

# Experiments with Random Projections for Machine Learning

Dmitriy Fradkin

Joint Work with David Madigan

# Purpose

To evaluate the effectiveness of Random Projections (RPs), compared to PCA, with different machine learning algorithms.

# Supervised Learning Problem

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

$$T = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}.$$

In what follows the inputs are vectors  $\mathbf{x}_i = [x_{i_1}, \dots, x_{i_p}]$  in  $\mathbb{R}^p$ ,  $y \in \{-1, 1\}$ , and we refer to  $f$  as a classifier. The objective is to produce a classifier that makes accurate predictions for future input vectors.

# The Need for Dimensionality Reduction

Data with large dimensionality ( $p$ ) presents problems for many machine learning algorithms:

- their computational complexity can be superlinear in  $p$
- they may need complexity control to avoid overfitting

Traditional methods such as PCA/SVD are computationally expensive:

- PCA is  $O(p^2n) + O(p^3)$  [Golub and van Loan, 1983]
- SVD is somewhat more efficient: for sparse matrices with  $r$  non-zero entries per column there are  $O(prn)$  algorithms [Papadimitriou et. al. 1998]

# Johnson-Lindenstrauss Theorem

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  $1 \pm \epsilon$  [Johnson and Lindenstrauss, 1984].

# “Database-Friendly Random Projections”

**Theorem 1** [*Achlioptas, 2001*] Given  $n$  points in  $\mathbb{R}^p$  (in form of an  $n \times p$  matrix  $X$ ), choose  $\epsilon, \beta > 0$  and  $q \geq \frac{4+2*\beta}{\frac{\epsilon^2}{2} - \frac{\epsilon^3}{3}} \log(n)$ , 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)

# Time Complexity of Random Projections

The above projections are easy to implement and to compute.

Constructing a  $p \times q$  random matrix is  $O(pq)$ . Performing the projection for  $n$  points is  $O(npq)$ .

