Experiments with Random Projections for Machine Learning

Dmitriy Fradkin
Division of Computer and Information Sciences
Rutgers University Piscataway, NJ
dfradkin@paul.rutgers.edu

David Madigan
Department of Statistics Rutgers University
Piscataway, NJ
madigan@stat.rutgers.edu

ABSTRACT

Random Projections have recently appeared as a tool for dimensionality reduction and have been used to produce a number of results, both theoretical and applied. In this paper we report a number of experiments to evaluate random projections in the context of inductive supervised learning. In particular, we compare random projections and PCA on a number of different datasets and using different machine learning methods. While we find that the random projection approach predictively underperforms PCA, its computational advantages may make it attractive for certain applications.

1. INTRODUCTION

Inductive supervised learning infers a functional relation $y = f(\mathbf{x})$ from a set of training examples

$$T = \{(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_n}, y_n)\}.$$

In what follows the inputs are vectors $[x_{i_1}, \ldots, x_{i_p}]$ in $\Re^p$, $y \in \{-1, 1\}$, and we refer to f as a classifier. The objective of this exercise is usually to produce a classifier that makes accurate predictions for future input vectors.

Applications where p is large pose particular challenges. The computational complexity of many algorithms is quadratic or even exponential in p. Furthermore, large p usually requires some sort of complexity control to avoid over-fitting the training data. A standard and widely used approach to dealing with large p is to first apply a dimensionality reduction procedure, ideally a procedure that preserves properties of the original space, such as distances or angles. Principal Components Analysis (PCA) is a popular choice. PCA's main drawback is its computational complexity which precludes its use in truly large-scale applications. In the 1990's, an alternative approach based on Random Projections (RP) received some attention in the literature. The computational cost of RP is low but it enjoys distance-preserving properties that make it an attractive candidate for certain dimensionality reduction tasks.

In this paper we describe experiments that examine the efficacy of RP for supervised learning and compare it with PCA. Previous papers have explored RP for clustering, mixture models, and other applications, but not, as far as we know, for supervised learning. We first discuss the theoretical background of PCA and, at some more length, random projections. We also present a short survey of the literature on Random Projections. We then proceed to describe the datasets we've used and the setup of the experiments we conducted. We finish by discussing the results.

2. METHODS

2.1 Principal Component Analysis (PCA)

2.1.1 *Theoretical Background*

There are several ways of looking at PCA. It can be seen as a method of transforming correlated variables into uncorrelated ones, as a method of finding linear combinations of original variables with large (or small) variance or as a way of doing dimensionality reduction [10].

Here we follow [12] in describing PCA as a dimensionality reduction / data approximation method. Given n data points in $\Re^p$, as an $n \times p$ matrix X, we want to find the best (in least squares sense) q-dimensional approximation for the data ($q < p$). That is, we want to define $f(\lambda) = \mu + V_q\lambda$ where μ is a location vector in $\Re^p$, V is a $p \times q$ matrix with q orthogonal unit vectors as columns and λ is a q vector of parameters, so as to minimize $\sum^N{}_{i=1} ||x_i - \mu - V_q\lambda||^2$.

If we first centralize X, then minimizing this sum leads to V_q being the first q columns of matrix V in a singular value decomposition (SVD) of data matrix X: $X = UDV^T$, where U is an $n \times p$ orthogonal matrix ($U^TU = I_p$), whose columns are left singular vectors, V is a $p \times p$ orthogonal matrix ($V^TV = I_p$) whose columns are right singular vectors, and U is a $p \times p$ diagonal matrix with diagonal elements $d_1 \geq d_2 ... \geq d_p \geq 0$ which are singular values of X.

This means that columns of V_q are unit eigenvectors corresponding to the q largest eigenvalues of X. The rows of an $N \times q$ matrix UD are the principal components and give λ vectors for representing corresponding data points. So PCA consists in finding eigenvectors of the data matrix that correspond to largest singular values and projecting data onto them.

We note that there is an alternative method of performing PCA involving the correlation matrix. If X is centered, correlation matrix is $S = \frac{1}{n-1}X^TX$. Therefore, using SVD decomposition of X, we have

$$S = \frac{1}{n-1}(UDV^T)^T * UDV^T = \frac{1}{n-1}VD^2V^T = VD'V^T$$

where $D' = \frac{1}{n-1}D^2$. Notice that the ordering of the diagonal elements of D' is the same as in D and the eigenvectors are the same for S as for X. It follows that PCA can be done either on X directly, or on S.

