a relative-error Frobenius norm SVD approximation algorithm (which is proved in [5]). The exact description of the corresponding algorithm implied by Lemma 6 is out of the scope of this work.

Lemma 6 (Lemma 11 in [5]). *Given $\mathbf{A} \in \mathbb{R}^{m\times n}$ of rank ρ, a target rank $2 \le k < \rho$, and $0 < \epsilon < 1$, there exists an $O(mnk/\epsilon)$ time randomized algorithm that computes a matrix $\mathbf{Z} \in \mathbb{R}^{n\times k}$ and a matrix*

$\mathbf{E} = \mathbf{A} - \mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}} \in \mathbb{R}^{m \times n}$ *such that* $\mathbf{Z}^{\mathrm{T}}\mathbf{Z} = \mathbf{I}_k$, $\mathbf{E}\mathbf{Z} = \mathbf{0}_{m \times k}$, *and*

$$\mathbb{E}\left[\|\mathbf{E}\|_{\mathrm{F}}^2\right] \leq (1+\epsilon)\,\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2.$$

We use $\mathbf{Z} = \mathit{FastApproximateSVD}(\mathbf{A}, k, \epsilon)$ *to denote this randomized procedure.*

The matrix $\mathbf{Z}$ is an approximate version of $\mathbf{V}_k$, hence the notion fast SVD.

3.3 Linear Algebraic Definition of k-means

We now give the linear algebraic definition of the k-means problem. Recall that $\mathcal{P} = \{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_m\} \in \mathbb{R}^{m \times n}$ contains the data points, k is the number of clusters, and $\mathcal{S}$ denotes a k-partition of $\mathcal{P}$. Define the data matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ as

$$\mathbf{A} = \begin{bmatrix} - \mathbf{p}_1^{\mathrm{T}} - \\ - \mathbf{p}_2^{\mathrm{T}} - \\ \vdots \\ - \mathbf{p}_m^{\mathrm{T}} - \end{bmatrix}.$$

We represent a clustering $\mathcal{S}$ by its cluster indicator matrix $\mathbf{X} \in \mathbb{R}^{m \times k}$. Each column $j = 1, \ldots, k$ of $\mathbf{X}$ represents a cluster. Each row $i = 1, \ldots, m$ indicates the cluster membership of the point $\mathbf{p}_i$. So,

$$\mathbf{X}_{ij} = 1/\sqrt{s_j},$$

if and only if the data point $\mathbf{p}_i$ is in cluster S_j ($s_j = |\mathcal{S}_j|$). Every row of $\mathbf{X}$ has one non-zero element, corresponding to the cluster to which the data point belongs to. There are s_j non-zero elements in column j, which indicates the points belonging to S_j. Obesrve that $\mathbf{X}$ is a matrix with orthonormal columns. We now obtain the matrix formulation of the k-means problem:

$$\begin{aligned} \mathcal{F}(\mathbf{A}, \mathbf{X}) &:= \|\mathbf{A} - \mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \\ &= \sum_{i=1}^{m} \|\mathbf{p}_i^{\mathrm{T}} - \mathbf{X}_i\mathbf{X}^{\mathrm{T}}\mathbf{A}\|_2^2 \\ &= \sum_{i=1}^{m} \|\mathbf{p}_i^{\mathrm{T}} - \boldsymbol{\mu}(\mathbf{p}_i)^{\mathrm{T}}\|_2^2 \\ &= \mathcal{F}(\mathcal{P}, \mathcal{S}), \end{aligned}$$

where we define $\mathbf{X}_i$ as the ith row of $\mathbf{X}$ and we have used the identity $\mathbf{X}_i\mathbf{X}^{\mathrm{T}}\mathbf{A} = \boldsymbol{\mu}(p_i)^{\mathrm{T}}$, for $i = 1, ..., m,$. This identity is true because $\mathbf{X}^{\mathrm{T}}\mathbf{A}$ is a matrix whose row j is $\sqrt{s_j}\boldsymbol{\mu}_j$, proportional to the centroid of the jth cluster; now, $\mathbf{X}_i$ picks the row corresponding to its non-zero element, i.e. the cluster corresponding to point i, and scales it by $1/\sqrt{s_j}$. Using this formulation, the goal of k-means is to find an indicator matrix $\mathbf{X}$ which minimizes $\|\mathbf{A} - \mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2$. From now on, $\mathbf{X} \in \mathbb{R}^{m \times k}$ will refer generically to such an indicator matrix. We will denote the best such indicator matrix $\mathbf{X}_{opt}$:

$$\mathbf{X}_{opt} = \underset{\mathbf{X} \in \mathbb{R}^{m \times k}}{\operatorname{argmin}} \|\mathbf{A} - \mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2;$$

so,

$$\mathcal{F}_{opt} = \|\mathbf{A} - \mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2.$$

Since $\mathbf{A}_k$ is the best rank k approximation to $\mathbf{A}$, $\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2 \leq \mathcal{F}_{opt}$, because $\mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A}$ has rank at most k.

Using the matrix formulation for k-means, we can restate the goal of a feature selection algorithm in matrix notation. So, the goal of feature selection is to construct points $\mathbf{C} \in \mathbb{R}^{m\times r}$, where $\mathbf{C}$ is a subset of r columns from $\mathbf{A}$, which represent the m points in the r-dimensional selected feature space. Note also that we will allow rescaling of the corresponding columns, i.e. multiplication of each column with a scalar. Now consider the optimum k-means partition of the points in $\mathbf{C}$:

$$\hat{\mathbf{X}}_{opt} = \underset{\mathbf{X}\in\mathbb{R}^{m\times k}}{\operatorname{argmin}} \|\mathbf{C} - \mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{C}\|_{\mathrm{F}}^2.$$

(recall that $\mathbf{X} \in \mathbb{R}^{m\times k}$ is restricted to be an indicator matrix.) The goal of feature selection is to construct the new set of points $\mathbf{C}$ such that,

$$\|\mathbf{A} - \hat{\mathbf{X}}_{opt}\hat{\mathbf{X}}_{opt}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq \alpha\|\mathbf{A} - \mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2.$$

More generally, we may use a γ-approximate clustering algorithm to construct an approximation to $\hat{\mathbf{X}}_{opt}$.

3.4 Spectral Sparsification and Rudelson's concentration Lemma

We now present the main tools we use to select the features in the context of k-means clustering. Let $\mathbf{\Omega} \in \mathbb{R}^{n\times r}$ be a matrix such that $\mathbf{A\Omega} \in \mathbb{R}^{m\times r}$ contains r columns of $\mathbf{A}$. The matrix $\mathbf{\Omega}$ is a "sampling" projection operator onto the r-dimensional subset of features. Let $\mathbf{S} \in \mathbb{R}^{r\times r}$ be a diagonal matrix, so $\mathbf{A\Omega S} \in \mathbb{R}^{m\times r}$ rescales the columns of $\mathbf{A}$ that are in $\mathbf{A\Omega}$. Intuitively, $\mathbf{A\Omega S}$ projects down to the chosen r dimensions and then rescales the data points along these dimensions.

In this section we present two deterministic and a randomized algorithm for constructing such matrices $\mathbf{\Omega}$ and $\mathbf{S}$. All three algorithms come from prior work but we include a full description of them for completeness. The corresponding lemmas in this subsection describe certain spectral properties of these matrices. The proofs of these Lemmas appeared in prior work, so we only give appropriate references and do not prove the results here.

Lemma 7 (Lemma 13 in [5]). *Let* $\mathbf{V}^{\mathrm{T}} \in \mathbb{R}^{k\times n}$ *and* $\mathbf{B} \in \mathbb{R}^{\ell_1\times n}$ *with* $\mathbf{V}^{\mathrm{T}}\mathbf{V} = \mathbf{I}_k$. *Let* $r > k$. *Algorithm 1 runs in* $O(rk^2n + \ell_1 n)$ *time and deterministically constructs a sampling matrix* $\mathbf{\Omega} \in \mathbb{R}^{n\times r}$ *and a rescaling matrix* $\mathbf{S} \in \mathbb{R}^{r\times r}$ *such that,*

$$\sigma_k(\mathbf{V}^{\mathrm{T}}\mathbf{\Omega S}) \geq 1 - \sqrt{k/r}; \qquad\qquad \|\mathbf{B\Omega S}\|_{\mathrm{F}} \leq \|\mathbf{B}\|_{\mathrm{F}}.$$