# Theoretical Effectiveness

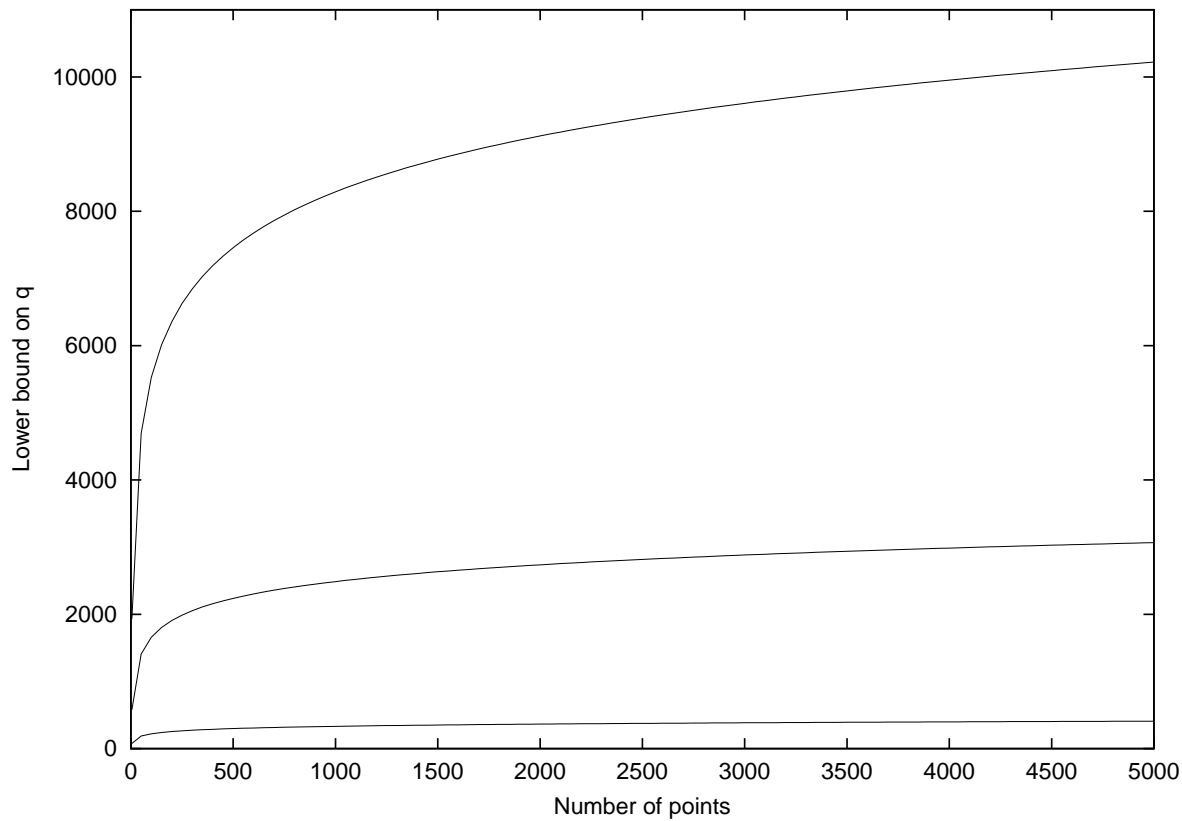


Figure 1: Plot of lower bound  $q$  of dimensionality of random projections as a function of number of points. Upper curve corresponds to  $\epsilon = 0.1$ , middle one - to  $\epsilon = 0.2$ , lowest one to  $\epsilon = 0.5$ .  $\beta = 1$  for all of these, allowing a deviation by a factor greater than  $\epsilon$  with probability  $\frac{1}{n}$ .

# Previous Experiments

[Bingham and Mannila, 2001] experimentally show that RP preserve similarity (inner products) well even when dimensionality of projection is moderate. They also compare performance of RP to PCA, SVD and DCT.

Their data had  $p = 5000$ ,  $n = 2262$  for text data, and  $p = 2500$ ,  $n = 1000$  for image data. Projections were done to  $q \in [1, 800]$ .

# Other Work with Random Projections

- Theoretical Approximate Nearest Neighbor algorithm with polynomial preprocessing and query time polynomial in  $p$  and  $\log n$  [Indyk and Motwani, 1998]. Also, the first tight bounds on the quality of randomized dimensionality reduction.
- Learning mixtures of Gaussians in high dimensions [Dasgupta 1999], [Dasgupta, 2000]. Combination of RP with EM algorithm gives good classification results on a hand-written digit dataset.
- Preservation of volumes and affine distances [Magen 2002].
- Deterministic algorithm for constructing JL mappings [Engebretsen, Indyk and O'Donnell 2002], used to derandomize several randomized algorithms.
- Approximate kernel computations [Achlioptas, McSherry and Schölkopf, 2001], similarity computations for histogram models [Thaper et. al 2002].

# Our Implementation of Random Projections

We chose to implement the first of the methods suggested by Achlioptas:

- $r_{ij} = \pm 1$  with probability 0.5 each

Since we are not concerned with preserving distances per se, but only with preserving separation between points, we do not scale our projection:  $E = XP$  instead of  $E = \frac{1}{\sqrt{q}}XP$

# Description of Data

Ionosphere, Spambase and Internet Ads were taken from UCI repository. Colon and Leukemia were first used in [Alon et. al 1999] and [Golub et. al. 1999] respectively.

Table 1:

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

# Choice of Projection Dimensions

- 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.

# Experimental Setup

We compare PCA and RP using a number of standard machine learning tools:

- decision trees (C4.5 - [Quinlan, 1993])
- linear SVM (SVMLight - [Joachims, 1999])
- nearest neighbor (NN)

Test set sizes were kept constant over different splits: Ionosphere - 51, Spambase - 1601, Colon - 12, Leukemia - 12, Ads - 1079.

