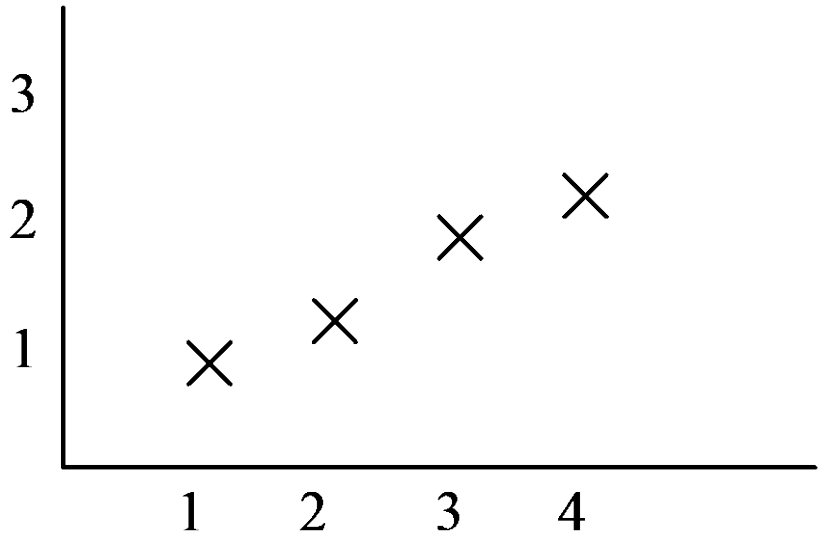
Dimensionality reduction

Usman Roshan

Dimensionality reduction

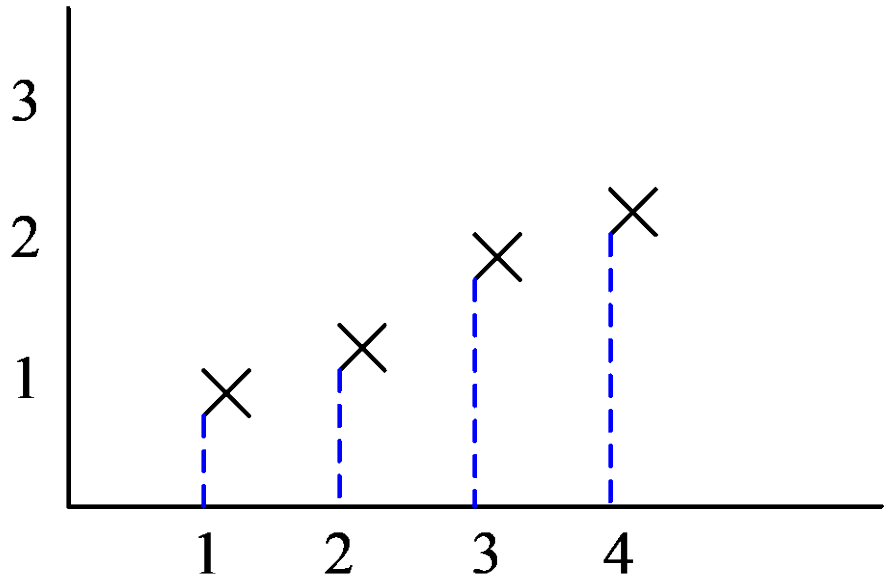
- What is dimensionality reduction?
  - Compress high dimensional data into lower dimensions
- How do we achieve this?
  - PCA (unsupervised): We find a vector w of length 1 such that the variance of the projected data onto w is maximized.
  - Binary classification (supervised): Find a vector w that maximizes ratio (Fisher) or difference (MMC) of means and variances of the two classes.

Data projection



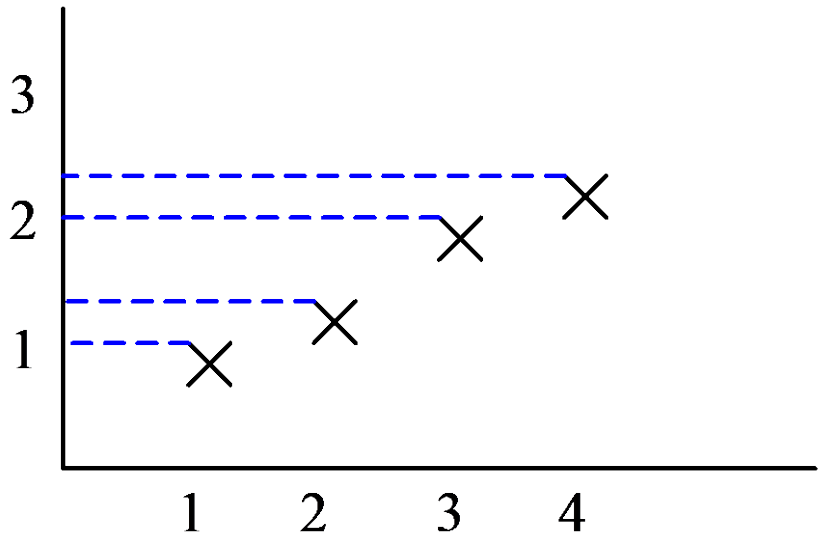
Data projection

- Projection on x-axis



Data projection

- Projection on y-axis



Mean and variance of data

- Original data

$$Mean: m = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$Variance = \frac{1}{n}\sum_{i=1}^{n} (x_i - m)^2$$