Algorithm 1 is a greedy technique that selects columns one at a time. To describe the algorithm in more detail, it is convenient to view the input matrices as two sets of n vectors, $\mathbf{V}^{\mathrm{T}} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n]$ and $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n]$. Given k and $r > k$, introduce the iterator $\tau = 0, 1, 2, ..., r-1$, and define

```
Input: $\mathbf{V}^{\mathrm{T}} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n] \in \mathbb{R}^{k\times n}$, $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n] \in \mathbb{R}^{\ell_1\times n}$, and $r > k$.
Output: Sampling matrix $\mathbf{\Omega} \in \mathbb{R}^{n\times r}$ and rescaling matrix $\mathbf{S} \in \mathbb{R}^{r\times r}$.

1: Initialize $\mathbf{A}_0 = \mathbf{0}_{k\times k}$, $\mathbf{\Omega} = \mathbf{0}_{n\times r}$, and $\mathbf{S} = \mathbf{0}_{r\times r}$.
2: Set constants $\delta_{\mathbf{B}} = \|\mathbf{B}\|_{\mathrm{F}}^2(1-\sqrt{k/r})^{-1};\ \delta_L = 1$.
3: for $\tau = 0$ to $r-1$ do
4:     Let $\mathrm{L}_\tau = \tau - \sqrt{rk}$.
5:     Pick index $i_\tau \in \{1,2,...,n\}$ and number $t_\tau > 0$ (see text for the definition of $U, L$):
           $U(\mathbf{b}_{i_\tau}, \delta_{\mathbf{B}}) \le \frac{1}{t_\tau} \le L(\mathbf{v}_{i_\tau}, \delta_L, \mathbf{A}_\tau, \mathrm{L}_\tau).$
6:     Update $\mathbf{A}_{\tau+1} = \mathbf{A}_\tau + t_\tau \mathbf{v}_{i_\tau}\mathbf{v}_{i_\tau}^{\mathrm{T}}$; set $\mathbf{\Omega}_{i_\tau,\tau+1} = 1$ and $\mathbf{S}_{\tau+1,\tau+1} = 1/\sqrt{t_\tau}$.
7: end for
8: Multiply all the weights in $\mathbf{S}$ by
           $\sqrt{r^{-1}(1-\sqrt{k/r})}.$
9: Return: $\mathbf{\Omega}$ and $\mathbf{S}$.
```

Algorithm 1: DeterministicSamplingI (Lemma 7)

the parameter $\mathrm{L}_\tau = \tau - \sqrt{rk}$. For a square symmetric matrix $\mathbf{A} \in \mathbb{R}^{k\times k}$ with eigenvalues $\lambda_1, \ldots, \lambda_k$, $\mathbf{v} \in \mathbb{R}^k$ and $\mathrm{L} \in \mathbb{R}$, define

$$\phi(\mathrm{L}, \mathbf{A}) = \sum_{i=1}^{k} \frac{1}{\lambda_i - \mathrm{L}},$$

and let $L(\mathbf{v}, \delta_L, \mathbf{A}, \mathrm{L})$ be defined as

$$L(\mathbf{v}, \delta_L, \mathbf{A}, \mathrm{L}) = \frac{\mathbf{v}^{\mathrm{T}}(\mathbf{A} - \mathrm{L}'\mathbf{I}_k)^{-2}\mathbf{v}}{\phi(\mathrm{L}', \mathbf{A}) - \phi(\mathrm{L}, \mathbf{A})} - \mathbf{v}^{\mathrm{T}}(\mathbf{A} - \mathrm{L}'\mathbf{I}_k)^{-1}\mathbf{v},$$

where

$$\mathrm{L}' = \mathrm{L} + \delta_L = \mathrm{L} + 1.$$

For a vector $\mathbf{z}$ and scalar $\delta > 0$, define the function

$$U(\mathbf{z}, \delta) = \frac{1}{\delta}\mathbf{z}^{\mathrm{T}}\mathbf{z}.$$

At each iteration τ, the algorithm selects i_τ, $t_\tau > 0$ for which

$$U(\mathbf{b}_{i_\tau}, \delta_{\mathbf{B}}) \le t_\tau^{-1} \le L(\mathbf{v}_{i_\tau}, \delta_L, \mathbf{A}_\tau, \mathrm{L}_\tau).$$

The running time of the algorithm is dominated by the search for an index i_τ satisfying

$$U(\mathbf{b}_{i_\tau}, \delta_{\mathbf{B}}) \le t_\tau^{-1} \le L(\mathbf{v}_{i_\tau}, \delta^{-1}, \mathbf{A}_\tau, \mathrm{L}_\tau)$$

```
Input: $\mathbf{V}^{\mathrm{T}} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n] \in \mathbb{R}^{k\times n}$, $\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_d] \in \mathbb{R}^{\ell_2\times n}$, and $r > k$.
Output: Sampling matrix $\mathbf{\Omega} \in \mathbb{R}^{n\times r}$ and rescaling matrix $\mathbf{S} \in \mathbb{R}^{r\times r}$.

1: Initialize $\mathbf{A}_0 = \mathbf{0}_{k\times k}$, $\mathbf{B}_0 = \mathbf{0}_{\ell_2\times\ell_2}$, $\mathbf{\Omega} = \mathbf{0}_{n\times r}$, and $\mathbf{S} = \mathbf{0}_{r\times r}$.
2: Set constants $\delta_{\mathbf{Q}} = (1+\ell_2/r)\left(1-\sqrt{k/r}\right)^{-1}$; $\delta_L = 1$.
3: for $\tau = 0$ to $r-1$ do
4:     Let $\mathrm{L}_\tau = \tau - \sqrt{rk}$; $\mathrm{U}_\tau = \delta_{\mathbf{Q}}\left(\tau + \sqrt{\ell_2 r}\right)$
5:     Pick index $i_\tau \in \{1,2,...,n\}$ and number $t_\tau > 0$ (see text for the definition of $U, L$):
           $\hat{U}(\mathbf{q}_{i_\tau}, \delta_{\mathbf{Q}}, \mathbf{B}_\tau, \mathrm{U}_\tau) \le \frac{1}{t_\tau} \le L(\mathbf{v}_{i_\tau}, \delta_L, \mathbf{A}_\tau, \mathrm{L}_\tau).$
6:     Update $\mathbf{A}_{\tau+1} = \mathbf{A}_\tau + t_\tau \mathbf{v}_{i_\tau}\mathbf{v}_{i_\tau}^{\mathrm{T}}$; $\mathbf{B}_{\tau+1} = \mathbf{B}_\tau + t_\tau \mathbf{q}_{i_\tau}\mathbf{q}_{i_\tau}^{\mathrm{T}}$, and
       set $\mathbf{\Omega}_{i_\tau,\tau+1} = 1$, $\mathbf{S}_{\tau+1,\tau+1} = 1/\sqrt{t_\tau}$.
7: end for
8: Multiply all the weights in $\mathbf{S}$ by $\sqrt{r^{-1}\left(1-\sqrt{k/r}\right)}$.
9: Return: $\mathbf{\Omega}$ and $\mathbf{S}$.
```

Algorithm 2: DeterministicSamplingII (Lemma 8)

(one can achieve that by exhaustive search). One needs $\phi(\mathrm{L}, \mathbf{A})$, and hence the eigenvalues of $\mathbf{A}$. This takes $O(k^3)$ time, once per iteration, for a total of $O(rk^3)$. Then, for $i = 1, \ldots, n$, we need to compute L for every $\mathbf{v}_i$. This takes $O(nk^2)$ per iteration, for a total of $O(rnk^2)$. To compute U, we need $\mathbf{b}_i^{\mathrm{T}}\mathbf{b}_i$ for $i = 1, \ldots, n$, which need to be computed only once for the whole algorithm and takes $O(\ell_1 n)$. So, the total running time is $O(nrk^2 + \ell_1 n)$.

Lemma 8 (Lemma 12 in [5]). *Let $\mathbf{V}^{\mathrm{T}} \in \mathbb{R}^{k\times n}$, $\mathbf{Q} \in \mathbb{R}^{\ell_2\times n}$, $\mathbf{V}^{\mathrm{T}}\mathbf{V} = \mathbf{I}_k$, and $\mathbf{Q}^{\mathrm{T}}\mathbf{Q} = \mathbf{I}_{\ell_2}$. Let $r > k$. Algorithm 2 runs in $O(rk^2 n + r\ell_2^2 n)$ time and deterministically constructs a sampling matrix $\mathbf{\Omega} \in \mathbb{R}^{n\times r}$ and a rescaling matrix $\mathbf{S} \in \mathbb{R}^{r\times r}$ such that,*

$$\sigma_k(\mathbf{V}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S}) \ge 1 - \sqrt{k/r}; \qquad \|\mathbf{Q}\mathbf{\Omega}\mathbf{S}\|_2 \le 1 + \sqrt{\ell_2/r}.$$

Moreover, if $\mathbf{Q} = \mathbf{I}_n$, the running time is $O(rk^2 n)$.

