Linear Algebra review

Linear Transformations and Eigenvectors

Recall: Matrices can store data

variables

observations

$$\begin{array}{cc} & \begin{array}{ccc}\text{gene1} & \text{gene2} & \text{gene3}\end{array} \\ \begin{array}{c}\text{Cell 1}\\ \text{Cell2}\\ \\ \\ \\ \end{array} & \begin{bmatrix} \boxed{d11 \quad d12 \quad \cdot \quad \cdot \quad \cdot} \\ d21 \quad d22 \qquad\qquad\quad \\ \cdot \qquad\qquad \cdot \qquad\qquad \\ \cdot \qquad\qquad\qquad \cdot \qquad \\ \cdot \qquad\qquad\qquad\quad \cdot \end{bmatrix}\end{array}$$

Each row is a vector

Datapoints (vectors) are locations in coordinate space

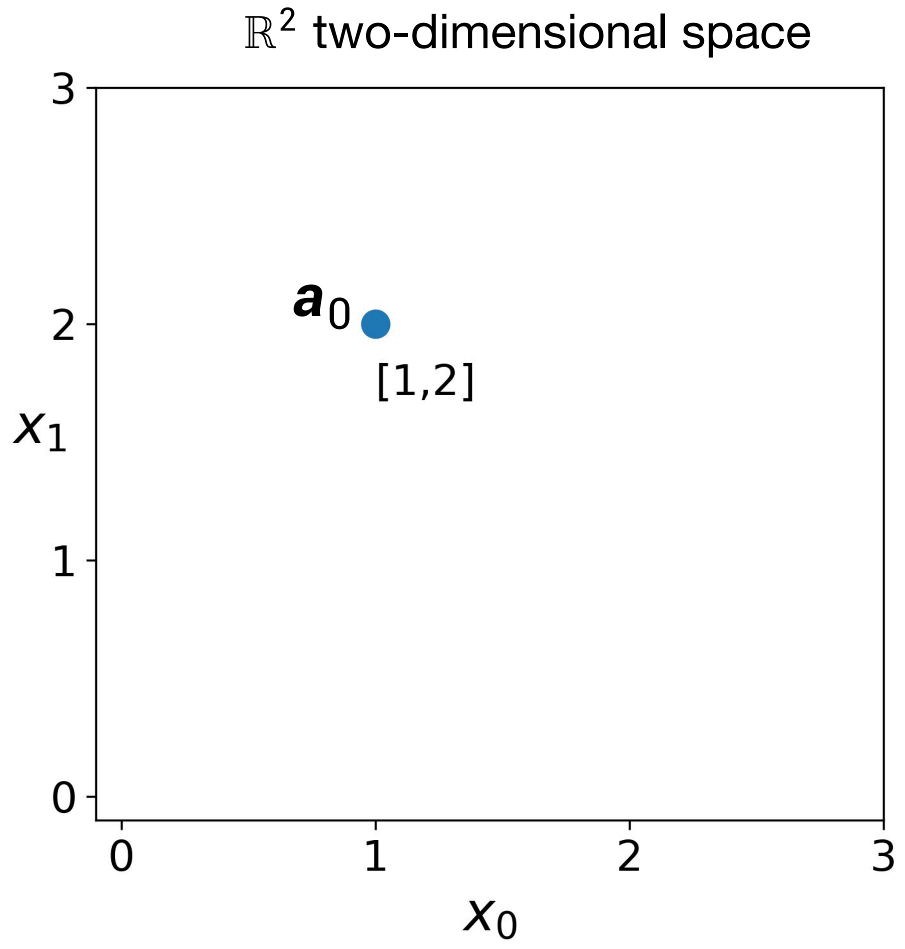
vector   coordinate space

$$\boldsymbol{a}_0 \in \mathbb{R}^2$$