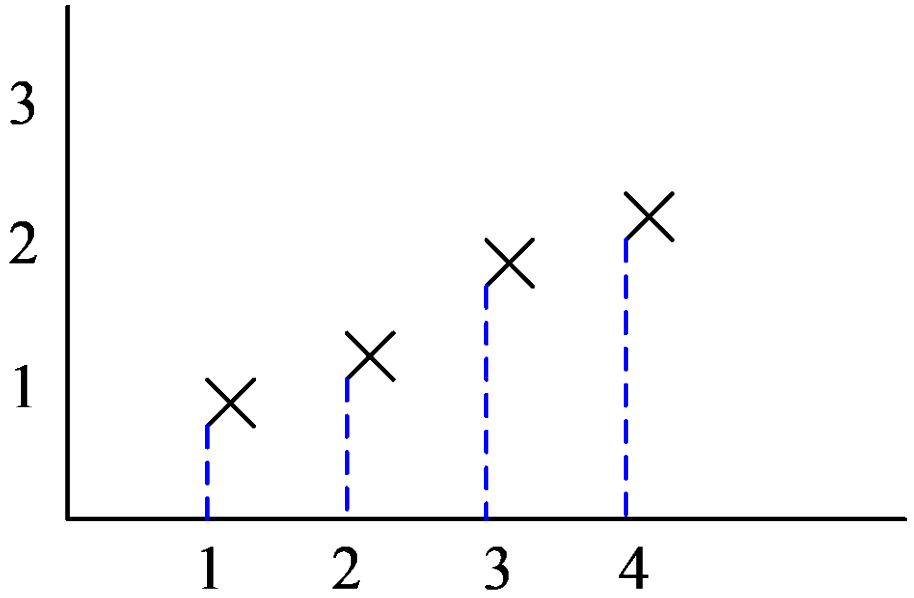
Projected data

$$Mean: m' = \frac{1}{n}\sum_{i=1}^{n} w^T x_i = w^T m$$

$$Variance = \frac{1}{n}\sum_{i=1}^{n} (w^T x_i - w^T m)^2$$

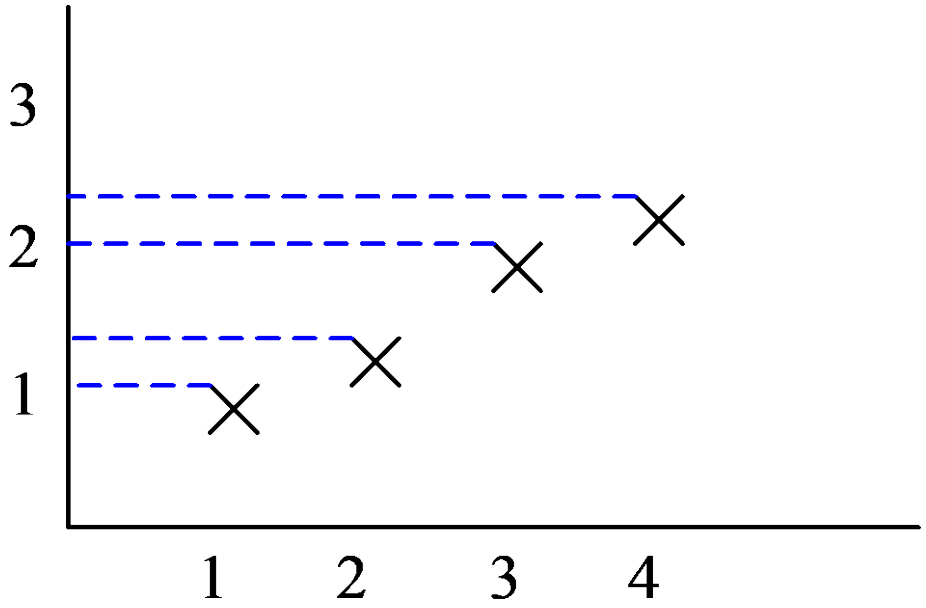
Data projection

- What is the mean and variance of projected data?



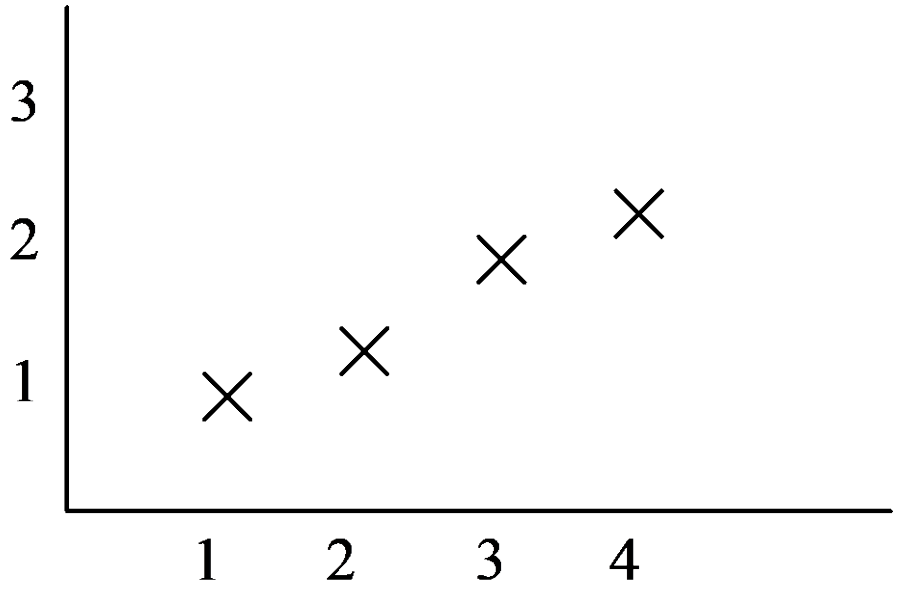
Data projection

- What is the mean and variance here?



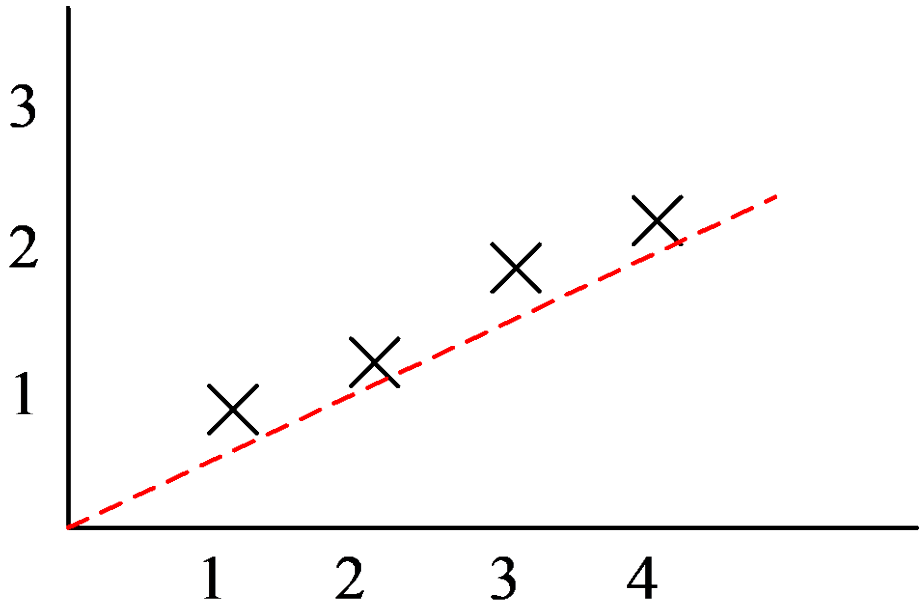
Data projection

- Which line maximizes variance?



Data projection

- Which line maximizes variance?