Algorithm 2 is similar to Algorithm 1; we only need to define the function $\hat{U}$. For a square symmetric matrix $\mathbf{B} \in \mathbb{R}^{\ell_2\times\ell_2}$ with eigenvalues $\lambda_1, \ldots, \lambda_{\ell_2}$, $\mathbf{q} \in \mathbb{R}^{\ell_2}$, $\mathrm{U} \in \mathbb{R}$, define:

$$\hat{\phi}(\mathrm{U}, \mathbf{B}) = \sum_{i=1}^{\ell_2} \frac{1}{\mathrm{U} - \lambda_i},$$

and let $\hat{U}(\mathbf{q}, \delta_{\mathbf{Q}}, \mathbf{B}, \mathrm{U})$ be defined as

$$\hat{U}(\mathbf{q}, \delta_{\mathbf{Q}}, \mathbf{B}, \mathrm{U}) = \frac{\mathbf{q}^{\mathrm{T}}(\mathbf{B} - \mathrm{U}'\mathbf{I}_{\ell_2})^{-2}\mathbf{q}}{\hat{\phi}(\mathrm{U}, \mathbf{B}) - \hat{\phi}(\mathrm{U}', \mathbf{B})} - \mathbf{q}^{\mathrm{T}}(\mathbf{B} - \mathrm{U}'\mathbf{I}_{\ell_2})^{-1}\mathbf{q},$$

Input: $\mathbf{V}^{\mathrm{T}} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n] \in \mathbb{R}^{k \times n}$, and the number of sampled columns $r > 4k \log k$.
Output: Sampling matrix $\mathbf{\Omega} \in \mathbb{R}^{n \times r}$ and rescaling matrix $\mathbf{S} \in \mathbb{R}^{r \times r}$.

```
1: For $i = 1, ..., n$ compute $p_i = \frac{1}{k}\|\mathbf{v}_i\|_2^2$.
2: Initialize $\mathbf{\Omega} = \mathbf{0}_{n \times r}$ and $\mathbf{S} = \mathbf{0}_{r \times r}$.
3: for $\tau = 1$ to $r$ do
4:     Select index $i \in \{1, 2, ..., n\}$ independently with the probability of selecting index $i$
       equal to $p_i$.
5:     Set $\mathbf{\Omega}_{i,\tau} = 1$ and $\mathbf{S}_{\tau,\tau} = 1/\sqrt{p_i r}$.
6: end for
7: Return: $\mathbf{\Omega}$ and $\mathbf{S}$.
```

Algorithm 3: RandomizedSampling (Lemma 9)

where

$$\mathrm{U}' = \mathrm{U} + \delta_{\mathbf{Q}} = \mathrm{U} + (1 + \ell_2/r)\left(1 - \sqrt{k/r}\right)^{-1}.$$

The running time of the algorithm is $O(nrk^2 + nr\ell_2^2)$.

Lemmas 7 and 8 are generalizations of the original work of Batson et al. [3], which presented a deterministic algorithm that operates only on $\mathbf{V}$. Lemmas 7 and 8 are proved in [5] (see Lemmas 12 and 13 in [5]). We will use Lemma 7 in a novel way: we will apply it to a matrix $\mathbf{B}$ of the form

$$\mathbf{B} = \begin{pmatrix} \mathbf{B}_1 \\ \mathbf{B}_2 \end{pmatrix},$$

so we will be able to control the sum of the Frobenius norms of two different matrices $\mathbf{B}_1$, $\mathbf{B}_2$, which is all we need in our application. In the above formula, $\mathbf{B}$ is a 2×1 block matrix with $\mathbf{B}_1$ and $\mathbf{B}_2$ being the underlying blocks. The above two lemmas will be used to prove our deterministic results for feature selection, i.e. Theorems 2 and 4.

We will also need the following result, which corresponds to the celebrated work of Rudelson and Vershynin [35, 36] and describes a randomized algorithm for constructing matrices $\mathbf{\Omega}$ and $\mathbf{S}$. The lower bound with the optimal constants 4 and 20 was obtained more recently as Lemma 15 in [30]. The Frobenius norm bounds are straightforward; a short proof can be found as Eqn. 36 in [11]. This lemma will be used to prove our hybrid randomized result for feature selection, i.e. Theorem 5.

Lemma 9. *Let* $\mathbf{V}^{\mathrm{T}} \in \mathbb{R}^{k \times n}$*,* $\mathbf{B} \in \mathbb{R}^{\ell_1 \times n}$ *and* $\mathbf{Q} \in \mathbb{R}^{\ell_2 \times n}$*, with* $\mathbf{V}^{\mathrm{T}}\mathbf{V} = \mathbf{I}_k$*. Let* $r > 4k \log k$*. Algorithm 3, in* $O(nk + r \log r)$ *time, constructs a sampling matrix* $\mathbf{\Omega} \in \mathbb{R}^{n \times r}$ *and a rescaling matrix* $\mathbf{S} \in \mathbb{R}^{r \times r}$ *such that*

$$\mathbb{E}\left[\|\mathbf{Q\Omega S}\|_{\mathrm{F}}^2\right] = \|\mathbf{Q}\|_{\mathrm{F}}^2; \qquad \mathbb{E}\left[\|\mathbf{B\Omega S}\|_{\mathrm{F}}^2\right] = \|\mathbf{B}\|_{\mathrm{F}}^2;$$

and with probability at least 0.9*,*

$$\sigma_k^2(\mathbf{V}^{\mathrm{T}}\mathbf{\Omega S}) \geq 1 - \sqrt{4k\log(20k)/r}.$$

4 A structural bound for clustering error with feature selection

In this section, we develop two lemmas which are general structural bounds for the clustering error when using feature selection. These lemmas are the basis for all of our algorithms, and hence are crucial to proving our results, specifically Theorems 2, 4, and 5 (recall that there is no algorithm for Corollary 3 since that result is non-constructive).

In the following two lemmas, the sampling and rescaling matrices $\mathbf{\Omega} \in \mathbb{R}^{n \times r}$ and $\mathbf{S} \in \mathbb{R}^{r \times r}$ are *arbitrary*, modulo the rank restriction in the lemmas. Thus, one can apply the lemmas to a general dimension reduction matrix $\mathbf{\Psi} = \mathbf{\Omega}\mathbf{S}$.

Lemma 10. *Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ be the input data matrix, $k > 0$ an integer, and $\mathbf{X}_{in} \in \mathbb{R}^{m \times k}$ an input k-clustering indicator matrix. Let $\mathbf{\Omega} \in \mathbb{R}^{n \times r}$ and $\mathbf{S} \in \mathbb{R}^{r \times r}$ be any matrices, and set $\mathbf{C} = \mathbf{A}\mathbf{\Omega}\mathbf{S} \in \mathbb{R}^{m \times r}$. Let $\mathbf{Z} \in \mathbb{R}^{n \times k}$ be any orthonormal matrix, and define the residual $\mathbf{E}$ by $\mathbf{A} = \mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}} + \mathbf{E}$, where the matrix $\mathbf{E} \in \mathbb{R}^{m \times n}$. Let $\mathbf{X}_{out}$ be the output of some γ-approximation algorithm on $(\mathbf{C}, k)$. If* $\mathrm{rank}(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S}) = k$, *then*

$$\|\mathbf{A} - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \le \|\mathbf{E}\|_{\mathrm{F}}^2 + 2\gamma \frac{\|(\mathbf{I}_n - \mathbf{X}_{in}\mathbf{X}_{in}^{\mathrm{T}})\mathbf{A}\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2 + \|\mathbf{E}\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2}{\sigma_k^2(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S})}.$$

This lemma bounds the clustering error of the output partition $\mathbf{X}_{out}$ that is constructed in the reduced dimension space. We get to pick $\mathbf{Z}\mathbf{Z}^{\mathrm{T}}$, a projection matrix; if we project $\mathbf{A}$ on the right using $\mathbf{Z}\mathbf{Z}^{\mathrm{T}}$, there is some residual $\mathbf{E}$. If this residual $\mathbf{E}$ is small (directly controlled by $\mathbf{Z}$) *and* if the sampling and rescaling matrices $\mathbf{\Omega}$, $\mathbf{S}$ are chosen so that: (i) the input partition has a reasonably small clustering error in the reduced dimension space; (ii) the size of the residual $\mathbf{E}$ does not increase significantly under sampling and rescaling; and (iii) the sampling and rescaling does not significantly alter the singular structure of the projector $\mathbf{Z}$, then the output clustering error is small. These are the basic three guidelines we need to follow in selecting $\mathbf{\Omega}$ and $\mathbf{S}$ in our algorithms.