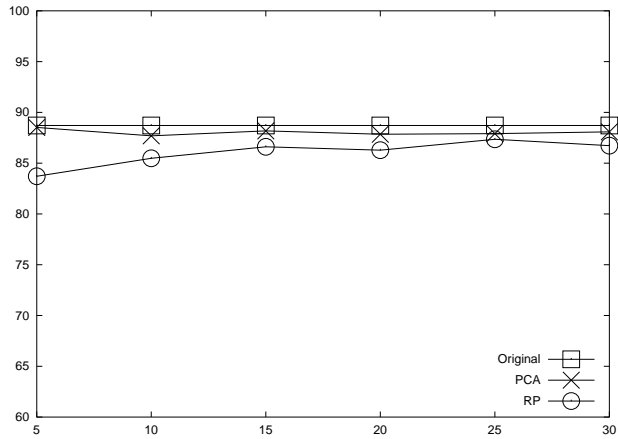
# Experimental Procedure

**Require:** Dataset  $D$ , set of projection dimensions  $\{d_1, \dots, d_k\}$ , number of test/training splits  $s$  to be done (we perform 30 splits for Ads and 100 splits for other datasets)

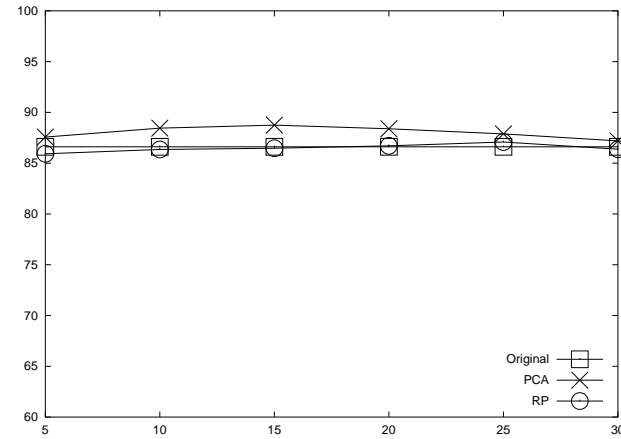
- 1: **for**  $i = 1, \dots, s$  **do**
- 2:   split  $D$  into training set and test set
- 3:   normalize the data (estimating mean and variance from the training set)
- 4:   **for**  $d' = d_1, \dots, d_k$  **do**
- 5:     do a PCA on training set and project both training and test data into  $\mathbb{R}^{d'}$
- 6:     create a random projection matrix as described above and project both training and test data into  $\mathbb{R}^{d'}$
- 7:     train learning methods on training sets and then apply them to respective test sets to evaluate performance
- 8:   **end for**
- 9: **end for**