$$\boldsymbol{a}_0 = [1,2]$$

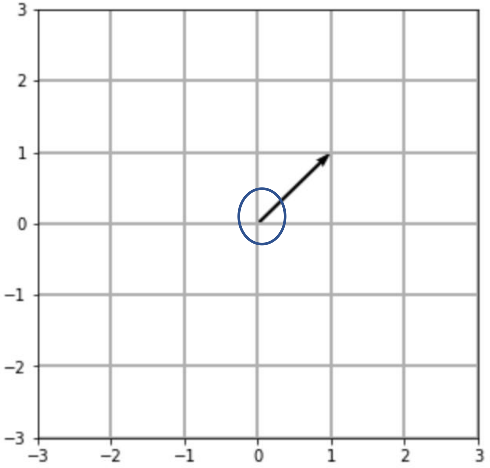
x_0 x_1

features



Matrices can transform vectors

- Specifically, they are **linear transformations**
- They transform ***vectors,*** for example by changing their length from the origin and angle from the the coordinate axes



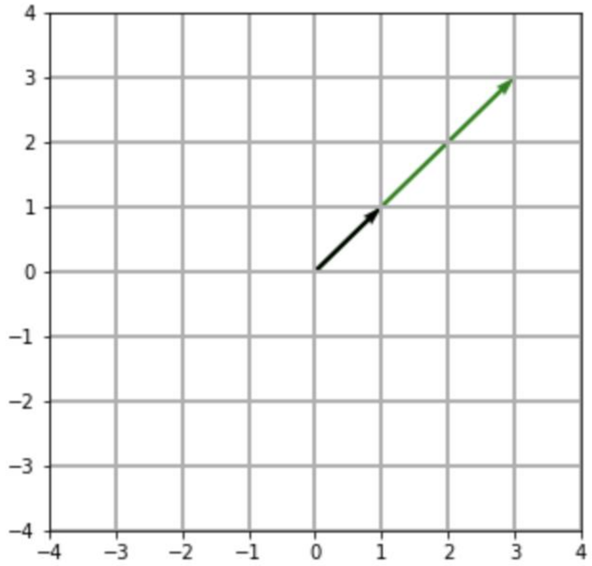
$$\begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Example 1: Stretching Matrix

Stretches a vector by a factor of K

$$\begin{bmatrix} K & 0 \\ 0 & K \end{bmatrix}$$

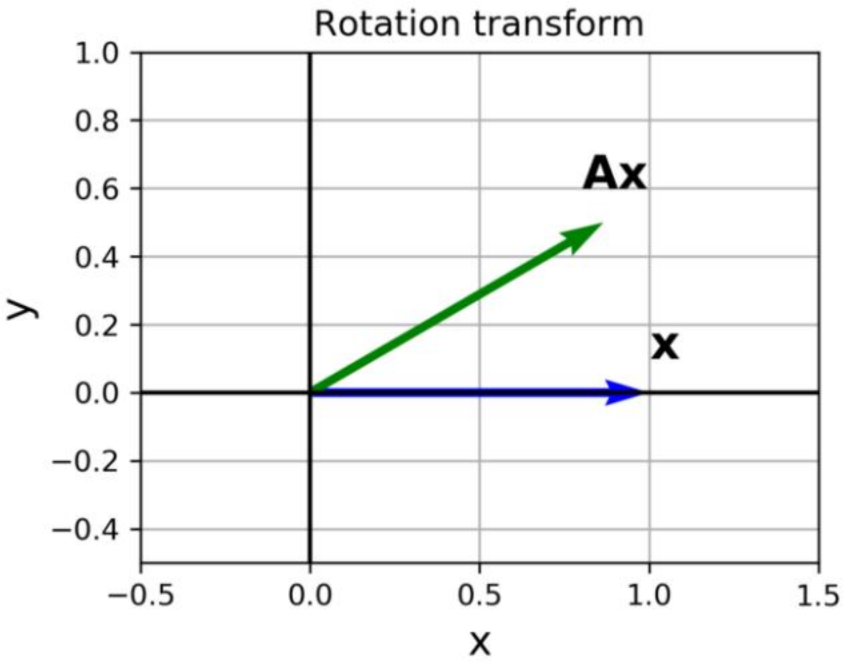
$$\begin{bmatrix} K & 0 \\ 0 & K \end{bmatrix} * \begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} KX \\ KY \end{bmatrix}$$