Proof. We start by manipulating the term $\|\mathbf{A} - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2$. First, observe that $\mathbf{A} = \mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}} + \mathbf{E}$. Next, from the Pythagorean Theorem for matrices[3] and using that $\mathbf{I}_m - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}}$ is a projection matrix we obtain,

$$\begin{aligned}
\|\mathbf{A} - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 &= \|(\mathbf{I}_m - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}})\mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}} + (\mathbf{I}_m - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}})\mathbf{E}\|_{\mathrm{F}}^2 \\
&= \|(\mathbf{I}_m - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}})\mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}}\|_{\mathrm{F}}^2 + \|(\mathbf{I}_m - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}})\mathbf{E}\|_{\mathrm{F}}^2 \\
&\le \|(\mathbf{I}_m - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}})\mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}}\|_{\mathrm{F}}^2 + \|\mathbf{E}\|_{\mathrm{F}}^2.
\end{aligned}$$

We now bound the first term in the last expression. Given $\mathbf{\Omega}$ and $\mathbf{S}$, for some residual matrix $\mathbf{Y} \in \mathbb{R}^{m \times n}$, let

$$\mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}} = \mathbf{A}\mathbf{\Omega}\mathbf{S}(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S})^{+}\mathbf{Z}^{\mathrm{T}} + \mathbf{Y}.$$

[3] Let $\mathbf{Y}_1, \mathbf{Y}_2 \in \mathbb{R}^{m \times n}$ satisfy $\mathbf{Y}_1\mathbf{Y}_2^{\mathrm{T}} = \mathbf{0}_{m \times m}$. Then, $\|\mathbf{Y}_1 + \mathbf{Y}_2\|_{\mathrm{F}}^2 = \|\mathbf{Y}_1\|_{\mathrm{F}}^2 + \|\mathbf{Y}_2\|_{\mathrm{F}}^2$.

(See Section 3.1 for the definition of the pseudo-inverse operator.) Then,

$$\begin{aligned}\|(\mathbf{I}_m - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}})\mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}}\|_{\mathrm{F}}^2 &\overset{(a)}{\leq} 2\|(\mathbf{I}_m - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}})\mathbf{A}\mathbf{\Omega}\mathbf{S}(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S})^{+}\mathbf{Z}^{\mathrm{T}}\|_{\mathrm{F}}^2 + 2\|\mathbf{Y}\|_{\mathrm{F}}^2 \\ &\overset{(b)}{\leq} 2\|(\mathbf{I}_m - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}})\mathbf{A}\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2 \cdot \|(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S})^{+}\|_2^2 + 2\|\mathbf{Y}\|_{\mathrm{F}}^2 \\ &\overset{(c)}{\leq} 2\gamma\|(\mathbf{I}_m - \mathbf{X}_{in}\mathbf{X}_{in}^{\mathrm{T}})\mathbf{A}\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2 \cdot \|(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S})^{+}\|_2^2 + 2\|\mathbf{Y}\|_{\mathrm{F}}^2.\end{aligned}$$

In (a), we used $\|\mathbf{Y}_1 + \mathbf{Y}_2\|_{\mathrm{F}}^2 \leq 2\|\mathbf{Y}_1\|_{\mathrm{F}}^2 + 2\|\mathbf{Y}_2\|_{\mathrm{F}}^2$ (for any two matrices $\mathbf{Y}_1, \mathbf{Y}_2$), which follows from the triangle inequality of matrix norms; further we have removed the projection matrix $\mathbf{I}_m - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}}$ from the second term, which can be done without increasing the Frobenius norm. In (b), we used spectral submultiplicativity and the fact that $\mathbf{Z}^{\mathrm{T}}$ is orthonormal, and so it can be dropped without increasing the spectral norm. Finally, in (c), we replaced $\mathbf{X}_{out}$ by $\mathbf{X}_{in}$ and the factor γ appeared in the first term. To understand why this can be done, notice that, by assumption, $\mathbf{X}_{out}$ was constructed by running the γ-approximation on $\mathbf{C} = \mathbf{A}\mathbf{\Omega}\mathbf{S}$. So, for any indicator matrix $\mathbf{X}$:

$$\|(\mathbf{I}_m - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}})\mathbf{A}\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2 \leq \gamma\|(\mathbf{I}_m - \mathbf{X}\mathbf{X}^{\mathrm{T}})\mathbf{A}\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2,$$

and we can set $\mathbf{X} = \mathbf{X}_{in}$. Finally, we bound the term $\|\mathbf{Y}\|_{\mathrm{F}}^2$. Recall that

$$\begin{aligned}\mathbf{Y} &= \mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}} - \mathbf{A}\mathbf{\Omega}\mathbf{S}(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S})^{+}\mathbf{Z}^{\mathrm{T}} \\ &= \mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}} - \mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S}(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S})^{+}\mathbf{Z}^{\mathrm{T}} - (\mathbf{A} - \mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}})\mathbf{\Omega}\mathbf{S}(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S})^{+}\mathbf{Z}^{\mathrm{T}}.\end{aligned}$$

Note that

$$\mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}} - \mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S}(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S})^{+}\mathbf{Z}^{\mathrm{T}} = \mathbf{0}_{m\times n},$$

since $\mathrm{rank}(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S}) = k$, and so

$$\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S}(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S})^{+} = \mathbf{I}_k.$$

So,

$$\begin{aligned}\|\mathbf{Y}\|_{\mathrm{F}}^2 &= \|(\mathbf{A} - \mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}})\mathbf{\Omega}\mathbf{S}(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S})^{+}\mathbf{Z}^{\mathrm{T}}\|_{\mathrm{F}}^2 \\ &\leq \|(\mathbf{A} - \mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}})\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2\|(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S})^{+}\mathbf{Z}\|_2^2 \\ &\leq \|(\mathbf{A} - \mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}})\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2\|(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S})^{+}\|_2^2 \\ &= \frac{\|(\mathbf{A} - \mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}})\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2}{\sigma_k^2(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S})}.\end{aligned}$$

In the first 3 steps, we have used spectral submultiplicativity, and in the last step we have used the definition of the spectral norm of the pseudo-inverse. Combining all these bounds together (and using $\gamma \geq 1$):

$$\|\mathbf{A} - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq \|\mathbf{E}\|_{\mathrm{F}}^2 + 2\gamma\frac{\|(\mathbf{I}_n - \mathbf{X}_{in}\mathbf{X}_{in}^{\mathrm{T}})\mathbf{A}\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2 + \|(\mathbf{A} - \mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}})\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2}{\sigma_k^2(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S})}.$$

```
Input: $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{X}_{in} \in \mathbb{R}^{m \times k}$, number of clusters $k$, and number of features $r > k$.
Output: $\mathbf{C} \in \mathbb{R}^{m \times r}$ containing $r$ rescaled columns of $\mathbf{A}$.

1: Compute the matrix $\mathbf{V}_k \in \mathbb{R}^{n \times k}$ from the SVD of $\mathbf{A}$.
2: Let $\mathbf{B} = \begin{pmatrix} \mathbf{A} - \mathbf{A}\mathbf{V}_k\mathbf{V}_k^{\mathrm{T}} \\ \mathbf{A} - \mathbf{X}_{in}\mathbf{X}_{in}^{\mathrm{T}}\mathbf{A} \end{pmatrix} \in \mathbb{R}^{2m \times n}$; $\mathbf{A} - \mathbf{A}\mathbf{V}_k\mathbf{V}_k^{\mathrm{T}}$, $\mathbf{A} - \mathbf{X}_{in}\mathbf{X}_{in}^{\mathrm{T}}\mathbf{A} \in \mathbb{R}^{m \times n}$.
3: Let $[\mathbf{\Omega}, \mathbf{S}] = \mathsf{DeterministicSamplingI}(\mathbf{V}_k^{\mathrm{T}}, \mathbf{B}, r)$. (see Algorithm 1)
4: return $\mathbf{C} = \mathbf{A}\mathbf{\Omega}\mathbf{S} \in \mathbb{R}^{m \times r}$.
```

Algorithm 4: Supervised Feature Selection (Theorem 2)

■

Lemma 11 is a simple corollary of Lemma 10 by using $\mathbf{Z} = \mathbf{V}_k \in \mathbb{R}^{n \times k}$, i.e. the matrix containing the top k right singular vectors of $\mathbf{A}$. Notice also that in this case $\mathbf{E} = \mathbf{A} - \mathbf{A}\mathbf{V}_k\mathbf{V}_k^{\mathrm{T}} = \mathbf{A} - \mathbf{A}_k$. Observe that

$$\|\mathbf{E}\|_{\mathrm{F}}^2 = \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2 \leq \|\mathbf{A} - \mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq \|\mathbf{A} - \mathbf{X}_{in}\mathbf{X}_{in}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2,$$

The first inequality is from the optimality of $\mathbf{A}_k$ (see also Section 3.3); and, the second inequality is from the optimality of the indicator matrix $\mathbf{X}_{opt}$ for clustering the rows of $\mathbf{A}$ (the high dimensional points).