In our experiments, we normalize the data. This way we don't need to worry about μ and relative scale of components. Then, given a data point X_i (a row of matrix X), we can project it into a q-dimensional space spanned by rows of V_q as follows: $X_i' = X_iV_q$.

2.1.2 Complexity

The computational complexity of PCA is $O(p^2n) + O(p^3)$. Computing the SVD decomposition, as we do, is somewhat more efficient. For sparse matrices of rank r there are $O(prn)$ algorithms ([4]). Performing the projection itself is just a matrix multiplication and takes $O(npq)$. We note that projecting to a dimension greater than the rank of the original matrix is pointless, since values of attributes after r-th will all be zero.

2.2 Random Projections

2.2.1 Theory of Random Projections

A theorem due to Johnson and Lindenstrauss (JL Theorem) states that for a set of points of size n in p-dimensional Euclidean space there exists a linear transformation of the data into a q-dimensional space, $q \geq O(\epsilon^{-2}log(n))$ that preserves distances up to a factor $1 \pm \epsilon$ ([1]).

Dasgupta and Gupta [8], present a simpler proof of the JL Theorem, giving tighter bounds on ϵ and q, as follows:

$$q \geq 4 * (\frac{\epsilon^2}{2} - \frac{\epsilon^3}{3})^{-1} ln(n). \tag{1}$$

In this paper the authors also indicate that a matrix whose entries are normally distributed represents such a mapping with probability at least $1/n$, and therefore doing $O(n)$ projections will result in projection with an arbitrarily high probability of preserving distances.

Achlioptas [1] shows that there are simpler ways of producing random projections. He also explicitly incorporates probability into his results:

THEOREM 1. *Given n points in $\Re^p$ (in form of an $n \times p$ matrix X), choose ϵ, $\beta > 0$ and $q \geq \frac{4+2*\beta}{\frac{\epsilon^2}{2} - \frac{\epsilon^3}{3}} ln(n)$, and let $E = \frac{1}{\sqrt{q}}XP$, for projection matrix P. Then mapping from X to E preserves distances up to factor $1 \pm \epsilon$ for all rows in X with probability $(1 - n^{-\beta})$. Projection matrix P, $p \times q$, can be constructed in one of the following ways:*

- *$r_{ij} = \pm 1$ with probability 0.5 each*
- *$r_{ij} = \sqrt{3} * (\pm 1$ with probability 1/6 each, or 0 with probability 2/3)*

These projections have an added benefit of being easy to implement and to compute.

We chose to implement the first of the methods suggested by Achlioptas. Since we are not concerned with preserving distances per se, but only with preserving separation between points, we do not scale our projection by $\frac{1}{\sqrt{q}}$.

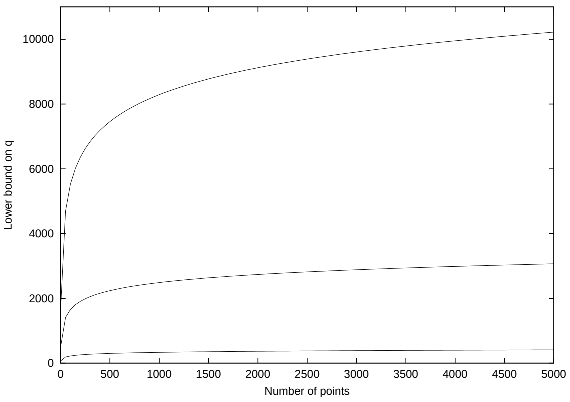

Figure 1: Plot of lower bound q of dimensionality of random projections as a function of number of points. Upper curve corresponds to $\epsilon = 0.1$, middle one - to $\epsilon = 0.2$, lowest one to $\epsilon = 0.5$

2.2.2 Complexity and Theoretical Effectiveness

The computational complexity of RP is easy to compute: projection of n points from $\Re^p$ to $\Re^q$ requires constructing an $p \times q$ projection matrix, which takes O(pq). Performing the projection itself is just a matrix multiplication and takes $O(npq)$.