Principal component analysis

- Find vector w of length 1 that maximizes variance of projected data

PCA optimization problem

$$\arg\max_{w} \frac{1}{n}\sum_{i=1}^{n}(w^T x_i - w^T m)^2 \text{ subject to } w^T w = 1$$

The optimization criterion can be rewritten as

$$\arg\max_{w} \frac{1}{n}\sum_{i=1}^{n}(w^T (x_i - m))^2 =$$

$$\arg\max_{w} \frac{1}{n}\sum_{i=1}^{n}(w^T (x_i - m))^T (w^T (x_i - m)) =$$

$$\arg\max_{w} \frac{1}{n}\sum_{i=1}^{n}((x_i - m)^T w)(w^T (x_i - m)) =$$

$$\arg\max_{w} \frac{1}{n}\sum_{i=1}^{n} w^T (x_i - m)(x_i - m)^T w =$$

$$\arg\max_{w} w^T \frac{1}{n}\sum_{i=1}^{n}(x_i - m)(x_i - m)^T w =$$

$$\arg\max_{w} w^T \sum w \text{ subject to } w^T w = 1$$

PCA optimization problem

$$\Sigma = \frac{1}{n}\sum_{i=1}^{n}(x_i - m)(x_i - m)^T$$

is also called the scatter matrix

If we let $X = [x_1 - m, x_2 - m, \square \ , x_n - m]$

where each x_i is a column vector then

$$\Sigma = XX^T$$

PCA solution

- Using Lagrange multipliers we can show that w is given by the largest eigenvector of $\sum$.
- With this we can compress all the vectors x_i into $w^T x_i$
- Does this help? Before looking at examples, what if we want to compute a second projection $u^T x_i$ such that $w^T u=0$ and $u^T u=1$?
- It turns out that u is given by the second largest eigenvector of $\sum$.

PCA space and runtime considerations

- Depends on eigenvector computation
- BLAS and LAPACK subroutines
  - Provides Basic Linear Algebra Subroutines.
  - Fast C and FORTRAN implementations.
  - Foundation for linear algebra routines in most contemporary software and programming languages.
  - Different subroutines for eigenvector computation available

PCA space and runtime considerations

- Eigenvector computation requires quadratic space in number of columns
- Poses a problem for high dimensional data
- Instead we can use the Singular Value Decomposition

PCA via SVD

- Every n by n symmetric matrix Σ has an eigenvector decomposition $\Sigma=QDQ^T$ where D is a diagonal matrix containing eigenvalues of Σ and the columns of Q are the eigenvectors of Σ.
- Every m by n matrix A has a singular value decomposition $A=USV^T$ where S is m by n matrix containing singular values of A, U is m by m containing left singular vectors (as columns), and V is n by n containing right singular vectors. Singular vectors are of length 1 and orthogonal to each other.

PCA via SVD

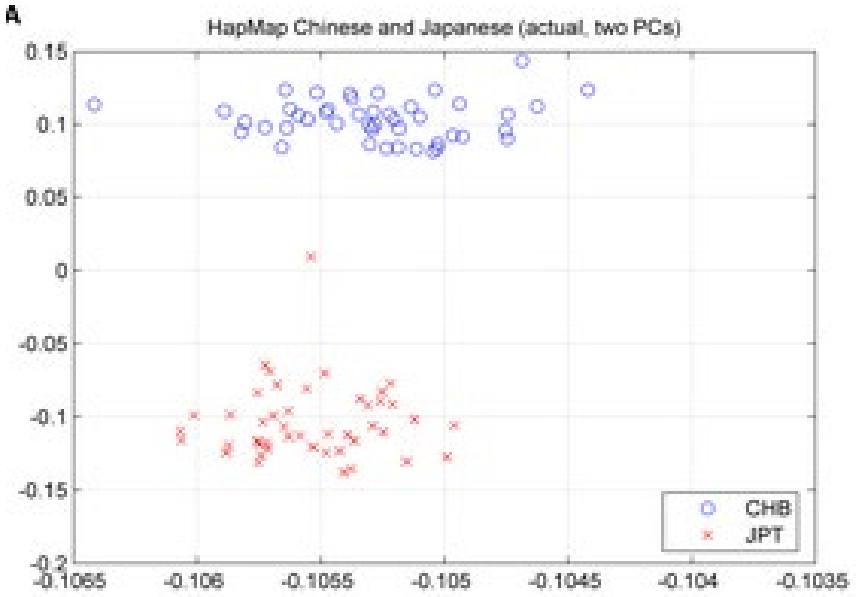
- In PCA the matrix $\Sigma = XX^T$ is symmetric and so the eigenvectors are given by columns of Q in $\Sigma = QDQ^T$.
- The data matrix X (mean subtracted) has the singular value decomposition $X = USV^T$.
- This gives
  - $\Sigma = XX^T = USV^T(USV^T)^T$
  - $USV^T(USV^T)^T = USV^TVSU^T$
  - $USV^TVSU^T = US^2U^T$
- Thus $\Sigma = XX^T = US^2U^T \Rightarrow XX^TU = US^2U^TU = US^2$
- This means the eigenvectors of Σ (principal components of X) are the columns of U and the eigenvalues are the diagonal entries of S^2.

PCA via SVD

- And so an alternative way to compute PCA is to find the left singular values of X.
- If we want just the first few principal components (instead of all cols) we can implement PCA in rows x cols space with BLAS and LAPACK libraries
- Useful when dimensionality is very high at least in the order of 100s of thousands.