Lemma 11. *Let $\mathbf{A} \in \mathbb{R}^{m \times n}$, $k > 0$, and $\mathbf{V}_k$ its top-k right singular matrix. Let $\mathbf{X}_{in} \in \mathbb{R}^{m \times k}$ be an input clustering indicator matrix and $\mathbf{E} = \mathbf{A} - \mathbf{A}\mathbf{V}_k\mathbf{V}_k^{\mathrm{T}} = \mathbf{A} - \mathbf{A}_k$. Let $\mathbf{\Omega} \in \mathbb{R}^{n \times r}$ and $\mathbf{S} \in \mathbb{R}^{r \times r}$ be any matrices and set $\mathbf{C} = \mathbf{A}\mathbf{\Omega}\mathbf{S} \in \mathbb{R}^{m \times r}$. Let $\mathbf{X}_{out}$ be the output of some γ-approximation algorithm on $(\mathbf{C}, k)$. If $\mathrm{rank}(\mathbf{V}_k^{\mathrm{T}}\mathbf{\Omega}\mathbf{S}) = k$, then*

$$\|\mathbf{A} - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \quad \leq \quad \|\mathbf{A} - \mathbf{X}_{in}\mathbf{X}_{in}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 + 2\gamma \frac{\|(\mathbf{A} - \mathbf{X}_{in}\mathbf{X}_{in}^{\mathrm{T}}\mathbf{A})\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2 + \|\mathbf{E}\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2}{\sigma_k^2(\mathbf{V}_k^{\mathrm{T}}\mathbf{\Omega}\mathbf{S})}.$$

5 Proofs of Main Theorems

5.1 Proof of Theorem 2

We state the corresponding algorithm as Algorithm 4. Theorem 2 will follow by using Lemmas 11 and 7. We would like to apply Lemma 11 for the matrices $\mathbf{\Omega}$ and $\mathbf{S}$ constructed with *DeterministicSamplingI*, i.e. Algorithm 1. To do that, we need

$$\mathrm{rank}(\mathbf{V}_k^{\mathrm{T}}\mathbf{\Omega}\mathbf{S}) = k.$$

This rank requirement follows from Lemma 7, because

$$\sigma_k(\mathbf{V}_k^{\mathrm{T}}\mathbf{\Omega}\mathbf{S}) > 1 - \sqrt{k/r} > 0.$$

```
Input: $\mathbf{A} \in \mathbb{R}^{m \times n}$, number of clusters $k$, and number of features $r > k$.
Output: $\mathbf{C} \in \mathbb{R}^{m \times r}$ containing $r$ rescaled columns of $\mathbf{A}$.

1: Compute the matrix $\mathbf{V}_k \in \mathbb{R}^{n \times k}$ from the SVD of $\mathbf{A}$.
2: Let $[\mathbf{\Omega}, \mathbf{S}] = \mathsf{DeterministicSamplingII}(\mathbf{V}_k^{\mathrm{T}}, \mathbf{I}_n, r)$. (see Algorithm 2)
3: return $\mathbf{C} = \mathbf{A\Omega S} \in \mathbb{R}^{m \times r}$.
```

Algorithm 5: Unsupervised Feature Selection (Theorem 4)

Hence, Lemma 11 gives

$$\|\mathbf{A} - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \le \|\mathbf{A} - \mathbf{X}_{in}\mathbf{X}_{in}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 + 2\gamma \frac{\|(\mathbf{A} - \mathbf{X}_{in}\mathbf{X}_{in}^{\mathrm{T}}\mathbf{A})\mathbf{\Omega S}\|_{\mathrm{F}}^2 + \|\mathbf{E\Omega S}\|_{\mathrm{F}}^2}{\sigma_k^2(\mathbf{V}_k^{\mathrm{T}}\mathbf{\Omega S})}. \tag{1}$$

We can now use the bound for $\sigma_k^2(\mathbf{V}_k^{\mathrm{T}}\mathbf{\Omega S})$ from Lemma 7:

$$\sigma_k^2(\mathbf{V}_k^{\mathrm{T}}\mathbf{\Omega S}) > \left(1 - \sqrt{k/r}\right)^2. \tag{2}$$

Also, we can use the Frobenius norm bound from Lemma 7. Our choice of the matrix $\mathbf{B} \in \mathbb{R}^{2m \times n}$ is

$$\mathbf{B} = \begin{pmatrix} \mathbf{A} - \mathbf{A}\mathbf{V}_k\mathbf{V}_k^{\mathrm{T}} \\ \mathbf{A} - \mathbf{X}_{in}\mathbf{X}_{in}^{\mathrm{T}}\mathbf{A} \end{pmatrix} = \begin{pmatrix} \mathbf{E} \\ \mathbf{A} - \mathbf{X}_{in}\mathbf{X}_{in}^{\mathrm{T}}\mathbf{A} \end{pmatrix}.$$

This bound from Lemma 7 gives $\|\mathbf{B\Omega S}\|_{\mathrm{F}}^2 \le \|\mathbf{B}\|_{\mathrm{F}}^2$, where, from our choice of $\mathbf{B}$,

$$\begin{aligned} \|\mathbf{B\Omega S}\|_{\mathrm{F}}^2 &= \|\mathbf{E\Omega S}\|_{\mathrm{F}}^2 + \|(\mathbf{A} - \mathbf{X}_{in}\mathbf{X}_{in}^{\mathrm{T}}\mathbf{A})\mathbf{\Omega S}\|_{\mathrm{F}}^2; \\ \|\mathbf{B}\|_{\mathrm{F}}^2 &= \|\mathbf{E}\|_{\mathrm{F}}^2 + \|\mathbf{A} - \mathbf{X}_{in}\mathbf{X}_{in}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2. \end{aligned}$$

so, the result of applying Lemma 7 is

$$\|\mathbf{E\Omega S}\|_{\mathrm{F}}^2 + \|(\mathbf{A} - \mathbf{X}_{in}\mathbf{X}_{in}^{\mathrm{T}}\mathbf{A})\mathbf{\Omega S}\|_{\mathrm{F}}^2 \le \|\mathbf{E}\|_{\mathrm{F}}^2 + \|\mathbf{A} - \mathbf{X}_{in}\mathbf{X}_{in}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2. \tag{3}$$

Using (2) and (3) in (1), we have

$$\|\mathbf{A} - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \le \|\mathbf{A} - \mathbf{X}_{in}\mathbf{X}_{in}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 + \frac{2\gamma}{(1 - \sqrt{k/r})^2}\left(\|\mathbf{A} - \mathbf{X}_{in}\mathbf{X}_{in}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 + \|\mathbf{E}\|_{\mathrm{F}}^2\right).$$

Since $\|\mathbf{E}\|_{\mathrm{F}}^2 = \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2 \le \|\mathbf{A} - \mathbf{X}_{in}\mathbf{X}_{in}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2$, the result follows.

Lastly, we compute the running time of the algorithm. Algorithm 4 computes the matrix $\mathbf{V}_k$ in $O(mn\min\{m,n\})$ time; then, $\mathbf{A} - \mathbf{A}\mathbf{V}_k\mathbf{V}_k^{\mathrm{T}}$ and $\mathbf{A} - \mathbf{X}_{in}\mathbf{X}_{in}^{\mathrm{T}}\mathbf{A}$ can be computed in $O(mnk)$. *DeterministicSamplingI* takes time $O(rk^2n + mn)$, from Lemma 7. Overall, the running time of Algorithm 4 in Theorem 2 is $O(mn\min\{m,n\} + rk^2n)$.

5.2 Proof of Theorem 4

We state the corresponding algorithm as Algorithm 5. Theorem 4 follows by combining Lemmas 11 and 8. The rank requirement in Lemma 11 is satisfied by the bound for the smallest singular value of $\mathbf{V}_k^{\mathrm{T}}\mathbf{\Omega}\mathbf{S}$ in Lemma 8. We are going to apply Lemma 11, with

$$\mathbf{X}_{in} = \mathbf{X}_{opt}.$$

Even though we apply Lemma 11 with $\mathbf{X}_{in} = \mathbf{X}_{opt}$, this is merely an artifact of the proof that is needed to obtain the final result; we do not actually need to compute $\mathbf{X}_{opt}$, as is evident in the description of Algorithm 5, which runs without knowledge of $\mathbf{X}_{opt}$. We have, from Lemma 11,

$$\|\mathbf{A} - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq \|\mathbf{A} - \mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 + 2\gamma\frac{\|(\mathbf{A} - \mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A})\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2 + \|\mathbf{E}\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2}{\sigma_k^2(\mathbf{V}_k^{\mathrm{T}}\mathbf{\Omega}\mathbf{S})}. \tag{4}$$