We can use Theorem 1 to compute theoretical limitations on dimensionality of a random projection. We limit ourselves to examining case $\beta = 1$, allowing a deviation by a factor greater than ϵ with probability $\frac{1}{n}$. Figure 1 shows a graph of this bound on q for different sizes (n) of dataset and different values of ϵ.

2.2.3 Applications and Experiments

In this section we mention some results from the literature on random projections.

A paper by Kaski [15] on random mappings that preserve similarity (defined as a cosine of the angle between vectors), describes how random projections were used on textual data in WEBSOM, a program that organizes document collections into Self-Organizing Map.

In a recent paper by Magen [16], the author shows how volumes and affine distances can be preserved. This result includes Kaski's observations as a special case (since preserving volume in 2D is equivalent to preserving distances and angles).

Bingham and Manilla [4] compare several dimensionality reduction methods, such as PCA (based on data covariance matrix), SVD (PCA performed directly on the data matrix), RP (using the second method of construction projection described in [1]) and Discrete Cosine Transform (DCT) on large-dimensional noiseless and noisy image data and on the Newsgroups text dataset (available from UCI archives). Their experiments involve comparing computational complexity and similarity preservation. Their results indicate that random projections are computationally simple while providing a high degree of accuracy. They note that JL Theorem and results in [1] and other papers give much higher bounds than those sufficing for good results.

Dasgupta [6], [7] describes experiments on learning mixtures of Gaussians in high dimensions using random projections and PCA. His results show that data from a mixture of

Table 1: Description of Datasets

Name	# Instances	# Attributes
Ionosphere	351	34
Colon	62	2000
Leukemia	72	3571
Spam	4601	57
Ads	3279	1554

k Gaussians can be projected into $O(\log k)$ dimensions while retaining the approximate level of separation. He also concludes that RPs result in more spherical clusters than those in the original dimension. RPs also do better than PCA on eccentric data (where PCA might fail completely). Dasgupta also combines RP with the Expectation Maximization (EM) algorithm and applies it to a hand-written digit dataset, achieving good results.

Indyk and Motwani [13] apply random projections to the nearest neighbor problem. This leads to an algorithm with polynomial preprocessing and query time polynomial in p and $\log n$. However, according to authors, since the exponent depends on $1/\epsilon$, this result is purely theoretical for small ϵ.

Engebretsen, Indyk and O'Donnell [9] present a deterministic algorithm for constructing mappings of the type described in the JL lemma (by use of the method of conditional probabilities) and use it to derandomize several randomized algorithms, such as MAXCUT and coloring. They also present a formal lemma relating the JL Theorem and preservation of angles.

Thaper, Guha, Indyk and Koudas also use random projections to enable their method of computing dynamic multidimensional histograms [17]. RPs are used their to perform approximate similarity computations between different histogram models, represented as high dimensional vectors.

A similar application of random projections in a different area is suggested in [2]. Authors argue for using random projections as a way of speeding up kernel computations in methods such as Kernel Principal Components analysis.

We are not aware of any existing results on using random projections in a supervised learning context.

3. DESCRIPTION OF DATA

Table 1 describes the datasets that we have used in our experiments. Ionosphere, Spambase and Internet Ads were taken from UCI repository [5]. Datasets Colon and Leukemia were first used in [3] and [11] respectfully.

Datasets are used without modifications, except for the Ads dataset that originally contained 3 more attributes with missing values. These attributes were removed.

A two-class classification problem is of primary substantive interest in each of these datasets. The attributes in each case are real-valued with the exception of the ads dataset; this dataset features binary attributes which we treat as continuous in the experiments. Our goal was to select datasets representing different "shapes." Ionosphere and Spam have many more instances than attributes. Colon and Leukemia are the other way around. Ads is somewhat in between. Each of the datasets has enough attributes that dimensionality reduction is of practical interest.

4. EXPERIMENTAL SETUP

We compare PCA and RP using a number of standard machine learning tools, such as decision trees (C4.5 - [18]), nearest neighbor methods and SVM (SVMLight - [14]). We are using the default settings with all of these methods. Our purpose is not to compare the performance of these methods to each other, but to see the differences in their performance when using PCA or RP.

