Approximated PCA

Rodrigo Arias

February 16, 2017

PCA algorithm

1. Take a dataset X.
2. Scale and center the variables.
3. From X compute the covariance matrix S.
4. **Compute the eigenvalues and eigenvectors of S.**
5. *Optional: Ignore some eigenvectors.*
6. Generate a new basis from the selected eigenvectors.
7. Project X into the new basis.

PCA algorithm

1. Take a dataset X.
2. Scale and center the variables.
3. From X compute the covariance matrix S.
4. **Compute the eigenvalues and eigenvectors of S.**
5. *Optional: Ignore some eigenvectors.*
6. Generate a new basis from the selected eigenvectors.
7. Project X into the new basis.

Computing the eigenvectors is the principal step of PCA.

Visual example

The input dataset with 800 variables $X =$

Visual example

Keep 200 eigenvectors. Output:

Visual example

Keep 100 eigenvectors. Output:

Visual example

Keep 40 eigenvectors. Output:

Computing eigenvalues and eigenvectors

1. Take the target matrix A.
2. Compute a tridiagonal matrix T, $A = PTP^T$.
3. From T compute a diagonal matrix D, $T = QTQ^T$.
4. The eigenvalues of A are in the diagonal of D.
5. Compute the eigenvectors from D.

Computing eigenvalues and eigenvectors

1. Take the target matrix A.
2. Compute a tridiagonal matrix T, $A = PTP^T$.
3. From T compute a diagonal matrix D, $T = QTQ^T$.
4. The eigenvalues of A are in the diagonal of D.
5. Compute the eigenvectors from D.

Computing a tridiagonal matrix is called **tridiagonalization**. For the diagonal, **diagonalization**. Several algorithms exists for both steps.

Tridiagonalization

These algorithms transform a **symmetric** matrix A into a new pair of matrices P and T such that P is orthogonal, T is tridiagonal, and $A = PTP^T$

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix} = Q \begin{pmatrix} t_{11} & t_{12} & & \\ t_{21} & t_{22} & t_{23} & \\ & t_{32} & t_{33} & t_{34} \\ & & t_{43} & t_{44} \end{pmatrix} Q^T$$

Tridiagonalization

Algorithm	Complexity	Iterative	Stability
Householder	$O(4n^3/3)$	No	Great
Givens	$O(kn^3)$	No	Good
Lanczos	$O(kpn^2)$	Yes	Bad
Others			

Where k is some constant, and p the number of iterations.

Diagonalization

These algorithms take a **tridiagonal** matrix T into a new pair of matrices Q and D such that Q is orthogonal, D is diagonal, and $T = QDQ^T$

$$\begin{pmatrix} t_{11} & t_{12} & & \\ t_{21} & t_{22} & t_{23} & \\ & t_{32} & t_{33} & t_{34} \\ & & t_{43} & t_{44} \end{pmatrix} = Q \begin{pmatrix} d_{11} & & & \\ & d_{22} & & \\ & & d_{33} & \\ & & & d_{44} \end{pmatrix} Q^T$$

The matrix D contains the **eigenvalues** in the diagonal.

Diagonalization

Algorithm	Complexity	Iterative	Convergence
QR	$O(6n^3)$	Yes	Cubic
Divide and conquer	$O(8n^3/3)$	Yes	Quadratic
Jacobi	$O(n^3)$	Yes	Quadratic
Power iteration	$O(n^3)$	Yes	Linear
Inverse iteration	$O(n^3)$	Yes	Linear
Others			

The selected method

Householder + QR