# Results on Ion Dataset

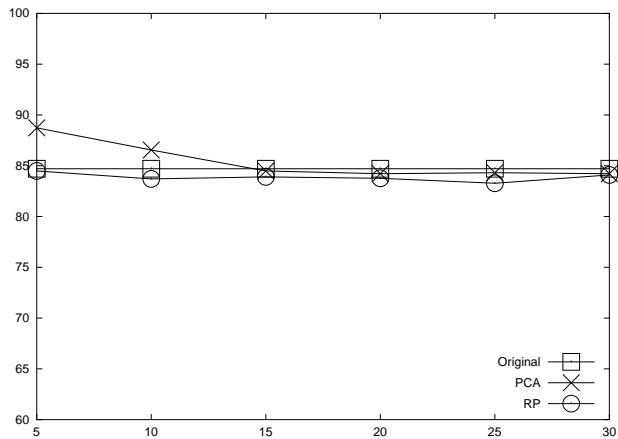
## C4.5



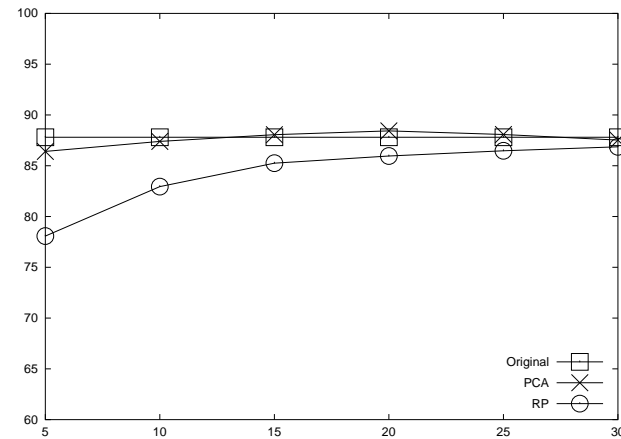
## 1NN



## 5NN

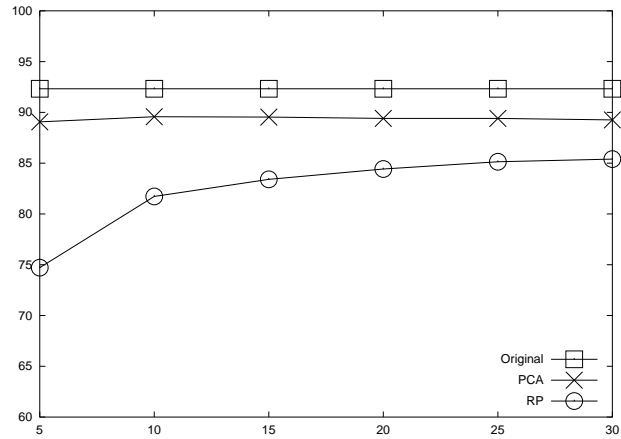


## SVM

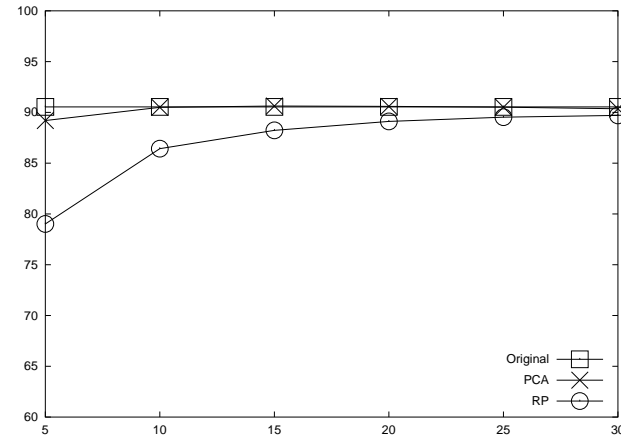


# Results on Spam Dataset

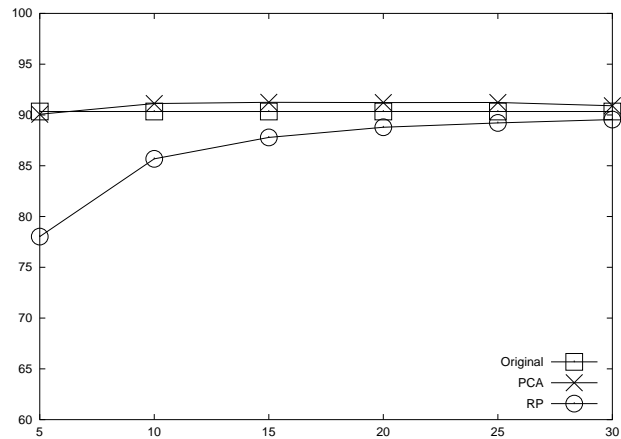
## C4.5



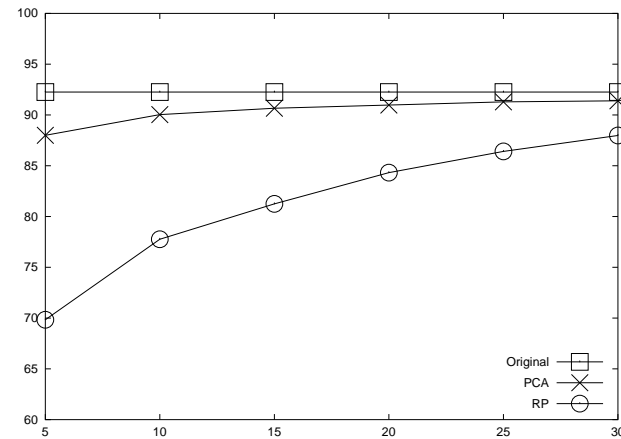
## 1NN



## 5NN

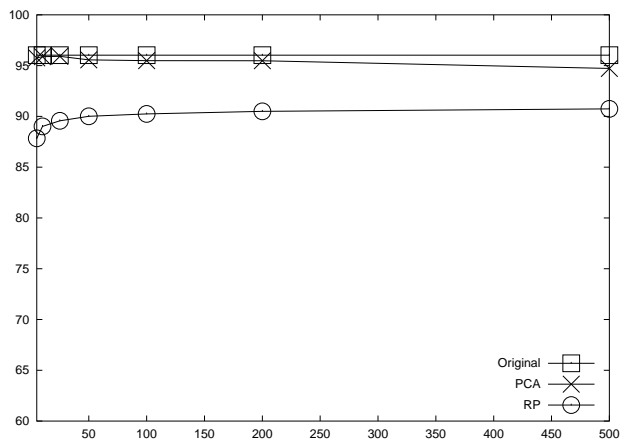


## SVM

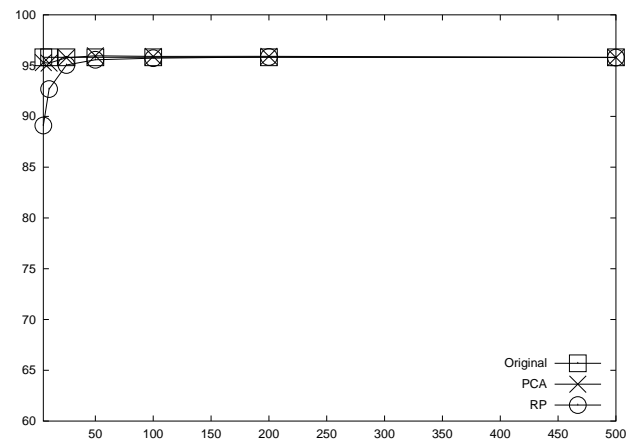


# Results on Ads Dataset

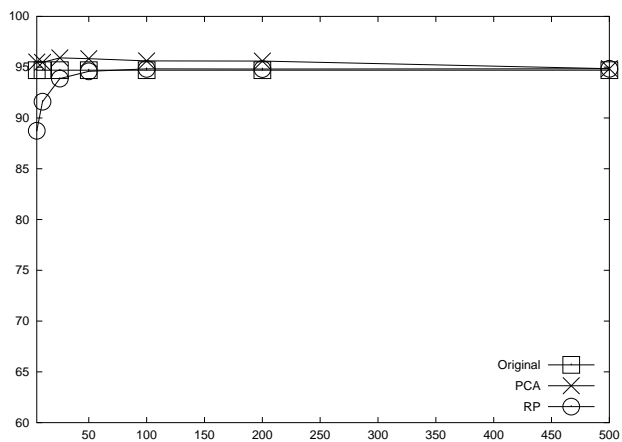
## C4.5



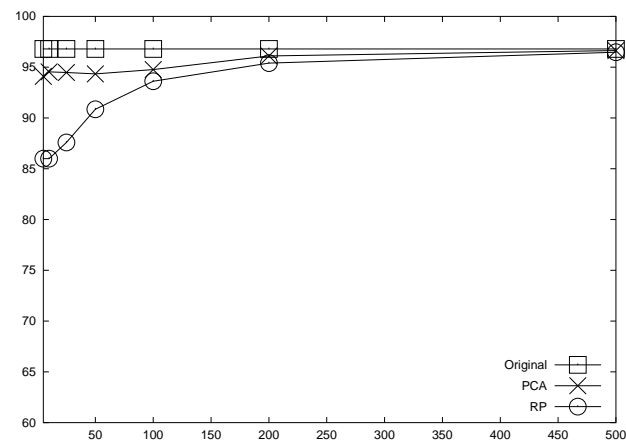
## 1NN



## 5NN

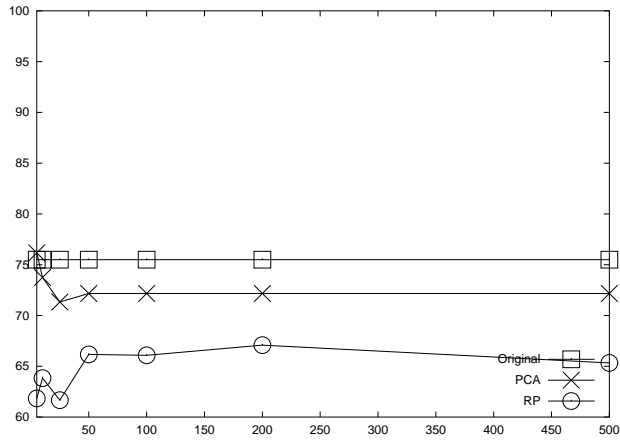


## SVM

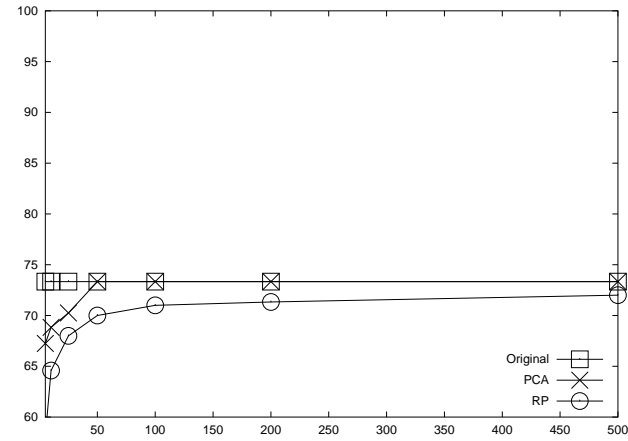


# Results on Colon Dataset

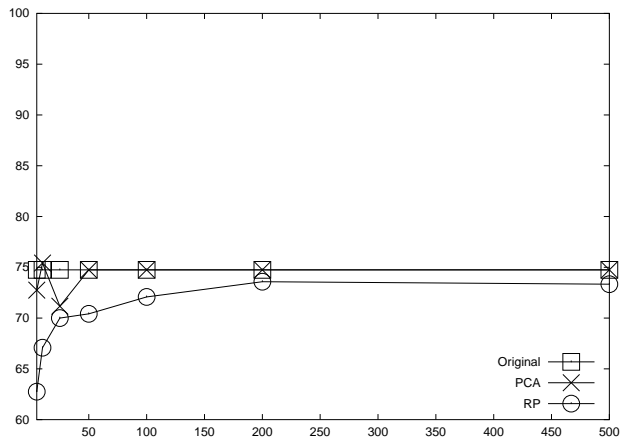
## C4.5



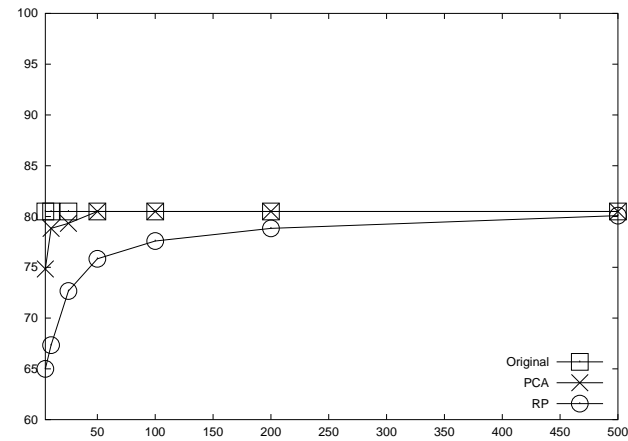