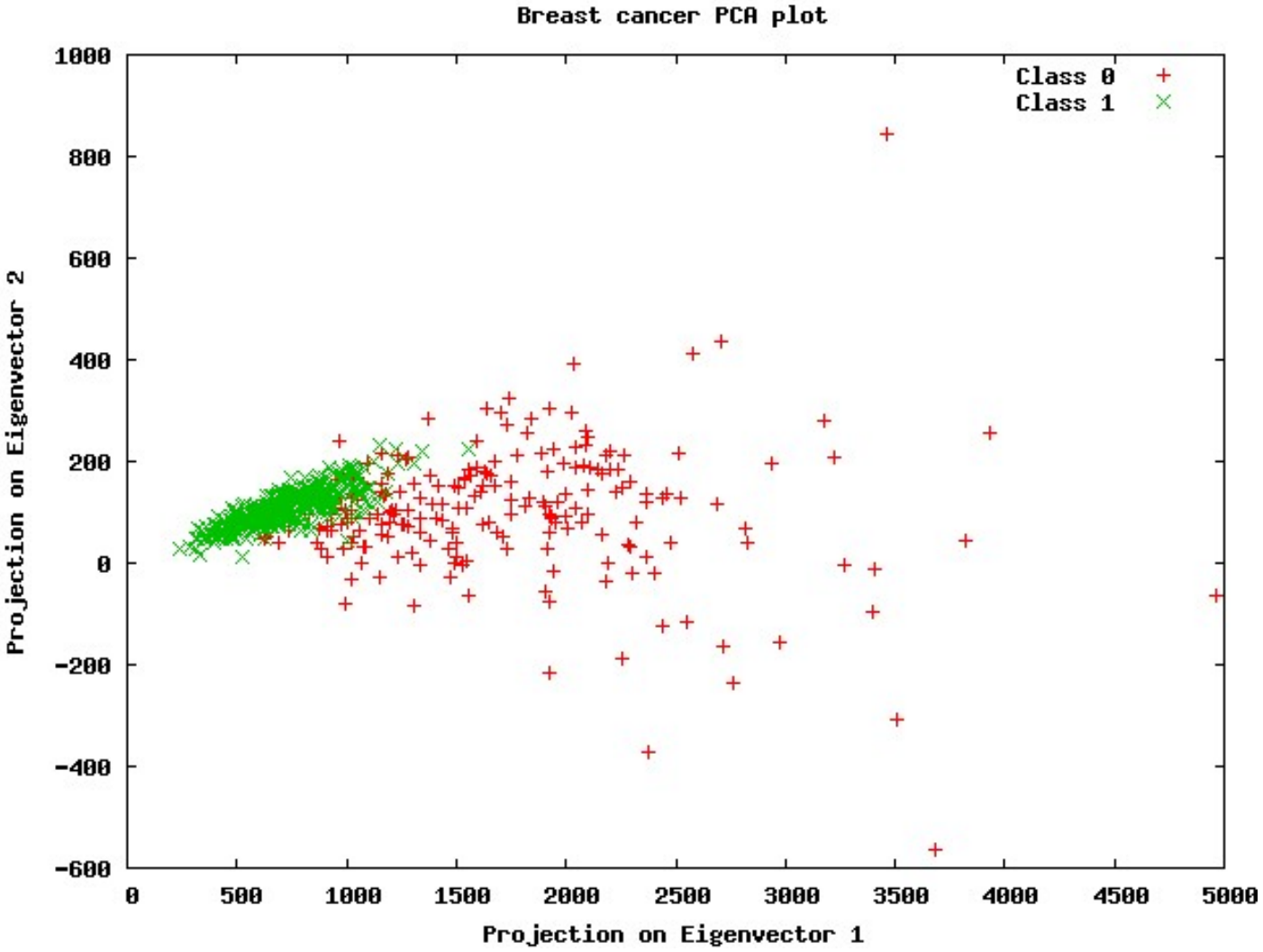
PCA on genomic population data

- 45 Japanese and 45 Han Chinese from the International HapMap Project
- PCA applied on 1.7 million SNPs

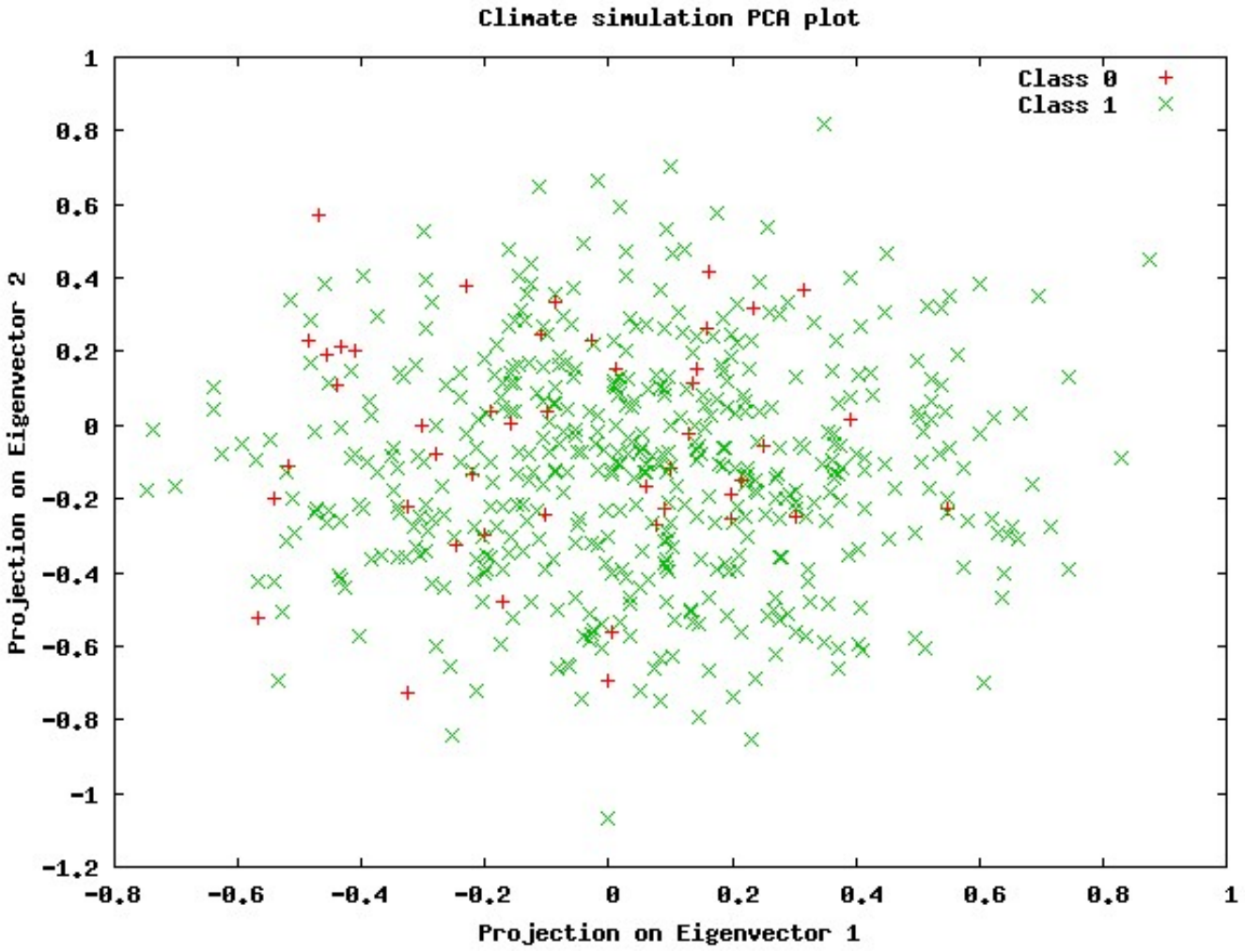


Taken from "PCA-Correlated SNPs for Structure Identification in Worldwide Human Populations" by Paschou et. al. in PLoS Genetics 2007