Our experiments are set up in the following way:

Algorithm 1 Experiment Pseudocode

```
Require: Dataset $D$, set of projection dimensions $\{d_1, \ldots, d_k\}$, number of test/training splits $s$ to be done (we perform 30 splits for Ads and 100 splits for other datasets)
for $i = 1, \ldots, s$ do
    split $D$ into training set and test set
    normalizing the data (estimating mean and variance from the training set)
    for $d' = d_1, \ldots, d_k$ do
        do a PCA on training set and project both training and test data into $\Re^{d'}$
        create a random projection matrix as described above and project both training and test data into $\Re^{d'}$
        train learning methods on training sets and then apply them to respective test sets to evaluate performance
    end for
end for
```

For a given dataset we keep the size of the test set constant over different splits. These size are as follows: Ionosphere - 51, Spambase - 1601, Colon - 12, Leukemia - 12, Ads - 1079. For small datasets we try to leave sufficient number of instances for training, while for larger datasets we take approximately a third of instances.

We note that such test sets also sets upper bound on the rank of training data matrix. Since the training set size for the Colon dataset is 50, and for Leukemia it is 60, PCA projections to spaces of higher dimensions will not yield better results.

In this respect PCA is quite different from RP, where projection to dimension lower than $O(ln(n))$ is considered ineffective. Since our purpose it to compare them, we perform projection to both low-dimensional and (relatively) high dimensional spaces. The theoretical quality of distance preservation with RP is quite low for all of these (compare with Figure 1). However, the literature shows that the theoretical bounds are quite conservative ([4]).

Colon and Leukemia datasets are of a high dimensionality but have few points. Thus we would expect RP to high dimensions to lead to good results, while PCA results should stop changing after some point. For these dataset we perform projections into spaces of dimensionality 5, 10, 25, 50, 100, 200 and 500.

Ionosphere and Spam are relatively low-dimensional but have many more points than Colon and Leukemia datasets. Such combination in theory leaves little space for RP to improve, while PCA should be able to do well. We project to dimensions 5, 10, 15, 20, 25 and 30.

Ads dataset is both large and high-dimensional, and seems to fall somewhere between the others. We perform projections are done to 5, 10, 25, 50, 100, 200 and 500.

5. RESULTS

The results of experiments can be seen in Table 3. All the columns contain accuracies for the specified methods. Entries in all columns are averages over all splits for a given dataset, with lower numbers in each cell being standard deviations. The last row in each subtable contains results for the original dimension and therefore is only given once.

In order to demonstrate differences in performance of PCA and RP with different methods, we also plot accuracies using PCA and RP for each dataset and learning method (Figures in Table 2). Accuracy of the learning method in the original space is drawn as a straight line on each graph for comparison (even though it is not a function of dimension).

Nearest Neighbor Methods appear to be least affected by reduction in dimensionality through PCA or RP, in the sense that their performance deteriorates less than that of C4.5 or of SVM. In some cases PCA projection into a low dimensional space actually improves NN's accuracy (on Ionosphere and Ads datasets). NN results with RP approach those in the original space (or PCA) quite rapidly. Such behavior of NN methods is to be expected since they rely on distance computations for their performance and are not concerned with separation of classes or informativeness of individual attributes. Thus one might expect that Nearest Neighbor methods would stand to benefit most from Random Projections.

SVM does worse in projection spaces (both with PCA and RP) than in the original space, but its performance improves noticeably as the dimensionality of projections increases. Performance of PCA is much better initially, but RPs are catching up to it. We kept track of the number of support vectors used in each projection (the averages are given in Table 4), since these numbers can serve as indicators of the complexity of data. Using PCA on Ads, Colon and Leukemia datasets led to fewer support vectors, while on Spam and Ionosphere data the number of support vectors was somewhat higher for PCA than in the original space. Using RP led to about the same number of support vectors on Colon and Leukemia Datasets but resulted in a much higher numbers on Ads, Spam and Ionosphere. A possible explanation is that since Colon and Leukemia datasets have few points, the training data was easy to separate even in a low dimensional space. For the datasets that have many points, the opposite was true. The number of support vectors when using PCA was always less than when using RP in lower dimensions, indicating that the data produced with RP is more difficult to separate. However, both for PCA and RP, as the dimensionality of the projections approached the original dimensionality, the number of support vectors approached that used in the original space.