To bound the second term on the right hand side, we use spectral submultiplicativity to obtain

$$\|(\mathbf{A} - \mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A})\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2 \leq \|(\mathbf{A} - \mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A})\|_{\mathrm{F}}^2 \cdot \|\mathbf{I}_n\mathbf{\Omega}\mathbf{S}\|_2^2,$$

and

$$\begin{aligned}\|\mathbf{E}\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2 &\leq \|\mathbf{E}\|_{\mathrm{F}}^2 \cdot \|\mathbf{I}_n\mathbf{\Omega}\mathbf{S}\|_2^2 \\ &\leq \|(\mathbf{A} - \mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A})\|_{\mathrm{F}}^2 \cdot \|\mathbf{I}_n\mathbf{\Omega}\mathbf{S}\|_2^2.\end{aligned}$$

where the last inequality follows because $\|\mathbf{E}\|_{\mathrm{F}}^2 = \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2 \leq \|\mathbf{A} - \mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2$. Plugging back into (4),

$$\|\mathbf{A} - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq \|\mathbf{A} - \mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 + 4\gamma\|\mathbf{A} - \mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2\frac{\|\mathbf{I}_n\mathbf{\Omega}\mathbf{S}\|_2^2}{\sigma_k^2(\mathbf{V}_k^{\mathrm{T}}\mathbf{\Omega}\mathbf{S})}.$$

The result now follows by using the bounds from Lemma 8,

$$\begin{aligned}\sigma_k(\mathbf{V}_k^{\mathrm{T}}\mathbf{\Omega}\mathbf{S}) &\geq 1 - \sqrt{k/r}; \\ \|\mathbf{I}_n\mathbf{\Omega}\mathbf{S}\|_2 &\leq 1 + \sqrt{n/r}.\end{aligned}$$

Finally, we comment on the running time of the algorithm. Algorithm 5 computes the matrix $\mathbf{V}_k$ in $O(mn\min\{m,n\})$ time. *DeterministicSamplingII* takes time $O(rk^2n)$, from Lemma 8. Overall, the running time of the algorithm is $O(mn\min\{m,n\} + rk^2n)$.

5.3 Proof of Theorem 5

We state the corresponding algorithm as Algorithm 6. To prove Theorem 5, we start with the general bound of Lemma 10; to apply the lemma, we need to satisfy the rank assumption, which will become clear shortly, during the course of the proof.

```
Input: $\mathbf{A} \in \mathbb{R}^{m \times k}$, number of clusters $k$, and number of features $k < r < 4k \log k$.
Output: $\mathbf{C} \in \mathbb{R}^{m \times r}$ containing $r$ rescaled columns of $\mathbf{A}$.

1: Compute the matrix $\mathbf{Z} \in \mathbb{R}^{n \times k}$ from the approximate SVD of $\mathbf{A}$ in Lemma 6:
   $\mathbf{Z} = FastApproximateSVD(\mathbf{A}, k, \frac{1}{2})$ .
2: Let $c = \max\{r, 16k \log(20k)\}$.
3: Let $[\mathbf{\Omega}_1, \mathbf{S}_1] = \mathsf{RandomizedSampling}(\mathbf{Z}^{\mathrm{T}}, c)$. (see Algorithm 3)
4: Compute $\tilde{\mathbf{V}} \in \mathbb{R}^{c \times k}$, the matrix of top $k$ right singular vectors of $\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}_1\mathbf{S}_1 \in \mathbb{R}^{k \times c}$.
5: Let $[\mathbf{\Omega}, \mathbf{S}] = \mathsf{DeterministicSamplingII}(\tilde{\mathbf{V}}^{\mathrm{T}}, \mathbf{I}_c, r)$. (see Algorithm 2)
6: return $\mathbf{C} = \mathbf{A}\mathbf{\Omega}_1\mathbf{S}_1\mathbf{\Omega}\mathbf{S} \in \mathbb{R}^{m \times r}$.
```

Algorithm 6: Randomized Unsupervised Feature Selection (Theorem 5)

The algorithm of Theorem 5 constructs the matrix $\mathbf{Z}$ by using the algorithm of Lemma 6 with ϵ set to a constant, $\epsilon = \frac{1}{2}$. Using the same notation as in Lemma 6,

$$\mathbf{E} = \mathbf{A} - \mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}},$$

and

$$\mathbb{E}\left[\|\mathbf{E}\|_{\mathrm{F}}^2\right] \leq \tfrac{3}{2}\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2.$$

We can now apply the Markov's inequality, to obtain that with probability at least 0.9,

$$\|\mathbf{E}\|_{\mathrm{F}}^2 \leq 15\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2. \tag{5}$$

The randomized construction in the third step of Algorithm 6 gives sampling and rescaling matrices $\mathbf{\Omega}_1$ and $\mathbf{S}_1$; the deterministic construction in the fifth step of Algorithm 6 gives sampling and rescaling matrices $\mathbf{\Omega}$ and $\mathbf{S}$. To apply Lemma 10, we will choose

$$\mathbf{X}_{in} = \mathbf{X}_{opt},$$

since the lemma gives us the luxury to pick any indicator matrix $\mathbf{X}_{in}$. Note that we do not need to actually compute $\mathbf{X}_{opt}$ in the algorithm; we are just using it in the proof to get the desired result, as in the proof of Theorem 4.

Algorithm 6 first selects c columns using $\mathbf{\Omega}_1\mathbf{S}_1 \in \mathbb{R}^{n \times c}$. Let $\mathbf{Y} = \mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}_1\mathbf{S}_1 \in \mathbb{R}^{k \times c}$, and consider its SVD,

$$\mathbf{Y} = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^{\mathrm{T}},$$

with $\tilde{\mathbf{U}} \in \mathbb{R}^{k \times k}$, $\tilde{\mathbf{\Sigma}} \in \mathbb{R}^{k \times k}$, and $\tilde{\mathbf{V}} \in \mathbb{R}^{c \times k}$. Lemma 9 now implies that with probability 0.9,

$$\sigma_k^2(\mathbf{Y}) \geq 1 - \sqrt{\frac{4k\log(20k)}{c}} = \frac{1}{2}, \tag{6}$$

because $c = 16k\log(20k)$. This means that $\mathrm{rank}(\tilde{\mathbf{V}}^{\mathrm{T}}) = k$. Since $\tilde{\mathbf{V}}^{\mathrm{T}}$ is the input to Deterministic

Sampling II, and because $r > k$, it follows from Lemma 8 that

$$\sigma_k(\tilde{\mathbf{V}}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S}) > 1 - \sqrt{k/r} > 0. \tag{7}$$

Hence,

$$\sigma_k(Z^{\mathrm{T}}\mathbf{\Omega}_1\mathbf{S}_1\mathbf{\Omega}\mathbf{S}) = \mathrm{rank}(\mathbf{Y}\mathbf{\Omega}\mathbf{S}) = \mathrm{rank}(\tilde{\mathbf{V}}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S}) = k,$$

and we can apply Lemma 10, with $\mathbf{E} = \mathbf{A} - \mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}}$:

$$\|\mathbf{A} - \mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \le \|\mathbf{E}\|_{\mathrm{F}}^2 + 2\gamma\frac{\|(\mathbf{A} - \mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A})\mathbf{\Omega}_1\mathbf{S}_1\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2 + \|\mathbf{E}\mathbf{\Omega}_1\mathbf{S}_1\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2}{\sigma_k^2(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}_1\mathbf{S}_1\mathbf{\Omega}\mathbf{S})}. \tag{8}$$

To bound the denominator of the second term, observe that

$$\sigma_k^2(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}_1\mathbf{S}_1\mathbf{\Omega}\mathbf{S}) = \sigma_k^2(\tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S}) = \sigma_k^2(\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S}),$$

where in the last equality we dropped $\tilde{\mathbf{U}}$, which is allowed since it is a full rotation. Now, we obtain

$$\sigma_k^2(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}_1\mathbf{S}_1\mathbf{\Omega}\mathbf{S}) = \sigma_k^2(\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S}) \ge \sigma_k^2(\tilde{\mathbf{\Sigma}})\sigma_k^2(\tilde{\mathbf{V}}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S}) = \sigma_k^2(\mathbf{Y})\sigma_k^2(\tilde{\mathbf{V}}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S}). \tag{9}$$

The last equality is because the singular values of $\tilde{\mathbf{\Sigma}}$ are exactly the singular values of $\mathbf{Y} = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^{\mathrm{T}}$. The first inequality follows from the fact that for any two matrices $\mathbf{A}$ and $\mathbf{B}$ where $\mathbf{A}$ has full column rank and $\mathbf{B}$ has full row rank: $\sigma_{\min}(\mathbf{A}\mathbf{B}) \ge \sigma_{\min}(\mathbf{A})\sigma_{\min}(\mathbf{B})$. To prove this fact, notice that $\sigma_{\min}(\mathbf{A}) = \frac{1}{\|\mathbf{A}^+\|_2}$, $\sigma_{\min}(\mathbf{B}) = \frac{1}{\|\mathbf{B}^+\|_2}$, and $\sigma_{\min}(\mathbf{A}\mathbf{B}) = \frac{1}{\|(\mathbf{A}\mathbf{B})^+\|_2} = \frac{1}{\|\mathbf{B}^+\mathbf{A}^+\|_2}$, where the latter equality follows because if $\mathbf{A}$ is full column rank and $\mathbf{B}$ full row rank, then $(\mathbf{A}\mathbf{B})^+ = \mathbf{B}^+\mathbf{A}^+$ [17]. By submultiplicativity $\|\mathbf{B}^+\mathbf{A}^+\|_2 \le \|\mathbf{B}^+\|_2\|\mathbf{A}^+\|_2$. In our case, $\mathbf{A} = \tilde{\mathbf{\Sigma}}$ and $\mathbf{B} = \tilde{\mathbf{V}}^{\mathrm{T}}\mathbf{\Omega}\mathbf{S}$.