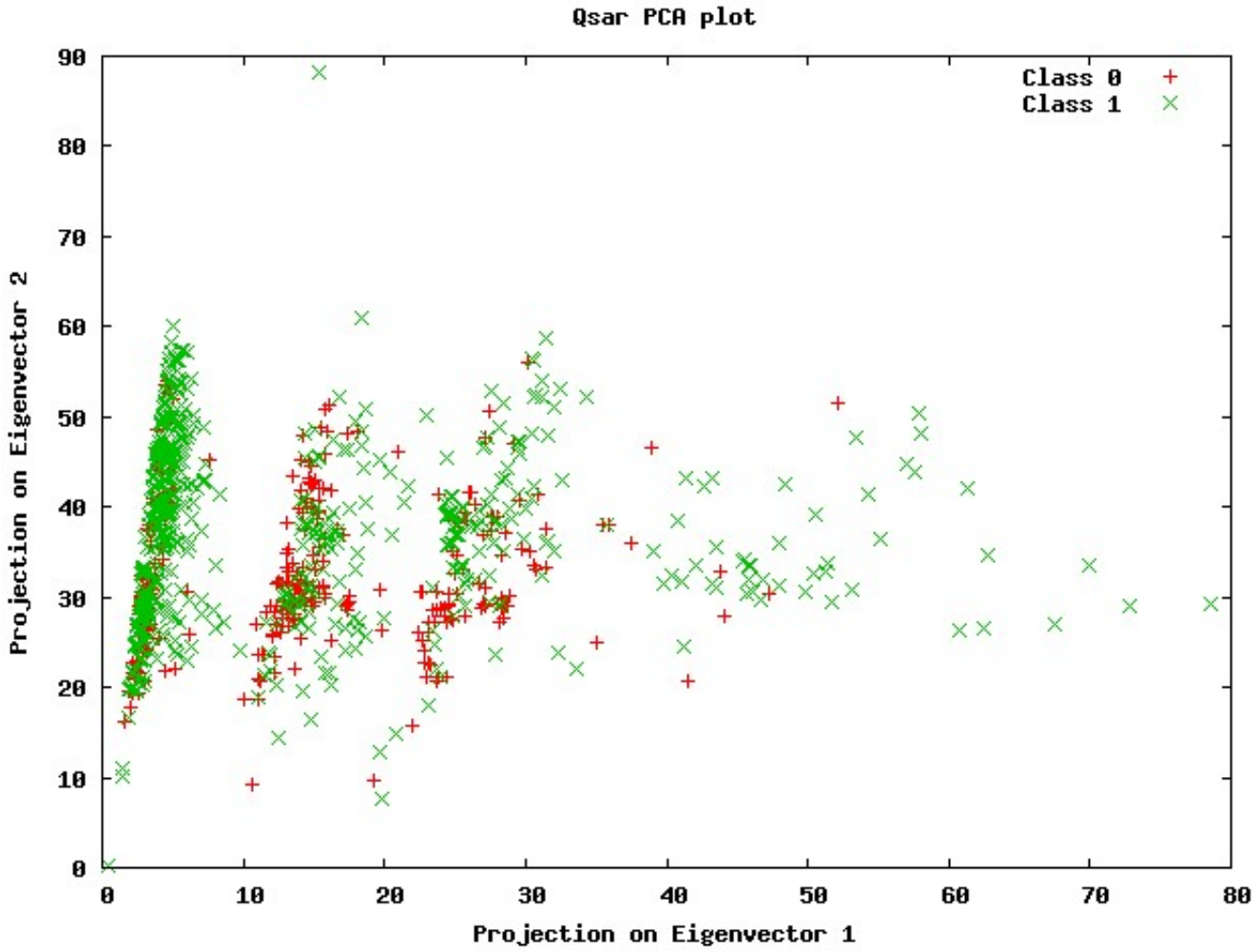
PCA on breast cancer data



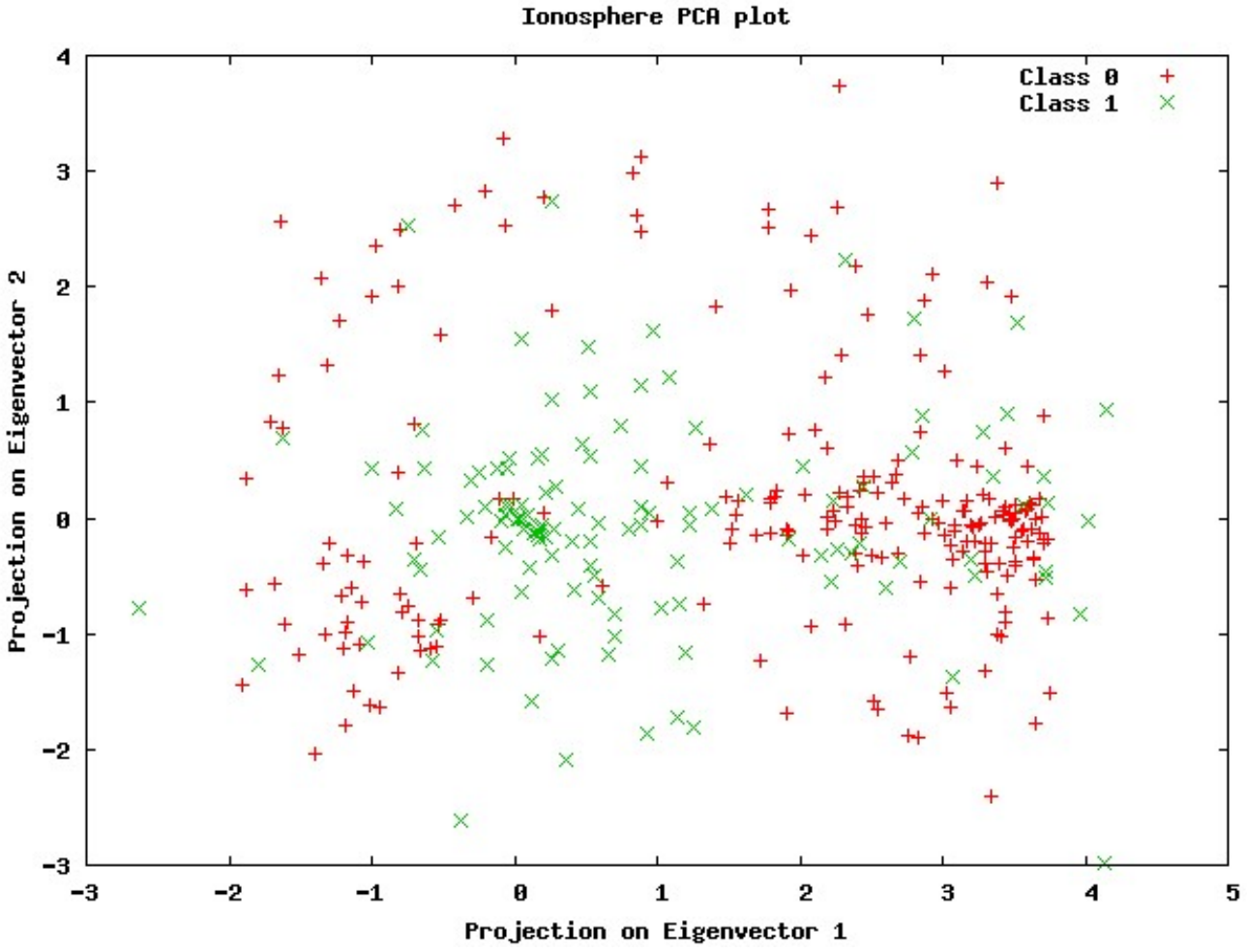
PCA on climate simulation



PCA on QSAR


Qsar PCA plot
Class 0
Class 1
Projection on Eigenvector 2
Projection on Eigenvector 1
0
10
20
30
40
50
60
70
80
90

PCA on Ionosphere



Kernel PCA

- Main idea of kernel version
  - $XX^Tw = \lambda w$
  - $X^TXX^Tw = \lambda X^Tw$
  - $(X^TX)X^Tw = \lambda X^Tw$
  - X^Tw is projection of data on the eigenvector w and also the eigenvector of X^TX
- This is also another way to compute projections in space quadratic in number of rows but only gives projections.

Kernel PCA

- In feature space the mean is given by