C4.5 does very well with low-dimensional PCA projections (on Ionosphere, Colon and Leukemia datasets), but its performance deteriorates after that and doesn't improve. Its performance with RP is also poor: after some initial improvement it seems to level out. Decision trees rely on informativeness of individual attributes and construct axis-parallel boundaries for their decision. Therefore, decision trees don't deal well with transformations of the attributes, and they are also quite sensitive to noise ([12]). These theoretical characteristics of decision trees support our experimental results and indicate that Random Projections and decision trees are perhaps not a good combination.

6. CONCLUSION

In this paper we compared the performance of different learning methods in conjunction with the dimensionality reduction techniques PCA and RP. While PCA is known to give good results and has a lot of useful properties, it is computationally expensive and is not feasible on large, high-dimensional data.

Random Projections are much cheaper computationally and also possess some desirable theoretical properties. In our experiments PCA predictively outformed RP. Nonetheless RP offers clear computational advantages. Furthermore, some trends in our results indicate that the predictive performance of RP does improve with increasing dimension, particularly in combination with right learning methods.

Our results indicate that RPs are best suited for use with Nearest Neighbor methods. They also combine well with SVM. However decision trees with RP were less satisfactory.

We hope that these results demonstrate potential usefulness of Random Projections in supervised learning context and will encourage further experimentation.

7. ACKNOWLEDGMENTS

We would like to thank Andrei Anghelescu for providing the kNN code.

8. REFERENCES

[1] D. Achlioptas. Database-friendly random projections. In *Symposium on Principles of Database Systems (PODS)*, pages 274–281, 2001.

[2] D. Achlioptas, F. McSherry, and B. Schölkopf. Sampling techniques for kernel methods. In S. B. T. G. Dietterich and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems.*

[3] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering of tumor and normal colon tissues probed by oligonucleotide arrays. In *Proc. Natl. Acad. Sci. USA*, volume 96, pages 6745–6750, June 1999.

[4] E. Bingham and H. Mannila. Random projection in dimensionality reduction: applications to image and text data. In *Knowledge Discovery and Data Mining*, pages 245–250, 2001.

[5] C. Blake and C. Merz. (uci) repository of machine learning databases, 1998.

[6] S. Dasgupta. Learning mixtures of gaussians. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 1999.

[7] S. Dasgupta. Experiments with random projections. In *In Proc. 16th Conf. Uncertainty in Artificial Intelligence (UAI)*, 2000.

[8] S. Dasgupta and A. Gupta. An elementary proof of the johnson-lindenstrauss lemma. Technical Report TR-99-006, International Computer Science Institute, Berkeley, California, USA, 1999.

[9] L. Engebretsen, P. Indyk, and R. O'Donnell. Derandomized dimensionality reduction with applications. In *Proceedings of the 13th Symposium on Discrete Algorithms. IEEE*, 2002.

[10] B. Flury. *Common Principal Components and Related Multivariate Models.* Wiley, New York, 1988.

[11] T. R. Golub, D. K. Slonim, P. Tamayo, C. H. M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, , and E. S. Lander. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. In *Science*, volume 286, pages 531–537, 1999.

[12] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning. Data Mining, Inference, and Prediction.* Springer, New York, 2001.

[13] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC)*, pages 604–613, 1998.

[14] T. Joachims. Making large-scale svm learning practical. In B. Schlkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning.* MIT-Press, 1999.

[15] S. Kaski. Dimensionality reduction by random mapping: Fast similarity computation for clustering. In *Proceedings of IJCNN'98, International Joint Conference on Neural Networks*, volume 1, pages 413–418. IEEE Service Center, Piscataway, NJ, 1998.

[16] A. Magen. Dimensionality reductions that preserve volumes and distance to affine spaces, and their algorithmic applications.

[17] P. I. N. Thaper, S. Guha and N. Koudas. Dynamic multidimensional histograms. In *Proc. ACM SIGMOD*, pages 428–439, May 2002.

[18] J. R. Quinlan. C4.5: Programs for machine learning, 1993.