Thus, using (6) and (7) in (9), we get that with probability at least 0.9,

$$\sigma_k^2(\mathbf{Z}^{\mathrm{T}}\mathbf{\Omega}_1\mathbf{S}_1\mathbf{\Omega}\mathbf{S}) \ge \tfrac{1}{2}(1 - \sqrt{k/r})^2. \tag{10}$$

We now bound $\|(\mathbf{A} - \mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A})\mathbf{\Omega}_1\mathbf{S}_1\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2$:

$$\begin{aligned}
\|(\mathbf{A} - \mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A})\mathbf{\Omega}_1\mathbf{S}_1\mathbf{\Omega}\mathbf{S}\|_2^2 &= \|(\mathbf{A} - \mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A})\mathbf{\Omega}_1\mathbf{S}_1\mathbf{I}_c\mathbf{\Omega}\mathbf{S}\|_2^2 \\
&\le \|(\mathbf{A} - \mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A})\mathbf{\Omega}_1\mathbf{S}_1\|_{\mathrm{F}}^2 \cdot \|\mathbf{I}_c\mathbf{\Omega}\mathbf{S}\|_2^2 \\
&\le (1 + \sqrt{c/r})^2\|(\mathbf{A} - \mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A})\mathbf{\Omega}_1\mathbf{S}_1\|_{\mathrm{F}}^2, \qquad (11)
\end{aligned}$$

where the first inequality is by spectral submultiplicativity and the second is because $\mathbf{I}_c$ is the input to DetrministicSampling II with $\ell_2 = c$, and so $\|\mathbf{I}_c\mathbf{\Omega}\mathbf{S}\|_2^2 \le 1 + \sqrt{c/r}$. Similarly, we bound $\|\mathbf{E}\mathbf{\Omega}_1\mathbf{S}_1\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2$:

$$\|\mathbf{E}\mathbf{\Omega}_1\mathbf{S}_1\mathbf{\Omega}\mathbf{S}\|_{\mathrm{F}}^2 \le (1 + \sqrt{c/r})^2\|\mathbf{E}\mathbf{\Omega}_1\mathbf{S}_1\|_{\mathrm{F}}^2.$$

From Lemma 9,

$$\mathbb{E}\left[\|\mathbf{E}\mathbf{\Omega}_1\mathbf{S}_1\|_{\mathrm{F}}^2\right] = \|\mathbf{E}\|_{\mathrm{F}}^2,$$

and

$$\mathbb{E}\left[\|(\mathbf{A}-\mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A})\mathbf{\Omega}_1\mathbf{S}_1\|_{\mathrm{F}}^2\right] = \|\mathbf{A}-\mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2.$$

By a simple application of Markov's inequality and the union bound, both of the equations below hold with probability at least 0.6,

$$\begin{aligned} \|\mathbf{E}\mathbf{\Omega}_1\mathbf{S}_1\|_{\mathrm{F}}^2 &\leq 5\|\mathbf{E}\|_{\mathrm{F}}^2; &(12)\\ \|(\mathbf{A}-\mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A})\mathbf{\Omega}_1\mathbf{S}_1\|_{\mathrm{F}}^2 &\leq 5\|\mathbf{A}-\mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2. &(13)\end{aligned}$$

Using (10),(11),(12) and (13) in (14), together with a union bound, we obtain that with probability at least 0.5

$$\|\mathbf{A}-\mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq \|\mathbf{E}\|_{\mathrm{F}}^2 + 20\gamma\left(\frac{1+\sqrt{c/r}}{1-\sqrt{k/r}}\right)^2\left(\|\mathbf{A}-\mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2+\|\mathbf{E}\|_{\mathrm{F}}^2\right). \quad (14)$$

We now use (5) in (14) and the fact that $\|\mathbf{A}-\mathbf{A}_k\|_{\mathrm{F}}^2 \leq \|\mathbf{A}-\mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2$; since (5) holds with probability at least 0.9 and (14) holds with probability at least 0.5, using a union bound, we conclude that with probability at least 0.4,

$$\|\mathbf{A}-\mathbf{X}_{out}\mathbf{X}_{out}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2 \leq \left(15+320\gamma\left(\frac{1+\sqrt{c/r}}{1-\sqrt{k/r}}\right)^2\right)\|\mathbf{A}-\mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2.$$

The result follows because $\mathcal{F}_{opt} = \|\mathbf{A}-\mathbf{X}_{opt}\mathbf{X}_{opt}^{\mathrm{T}}\mathbf{A}\|_{\mathrm{F}}^2$ and $c = 16k\log(20k)$. (Note, we have made no attempt to optimize the constants.)

The running time of Algorithm 6 is $O\left(mnk + rk^3\log(k) + r\log r\right)$, since it employs the approximate SVD of Lemma 6, the randomized technique of Lemma 9, and the deterministic technique of Lemma 7.

6 Related work

Feature selection has received considerable attention in the machine learning and pattern recognition communities. A large number of different techniques appeared in prior work, addressing feature selection within the context of both clustering and classification. Surveys include [16], as well as [19], which reports the results of the NIPS 2003 challenge in feature selection. Popular feature selection techniques include the Laplacian scores [22], the Fisher scores [14], or the constraint scores [41]. None of these feature selection algorithms have theoretical guarantees on the performance of the clusters obtained using the dimension-reduced features.

We focus our discussion of related work on the family of feature selection methods that resemble our approach, in that they select features by looking at the right singular vectors of the data matrix $\mathbf{A}$. Given the input $m \times n$ object-feature matrix $\mathbf{A}$, and a positive integer k, a line of research tries to construct features for (unsupervised) data reconstruction, specifically for Principal Components Analysis (PCA). PCA corresponds to the task of identifying a subset of k linear combinations of columns from $\mathbf{A}$ that best reconstruct $\mathbf{A}$. Subset selection for PCA asks to find the columns of $\mathbf{A}$ that reconstruct $\mathbf{A}$ with comparable error as do its top Principal Components. Jolliffe [24] surveys various methods for the above task. Four of them (called $B1$, $B2$, $B3$, and $B4$ in [24]) employ the Singular Value Decomposition of $\mathbf{A}$ in order to identify columns that are somehow correlated with its top k left singular vectors. In particular, $B3$ employs a deterministic algorithm which is very similar to Algorithm 6 that we used in this work; no theoretical results are reported. An experimental evaluation of the methods of [24] on real datasets appeared in [25]. Another approach employing the matrix of the top k right singular vectors of $\mathbf{A}$ and a Procrustes-type criterion appeared in [26]. From an applications perspective, [39] employed the methods of [24] and [26] for gene selection in microarray data analysis.

Feature selection for clustering seeks to identify those features that have the most discriminative power among all the features. [29] describes a method where one first computes the matrix $\mathbf{V}_k \in \mathbb{R}^{n\times k}$, and then clusters the rows of $\mathbf{V}_k$ by running, for example, the k-means algorithm. One finally selects those k rows of $\mathbf{V}_k$ that are *closest* to the centroids of the clusters computed by the previous step. The method returns those columns from $\mathbf{A}$ that correspond to the selected rows from $\mathbf{V}_k$. A different approach is described in [9]. The method in [9] selects features one at a time; it first selects the column of $\mathbf{A}$ which is *most correlated* with the top left singular vector of $\mathbf{A}$, then projects $\mathbf{A}$ to this singular vector, removes the projection from $\mathbf{A}$, computes the top left singular vector of the resulting matrix, and selects the column of $\mathbf{A}$ which is *most correlated* with the latter singular vector, etc. Greedy approaches similar to the method of [9] are described in [31] and [32]. There are no known theoretical guarantees for any of these methods. While these methods are superficially similar to our method, in that they use the right singular matrix $\mathbf{V}_k$ and are based on some sort of greedy algorithm, the techniques we developed to obtain theoretical guarantees are entirely different and based on linear-algebraic sparsification results [3, 4].

The result most closely related to ours is the work in [6, 8]. This work provides a randomized algorithm which offers a theoretical guarantee. Specifically, for $r = \Omega(k\epsilon^{-2}\log k)$, it is possible to select r features such that the optimal clustering in the reduced-dimension space is a $(3+\epsilon)$-approximation to the optimal clustering. Our result improves upon this in two ways. First, our algorithms are deterministic; second, by using our deterministic algorithms in combination with this randomized algorithm, we can select $r = O(k)$ features and obtain a competitive theoretical guarantee.