Example 2: Rotation matrix

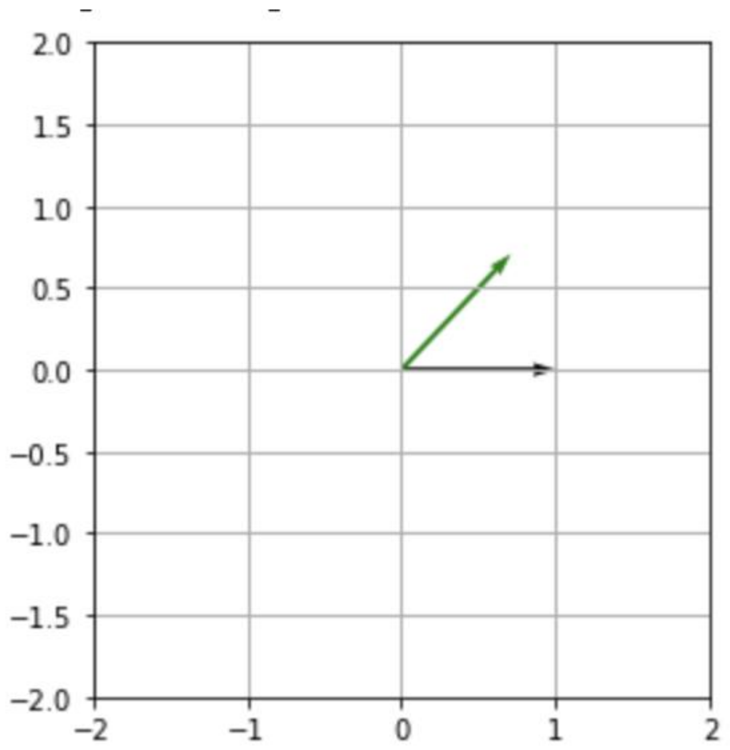
$$\begin{bmatrix} \cos(\varnothing) & -\sin(\varnothing) \\ \sin(\varnothing) & \cos(\varnothing) \end{bmatrix}$$

Rotates by an angle of Ø



$$\begin{bmatrix} \cos(\varnothing) & -\sin(\varnothing) \\ \sin(\varnothing) & \cos(\varnothing) \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} X cos(\varnothing) - Y sin(\varnothing) \\ Y sin(\varnothing) + Y cos(\varnothing) \end{bmatrix}$$