## 1NN



## 5NN

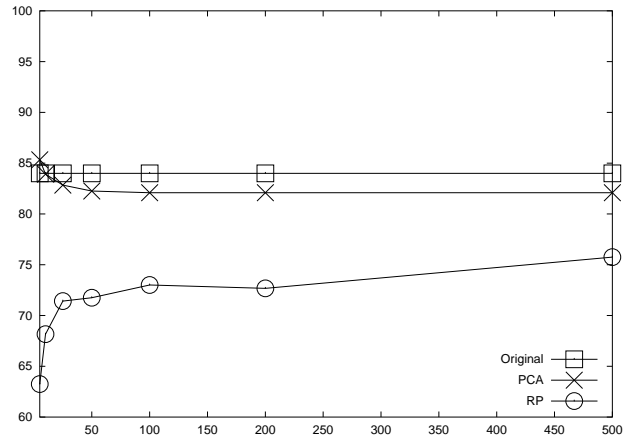


## SVM

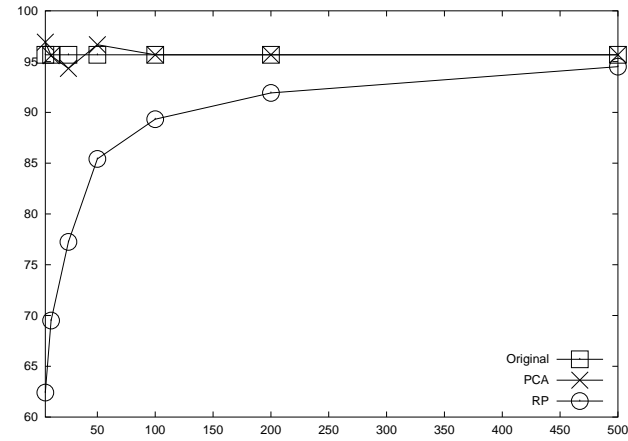


# Results on Leukemia Dataset

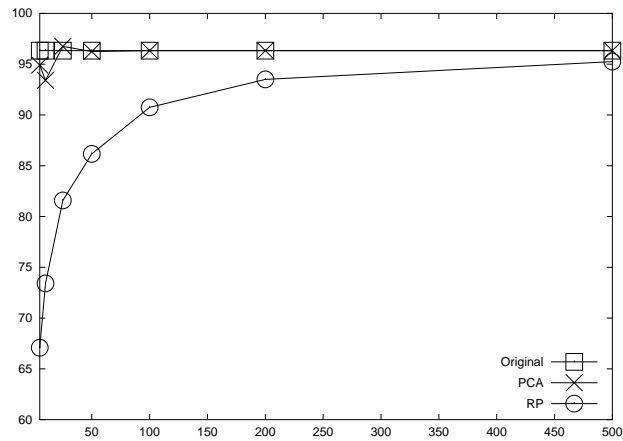
## C4.5



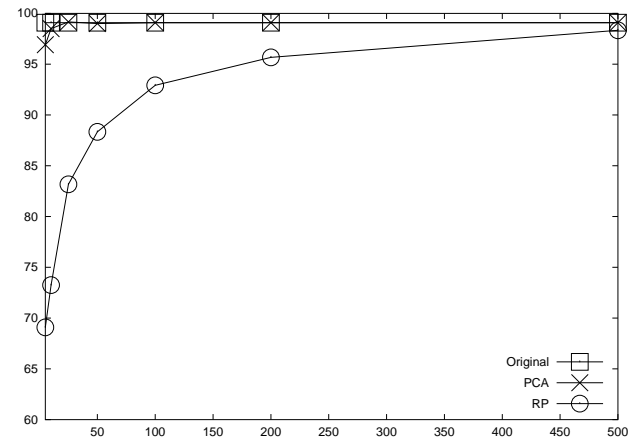
## 1NN



## 5NN



## SVM



# Discussion of C4.5 performance

- C4.5 does well with low-dimensional PCA projections (on Ionosphere, Colon and Leukemia datasets), but its performance deteriorates after that and doesn't improve.
- Performance with RP is poor: after some initial improvement the accuracy curve seems to level out.

Decision trees rely on informativeness of individual attributes and construct axis-parallel boundaries for their decisions. They don't deal well with transformations of the attributes, and are sensitive to noise. Random Projections and decision trees are perhaps not a good combination.

# Discussion of NN performance

- Nearest Neighbor Methods appear to be least affected by reduction in dimensionality through PCA or RP.
- 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 can be explained by their exclusive reliance on distance computations.

# Discussion of SVM performance

- SVM does worse in projection spaces (both with PCA and RP) than in the original space.
- Its performance improves noticeably as the dimensionality of projections increases.
- Performance of PCA is much better initially, but RPs are catching up to it.

# Discussion of data complexity

We kept track of the number of support vectors used in each projection:

- 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.
- RPs resulted in about the same number of support vectors on Colon and Leukemia Datasets, but much higher numbers on Ads, Spam and Ionosphere.
- For both 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.
- The number of support vectors when using PCA was always less than when using RP in lower dimensions.

# Conclusions

- RPs performance was (predictably) below the level of PCA.
- RPs performance was improving noticeably with increasing dimensionality.
- RPs seem well suited for use with Nearest Neighbor methods.
- Decision tree did not combine with RP in a satisfactory way.

# Directions for Further Study

- Explore performance on significantly larger datasets
- Ensembles of classifiers trained on different projections
  - different projections to the same dimension
  - projections to different dimensions

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