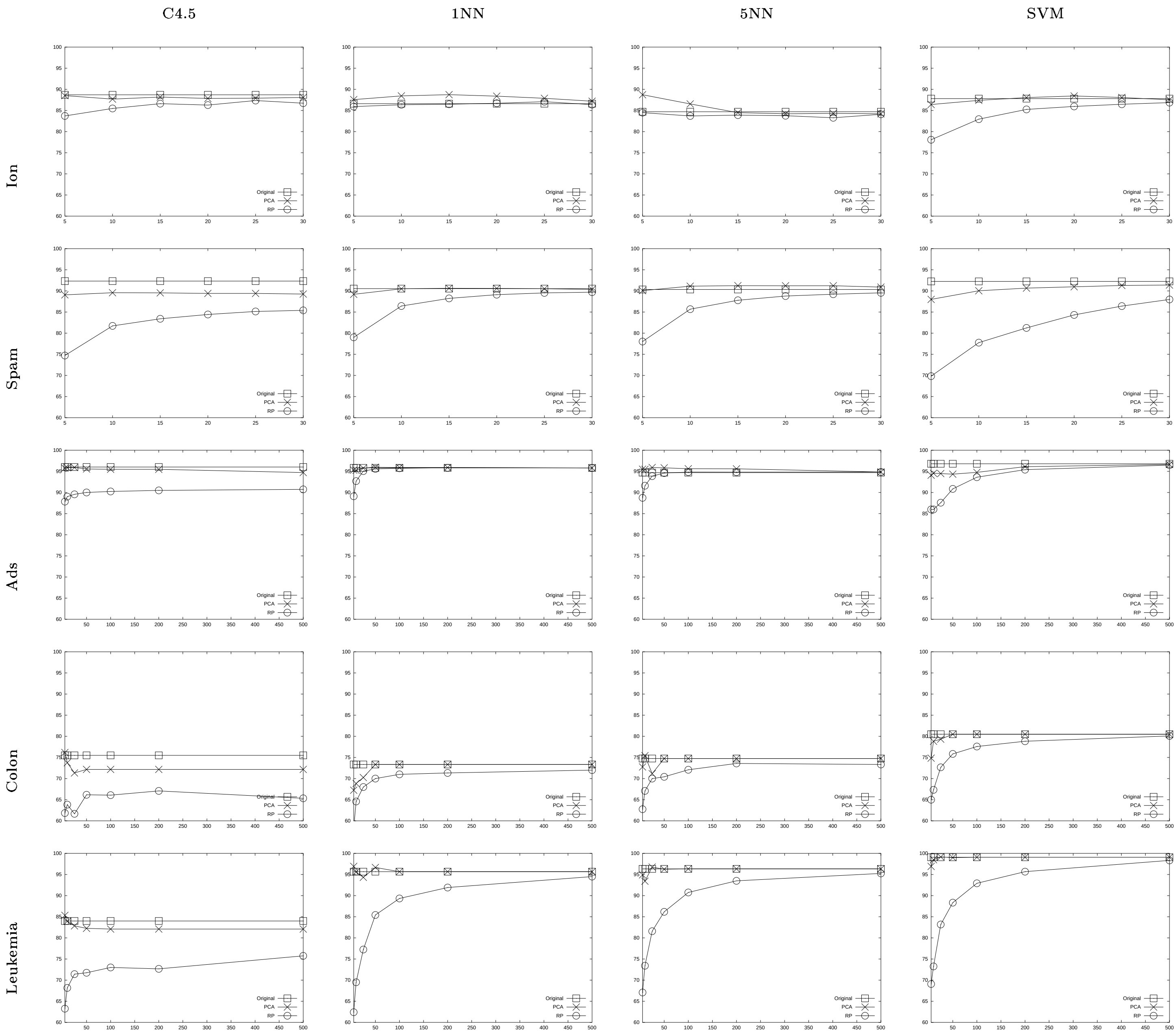

Table 2: Accuracy (Y-axis) using PCA and RP, compared to performance in the original dimension, plotted against the projection dimension (X-axis)