Example 2: Rotation matrix

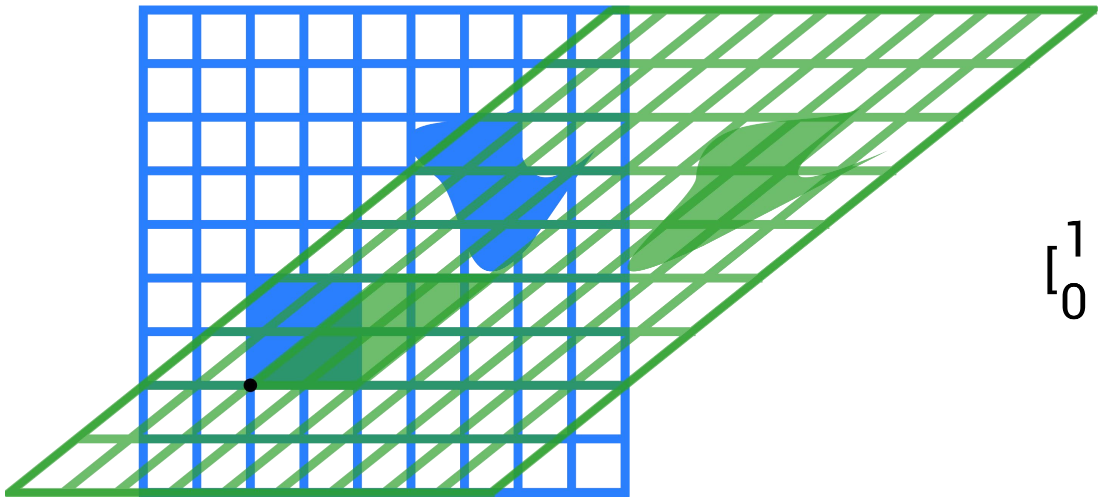


Here ∅ is $\frac{\pi}{4}$

Example 3: Shearing matrix

Shears a vector in one direction

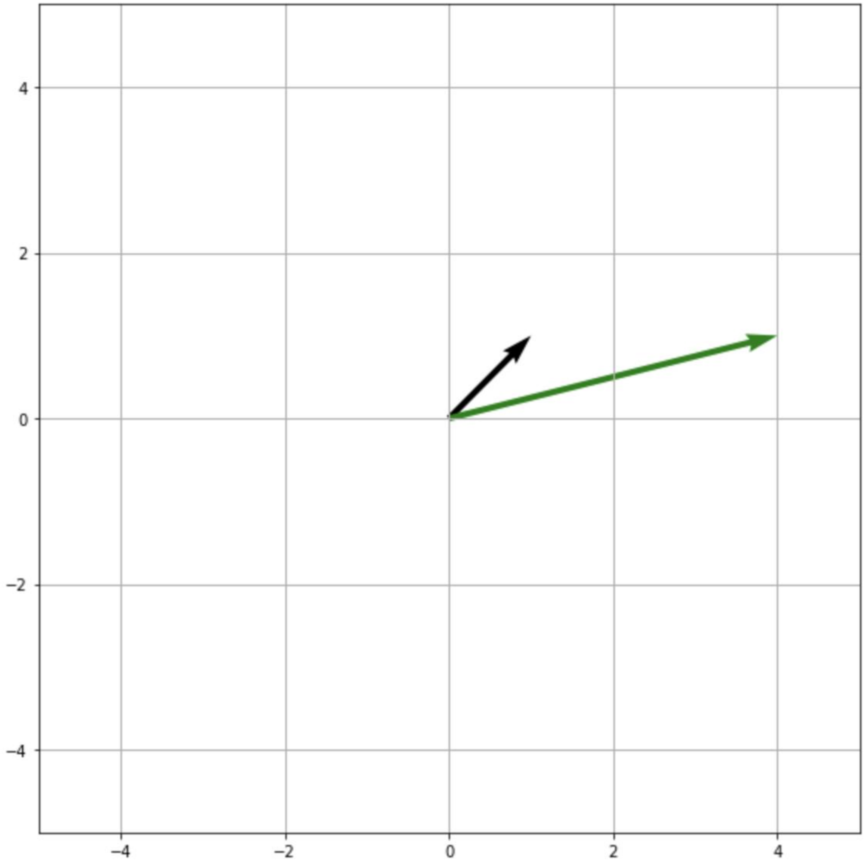
$$\begin{bmatrix} 1 & m \\ 0 & 1 \end{bmatrix}$$



From Wikipedia: sheer mapping

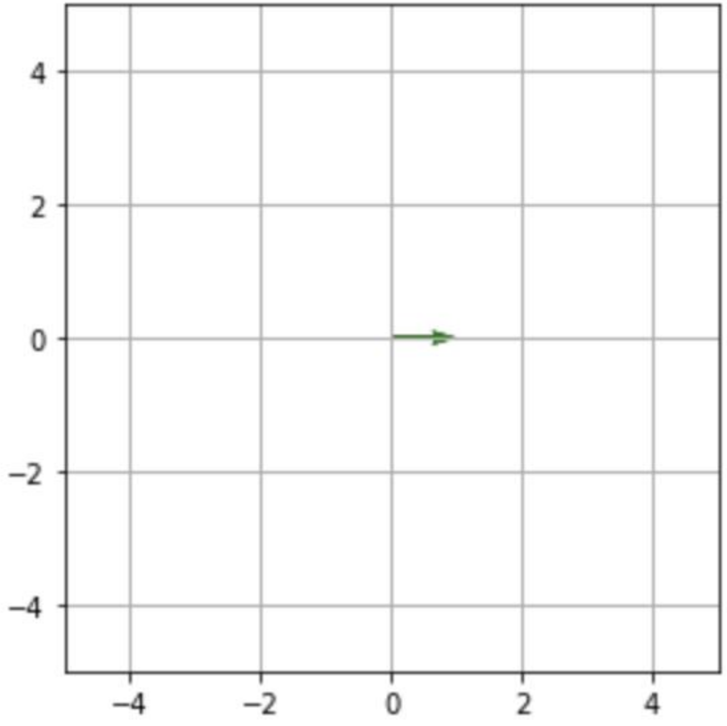
$$\begin{bmatrix} 1 & M \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} X + MY \\ Y \end{bmatrix}$$