Next, we should mention that if one allows linear combinations of the features (feature *extraction* rather than feature *selection*), then there are algorithms that offer theoretical guarantees. First there is the SVD itself, which constructs k (mixed) features for which the optimal clustering in this feature space is a 2-approximation to the optimal clustering [12]. It is possible to improve the efficiency of this SVD algorithm considerably by using the approximate SVD (as in Lemma 6) instead of the exact SVD to get nearly the same approximation guarantee with k features. The exact statement of this improvement can be found in [8]. Boutsidis et al. [7] show how to select $O(k\epsilon^{-2})$ (mixed) features with random projections and also obtaining a $(2+\epsilon)$-guarantee. While these algorithms are interesting, they do not produce features that preserve the integrity of the original features. The focus of this work is on what one can achieve while preserving the original features.

A complementary line of research [20, 21, 15, 13, 2, 1] approaches the k-means problem by subsampling the points of the dataset; such a subset of points is called *coreset*. The idea here is to select a small subset of the points and by using only this subset obtain a partition for all the points that is as good as the partition that would have been obtained by using all the points. [20, 21, 15, 13, 2, 1] offer algorithms for $(1+\epsilon)$ approximate partitions. Note that we were able to give only constant factor approximations. For example, [15] shows the existence of an $(1+\epsilon)$-approximate coreset of size $r = O(k^3/\epsilon^{n+1})$ (n is the number of features). [13] provides a coreset of size $r = poly(k, \epsilon^{-1})$. The techniques used for all these coresets are different from the techniques we used for feature selection; further, there is no clear way on how to use a coreset selection algorithm to select features. Moreover, the authors of [23] use the coreset-based algorithm from [2] to design a PTAS for k-means and other clustering problems. It would be interesting to understand whether the techniques from [20, 21, 15, 13, 2, 1, 23] are useful for feature selection as well. In particular, it appears that there is potential to obtain relative error feature selection k-means algorithms by using such approaches.

Acknowledgements

Christos Boutsidis acknowledges the support from XDATA program of the Defense Advanced Research Projects Agency (DARPA), administered through Air Force Research Laboratory contract FA8750-12-C-0323.

References

[1] A. Aggarwal, A. Deshpande, and R. Kannan. Adaptive sampling for k-means clustering. In *Proceedings of the Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*, 2009.

[2] D. Arthur and S. Vassilvitskii. k-means++: the advantages of careful seeding. In *Proceedings of the Annual ACM-SIAM Symposium on Discrete algorithms (SODA)*, 2007.

[3] J. Batson, D. Spielman, and N. Srivastava. Twice-ramanujan sparsifiers. In *Proceedings of the ACM symposium on Theory of Computing (STOC)*, 2009.

[4] C. Boutsidis, P. Drineas, and M. Magdon-Ismail. Near-optimal column-based matrix reconstruction. In *Proceedings of the Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2011.

[5] C. Boutsidis, P. Drineas, and M. Magdon-Ismail. Near-optimal column-based matrix reconstruction. Technical Report, available on ArXiv:1103.0995.

[6] C. Boutsidis, M. W. Mahoney, and P. Drineas. Unsupervised feature selection for the k-means clustering problem. In *Neural Information Processing Systems (NIPS)*, 2009.

[7] C. Boutsidis, A. Zouzias, and P. Drineas. Random projections for k-means clustering. In *Neural Information Processing Systems (NIPS)*, 2010.

[8] C. Boutsidis, A. Zouzias, M.W. Mahoney, and P. Drineas. Stochastic Dimensionality Reduction for k-means clustering. *Manuscript, Under Review*, 2011.

[9] Y. Cui and J. G. Dy. Orthogonal principal feature selection. *manuscript.*

[10] A. d Aspremont, L. El Ghaoui, M. Jordan, and G. Lanckriet. A direct formulation for sparse PCA using semidefinite programming. *SIAM review*, 49(3):434, 2007.

[11] P. Drineas, M. Mahoney, and S. Muthukrishnan Relative-Error CUR Matrix Decompositions. *SIAM Journal on Matrix Analysis and Applications*, 30:844-881, 2008.

[12] P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay. Clustering in large graphs and matrices. In *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 1999.

[13] D. Feldman, M. Monemizadeh, and C. Sohler. A PTAS for k-means clustering based on weak coresets. In *Proceedings of the Annual Symposium on Computational Geometry (SoCG)*, 2007.

[14] D. Foley and J. Sammon, J.W. An optimal set of discriminant vectors. *IEEE Transactions on Computers*, C-24(3):281–289, March 1975.

[15] G. Frahling and C. Sohler. A fast k-means implementation using coresets. In *Proceedings of the Annual Symposium on Computational Geometry (SoCG)*, 2006.

[16] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research (JMLR)*, 3:1157–1182, 2003.

[17] T.N.E. Greville. Note on the generalized inverse of a matrix product. *SIAM Review*, 8(4):518–521, 1966.

[18] G.H. Golub and C.F. Van Loan. Matrix computations. *Johns Hopkins Univ Pr*, 1996.

[19] I. Guyon, S. Gunn, A. Ben-Hur, and G. Dror. Result analysis of the nips 2003 feature selection challenge. In *Neural Information Processing Systems (NIPS)*, 2005.

[20] S. Har-Peled and A. Kushal. Smaller coresets for k-median and k-means clustering. In *Proceedings of the Annual Symposium on Computational Geometry (SoCG)*, 2005.

[21] S. Har-Peled and S. Mazumdar. On coresets for k-means and k-median clustering. In *Proceedings of the Annual ACM Symposium on Theory of Computing (STOC)*, 2004.

[22] X. He, D. Cai, and P. Niyogi. Laplacian score for feature selection. In *Advances in Neural Information Processing Systems*, 2006.

[23] R. Jaiswal1 and A Kumar and S. Sen A simple D^2-sampling based PTAS for k-means and other Clustering problems. *Manuscript*, http://arxiv.org/pdf/1201.4206.pdf, 2012.

[24] I. Jolliffe. Discarding variables in a principal component analysis. I: Artificial data. *Applied Statistics*, 21(2):160–173, 1972.

[25] I. Jolliffe. Discarding variables in a principal component analysis. II: Real data. *Applied Statistics*, 22(1):21–31, 1973.

[26] W. Krzanowski. Selection of variables to preserve multivariate data structure, using principal components. *Applied Statistics*, 36(1):22–33, 1987.

[27] A. Kumar, Y. Sabharwal, and S. Sen. A simple linear time $(1+\epsilon)$-approximation algorithm for k-means clustering in any dimensions. In *Proceedings of the Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2004.

[28] S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[29] Y. Lu, I. Cohen, X. S. Zhou, and Q. Tian. Feature selection using principal feature analysis. In *Proceedings of the 15th international conference on Multimedia*, pages 301–304, 2007.

[30] M. Magdon-Ismail. Row Sampling for Matrix Algorithms via a Non-Commutative Bernstein Bound. *ArXiv Report*, http://arxiv.org/abs/1008.0587, 2010.

[31] A. Malhi and R. Gao. PCA-based feature selection scheme for machine defect classification. *IEEE Transactions on Instrumentation and Measurement*, 53(6):1517–1525, Dec. 2004.

[32] K. Mao. Identifying critical variables of principal components for unsupervised feature selection. *IEEE Transactions on Systems, Man, and Cybernetics*, 35(2):339–344, April 2005.

[33] R. Ostrovsky and Y. Rabani. Polynomial time approximation schemes for geometric k-clustering. In *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS)*, 2000.

[34] R. Ostrovsky, Y. Rabani, L. J. Schulman, and C. Swamy. The effectiveness of Lloyd-type methods for the k-means problem. In *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS)*, 2006.

[35] M. Rudelson. Random Vectors in the Isotropic Position. *Journal of Functional Analysis*, 164, 1:60–72, 1999.

[36] M. Rudelson and R. Vershynin. Sampling from large matrices: An approach through geometric functional analysis. *Journal of the ACM (JACM)*, 54, 2007.

[37] A. Smola and B. Schölkopf. Sparse greedy matrix approximation for machine learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2000.

[38] J. Vaidya and C. Clifton Privacy-preserving k-means clustering over vertically partitioned data. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*, 2003

[39] A. Wang and E. A. Gehan. Gene selection for microarray data analysis using principal component analysis. *Stat Med*, 24(13):2069–2087, July 2005.

[40] X. Wu et al. Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14(1):1–37, 2008.

[41] D. Zhang, S. Chen, and Z.-H. Zhou. Constraint score: A new filter method for feature selection with pairwise constraints. *Pattern Recognition*, 41(5):1440–1451, 2008.