Ads	C4.5		1NN		5NN		SVM	
	PCA	RP	PCA	RP	PCA	RP	PCA	RP
5	95.8	87.8	95.3	89.1	95.5	88.7	94.1	86.0
	0.6	0.7	0.6	0.9	0.5	1.0	1.3	0.9
10	95.9	89.0	95.2	92.7	95.5	91.6	94.5	86.0
	0.6	0.9	0.5	0.8	0.5	1.0	0.8	0.9
25	95.9	89.6	95.8	95.1	95.9	93.9	94.5	87.6
	0.5	0.8	0.6	0.6	0.5	0.6	0.8	1.1
50	95.6	90.0	96.0	95.6	95.8	94.6	94.3	90.9
	0.6	1.1	0.5	0.5	0.6	0.7	0.9	0.9
100	95.5	90.2	95.9	95.7	95.6	94.8	94.8	93.6
	0.6	1.0	0.5	0.5	0.5	0.6	0.7	0.8
200	95.5	90.5	95.9	95.9	95.6	94.8	96.1	95.4
	0.6	0.9	0.5	0.5	0.6	0.7	0.5	0.6
500	94.7	90.7	95.8	95.8	94.9	94.8	96.6	96.5
	0.7	0.9	0.4	0.5	0.5	0.6	0.5	0.4
1554	96.0		95.8		94.7		96.8	
	0.6		0.5		0.6		0.4	

Ion.	C4.5		1NN		5NN		SVM	
	PCA	RP	PCA	RP	PCA	RP	PCA	RP
5	88.5	83.7	87.6	85.9	88.7	84.5	86.4	78.1
	4.4	5.2	3.9	5.4	3.9	5.7	4.3	6.7
10	87.7	85.5	88.5	86.4	86.5	83.7	87.4	82.9
	4.8	4.6	4.1	4.7	4.5	5.4	4.3	5.7
15	88.2	86.6	88.7	86.5	84.5	83.9	88.1	85.3
	4.0	5.2	4.7	5.0	4.6	5.1	4.0	4.9
20	87.9	86.3	88.4	86.7	84.2	83.8	88.4	86.0
	4.5	5.0	4.4	5.0	4.4	5.0	4.3	5.4
25	87.9	87.4	87.9	87.1	84.3	83.3	88.1	86.5
	4.4	4.5	4.5	4.8	4.4	5.7	4.3	4.8
30	88.1	86.7	87.2	86.4	84.2	84.1	87.5	86.9
	4.5	5.2	4.6	4.5	4.5	5.2	4.6	4.8
34	88.7		86.6		84.7		87.8	
	4.7		4.4		4.5		4.5	

Colon	C4.5		1NN		5NN		SVM	
	PCA	RP	PCA	RP	PCA	RP	PCA	RP
5	76.2	61.8	67.2	58.1	72.8	62.8	74.8	65.0
	11.8	14.0	11.0	14.9	10.9	14.7	12.7	14.5
10	73.7	63.8	68.8	64.6	75.4	67.1	78.8	67.3
	12.2	14.2	11.8	12.8	12.4	11.4	12.3	14.5
25	71.3	61.7	70.2	68.0	71.2	70.0	79.3	72.7
	13.2	13.1	11.3	11.2	14.0	11.1	12.0	15.2
50	72.2	66.2	73.3	70.0	74.8	70.4	80.5	75.8
	13.2	13.9	12.4	12.3	13.1	12.3	11.7	13.6
100	72.2	66.1	73.3	71.0	74.8	72.1	80.5	77.6
	13.2	12.3	12.4	12.2	13.1	12.5	11.7	12.8
200	72.2	67.1	73.3	71.3	74.8	73.6	80.5	78.8
	13.2	14.7	12.4	10.9	13.1	12.5	11.7	12.6
500	72.2	65.3	73.3	72.0	74.8	73.3	80.5	80.1
	13.2	12.6	12.4	12.4	13.1	12.7	11.7	12.4
2000	75.5		73.3		74.8		80.5	
	11.9		12.4		13.1		11.7	