$$m_\Phi = \frac{1}{n}\sum_{i=1}^{n}\Phi(x_i)$$

- Suppose for a moment that the data is mean subtracted in feature space. In other words mean is 0. Then the scatter matrix in feature space is given by

$$\Sigma_\Phi = \frac{1}{n}\sum_{i=1}^{n}\Phi(x_i)\Phi^T(x_i)$$

Kernel PCA

- The eigenvectors of Σ_Φ give us the PCA solution. But what if we only know the kernel matrix?
- First we center the kernel matrix so that mean is 0

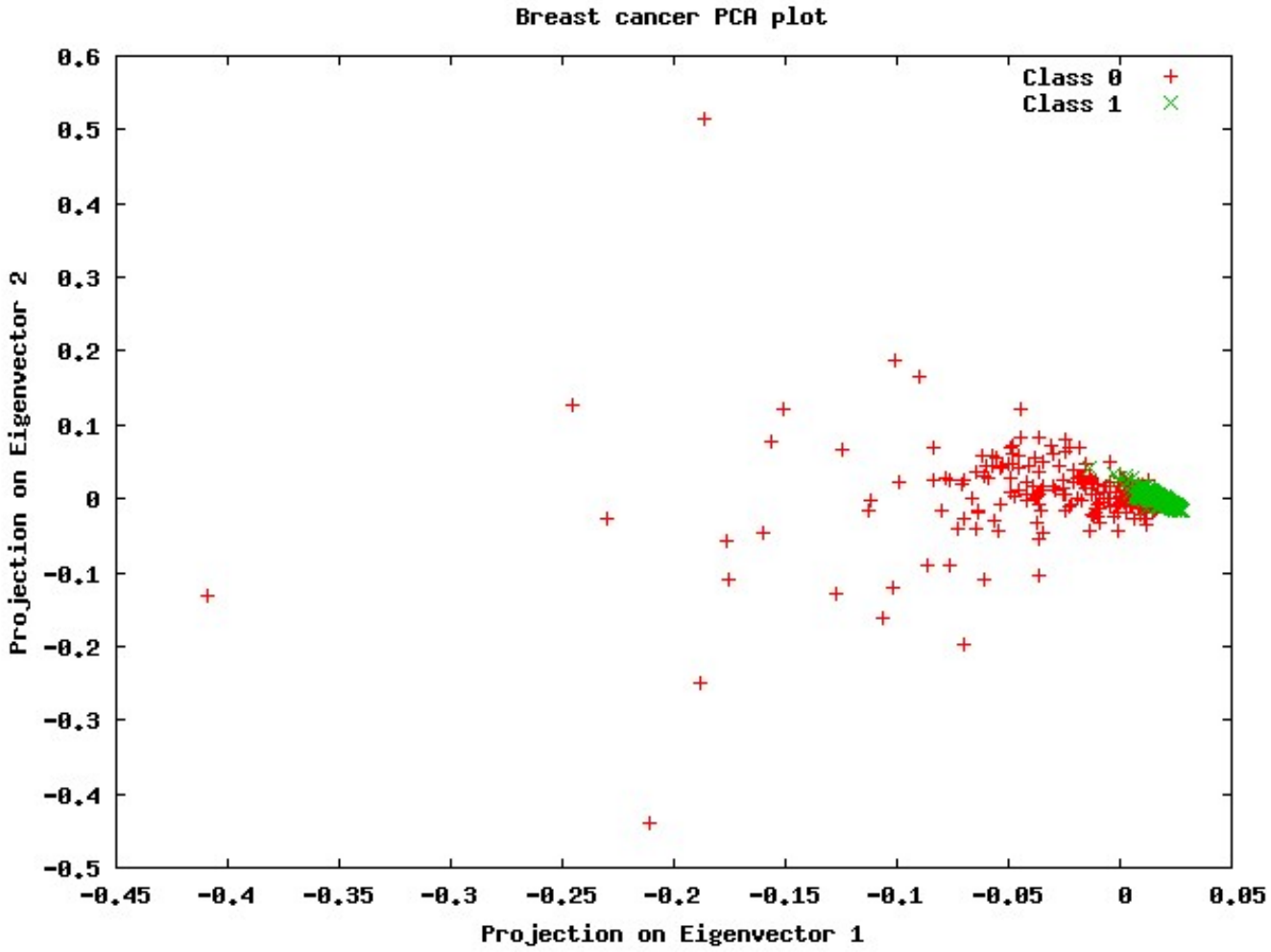
$$\hat{\mathbf{K}} = \mathbf{K} - \frac{1}{\ell}\mathbf{j}\mathbf{j}'\mathbf{K} - \frac{1}{\ell}\mathbf{K}\mathbf{j}\mathbf{j}' + \frac{1}{\ell^2}\left(\mathbf{j}'\mathbf{K}\mathbf{j}\right)\mathbf{j}\mathbf{j}'$$