Spam	C4.5		1NN		5NN		SVM	
	PCA	RP	PCA	RP	PCA	RP	PCA	RP
5	89.1	74.7	89.2	79.0	90.1	78.0	88.0	69.8
	0.9	3.8	0.8	2.2	0.8	2.6	0.9	5.3
10	89.6	81.7	90.5	86.4	91.1	85.7	90.0	77.8
	0.8	2.1	0.7	1.2	0.7	1.5	0.7	4.1
15	89.5	83.4	90.6	88.2	91.2	87.8	90.7	81.2
	0.8	1.5	0.7	0.9	0.7	1.1	0.8	2.8
20	89.4	84.4	90.6	89.1	91.2	88.8	91.0	84.3
	0.8	1.5	0.7	0.9	0.7	1.0	0.7	2.6
25	89.4	85.1	90.5	89.5	91.2	89.2	91.3	86.4
	0.9	1.2	0.8	0.7	0.7	0.9	0.7	1.9
30	89.3	85.4	90.4	89.7	90.9	89.5	91.4	88.0
	0.8	1.4	0.8	0.9	0.7	0.8	0.6	2.1
57	92.3		90.5		90.3		92.3	
	0.6		0.7		0.6		0.6	

Leuk.	C4.5		1NN		5NN		SVM	
	PCA	RP	PCA	RP	PCA	RP	PCA	RP
5	85.3	63.2	96.9	62.4	94.9	67.1	96.9	69.1
	10.2	16.2	4.8	15.3	6.7	13.3	4.2	13.7
10	83.9	68.2	95.6	69.5	93.4	73.4	98.5	73.2
	10.6	14.3	6.3	13.1	6.7	11.9	3.4	12.3
25	82.8	71.4	94.3	77.2	96.8	81.6	99.2	83.2
	10.5	13.8	6.1	14.1	4.7	13.0	2.5	11.9
50	82.3	71.7	96.7	85.4	96.2	86.2	99.0	88.3
	10.5	13.6	5.5	9.9	6.1	9.0	2.7	8.7
100	82.1	73.0	95.7	89.3	96.3	90.8	99.1	92.9
	10.5	12.5	5.5	8.4	4.9	9.4	2.6	7.1
200	82.1	72.7	95.7	91.9	96.3	93.5	99.1	95.7
	10.5	11.9	5.5	7.8	4.9	6.7	2.6	5.5
500	82.1	75.8	95.7	94.5	96.3	95.2	99.1	98.3
	10.5	14.0	5.5	6.4	4.9	5.8	2.6	3.3
3572	84.0		95.7		96.3		99.1	
	9.0		5.5		4.9		2.6	

Table 3: Accuracies (and their standard deviations) for each dataset and projection

Ads	PCA	RP
5	323.6	629.9
	23.2	21.3
10	316.8	635.6
	26.5	22.2
25	320.1	623.3
	29.2	39.3
50	347.2	548.6
	41.1	34.0
100	347.6	453.3
	33.3	30.5
200	345.2	402.7
	29.6	17.5
500	432.8	386.4
	27.9	16.5
1554	404.3	
	13.4	

Colon	PCA	RP
5	33.4	36.3
	3.8	3.6
10	33.5	36.2
	3.4	2.9
25	35.7	36.5
	2.2	3.0
50	37.6	36.9
	2.0	2.4
100	37.6	37.3
	2.0	2.5
200	37.6	37.5
	2.0	2.5
500	37.6	37.4
	2.0	2.1
2000	37.6	
	2.0	

Leuk.	PCA	RP
5	18.1	38.5
	2.3	5.7
10	20.4	35.4
	2.4	4.8
25	26.2	34.0
	1.7	4.0
50	37.2	34.2
	1.6	3.2
100	39.8	34.6
	1.6	3.1
200	39.8	36.4
	1.6	2.8
500	39.8	38.4
	1.6	2.2
3572	39.8	
	2.0	

Ion.	PCA	RP
5	120.8	174.2
	11.4	22.8
10	114.6	149.1
	13.8	17.9
15	112.2	134.0
	13.7	15.4
20	112.2	128.6
	11.3	11.4
25	112.7	127.2
	10.8	9.2
30	113.7	123.6
	11.0	8.6
34	114.2	
	2.5	

Spam	PCA	RP
5	979.9	2058.1
	47.2	229.7
10	892.0	1684.9
	69.4	228.2
15	862.5	1476.4
	70.7	177.4
20	845.6	1292.4
	71.9	145.5
25	834.5	1169.1
	81.0	125.5
30	835.8	1090.5
	85.5	111.7
57	802.0	
	21.0	

Table 4: Average number (and its standard deviation) of Support Vectors for each dataset and projection