where j is a vector of 1's.K = K

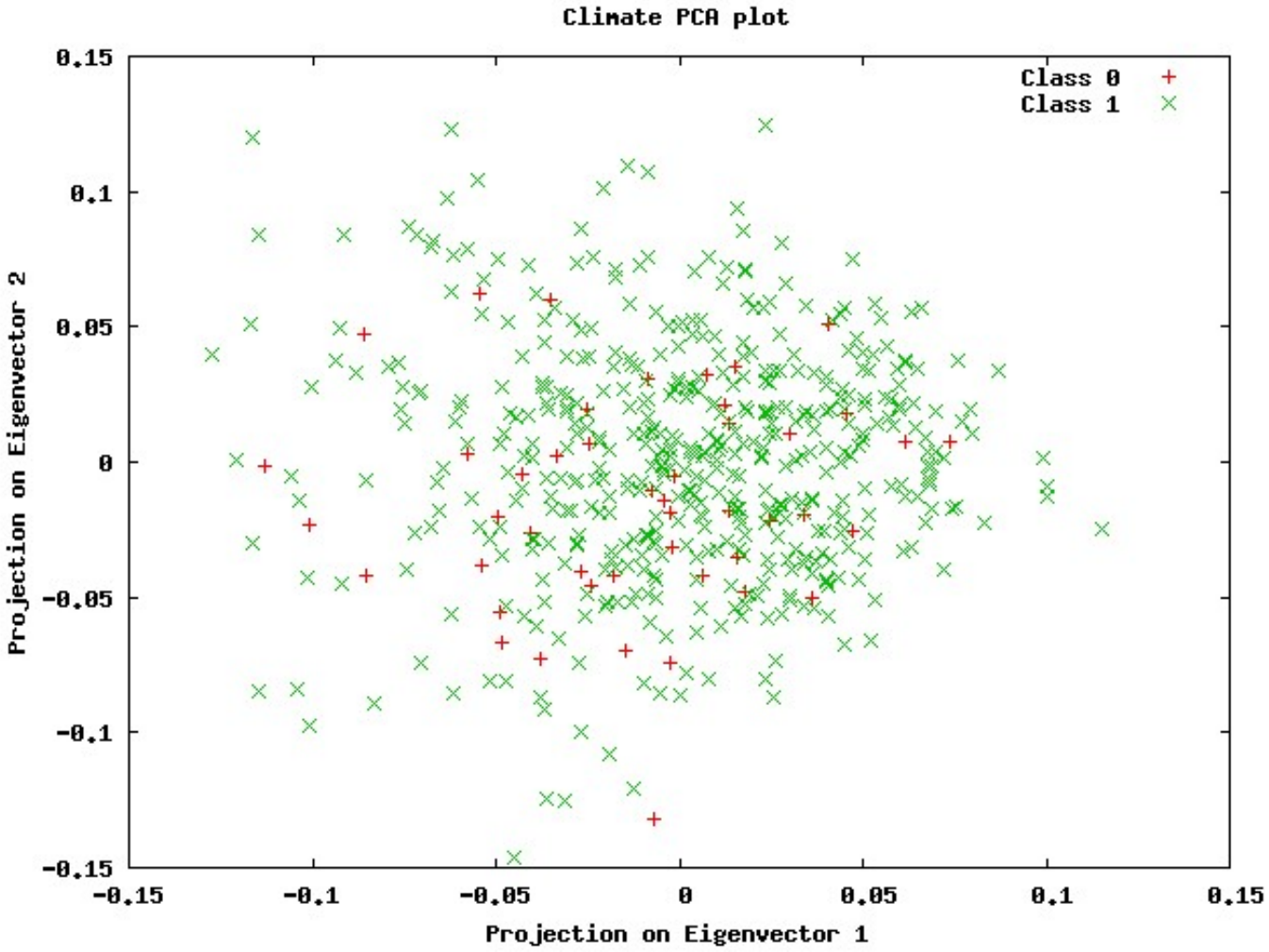
Kernel PCA

- Recall from earlier
  - $XX^Tw = \lambda w$
  - $X^TXX^Tw = \lambda X^Tw$
  - $(X^TX)X^Tw = \lambda X^Tw$
  - X^Tw is projection of data on the eigenvector w and also the eigenvector of X^TX
  - X^TX is the linear kernel matrix
- Same idea for kernel PCA
- The projected solution is given by the eigenvectors of the centered kernel matrix.

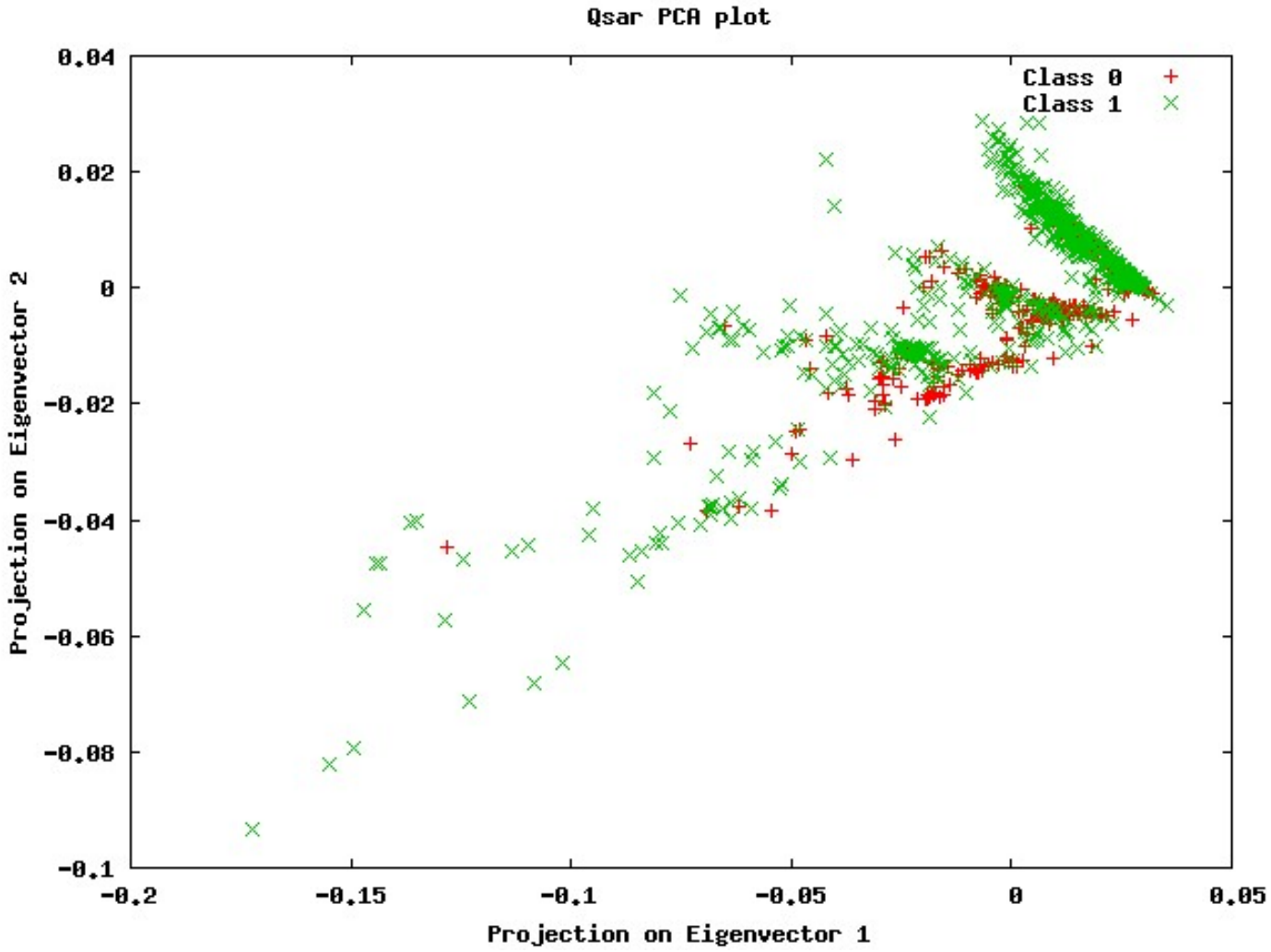
Polynomial degree 2 kernel Breast cancer



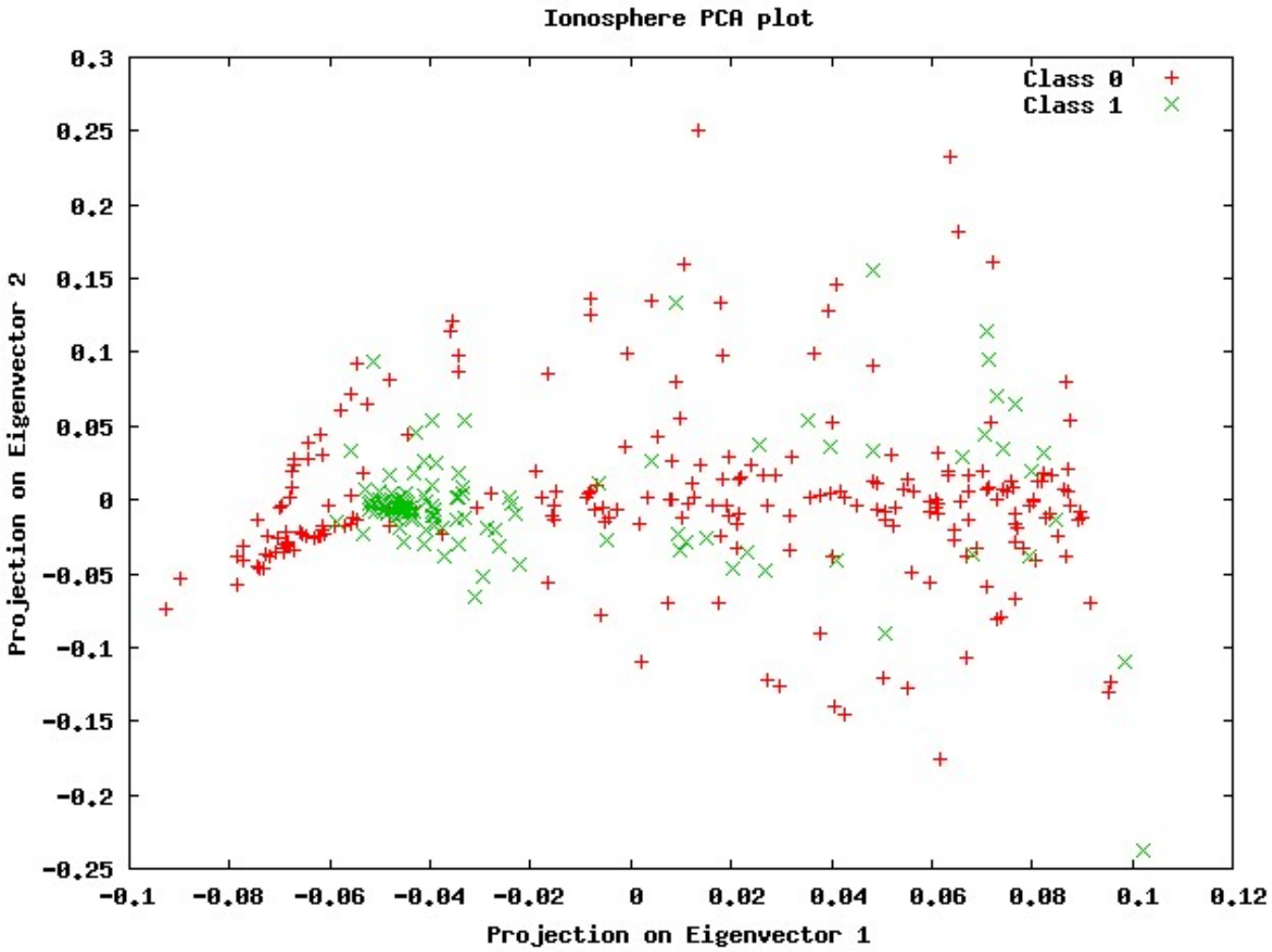
Polynomial degree 2 kernel Climate



Polynomial degree 2 kernel Qsar



Polynomial degree 2 kernel Ionosphere



Random projections

- What if we projected our data onto random vectors instead of PCA or LDA?
- Turns out that random projections preserve distances upto a certain error

Johnson-Lindenstrauss lemma

- Given any ε and n and k >= O(log(n)/ε^2), for any set of P of n points in R^d there exists a lower dimensional mapping f(x) (x in P) to R^k such that for any u,v in P

$$(1-\varepsilon)\left\|u-v\right\|^2 \le \left\|f(u)-f(v)\right\|^2 \le (1+\varepsilon)\left\|u-v\right\|^2$$

- Furthermore, this mapping can be found in randomized polynomial time. Simply let each random vector be randomly sampled from thenormal Gaussian distribution.
- Why does this work? Because random projections of vectors preserve length and we model distance between vectors u and v as vectors.