Example 3: Shearing matrix



Shearing with M=3

Example 3: Shearing matrix



Shearing with M=3 applied to vector [1 0]

Did not change the vector!

Eigenvectors and Eigenvalues

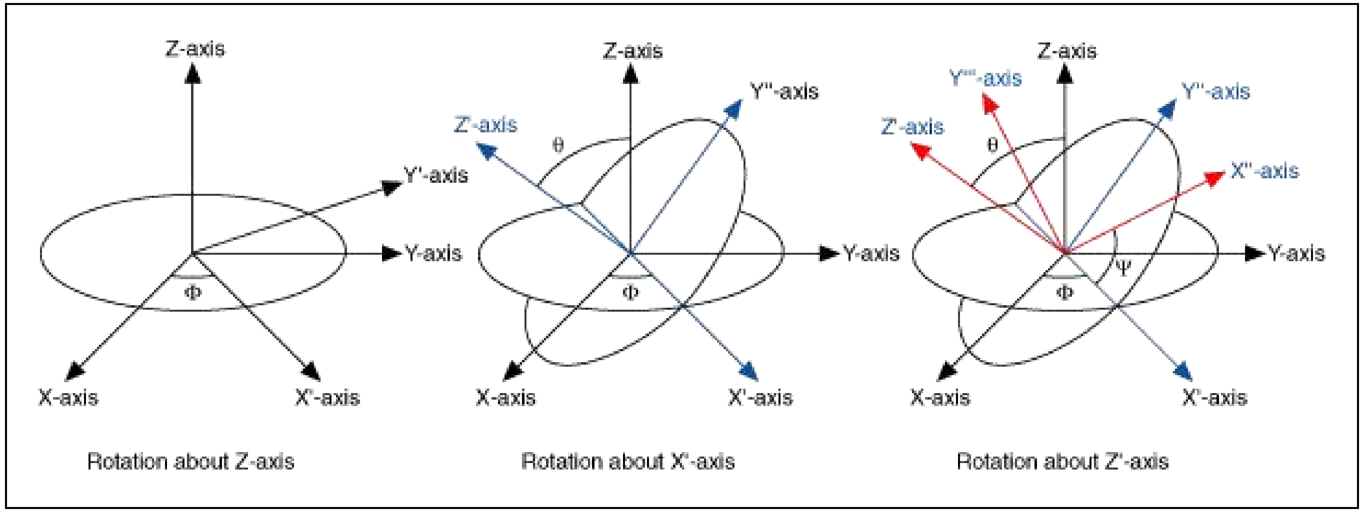
- Eigenvectors are vectors **x** that are only stretched by a linear transformation

- “Stretching” a vector is simply multiplying its coordinates by a scalar $Ax = \lambda x$

- Here this scalar λ is called the eigenvalue (or characteristic value), and x is called the **eigenvector**

Eigenvector meaning

- Linear transformations can be characterized by their invariant spaces, or vectors that they leave unchanged (but for scaling)
- Example: In a 3-D rotation, it can give you the axis of rotation



From Wikipedia

Eigendecomposition

- Decomposing a matrix into a product of eigenvectors, eigenvalues, and the inverse of the eigenvectors

$$A = U\Lambda U^{-1}$$

- This is equivalent to a change of basis to the eigenspace -->stretch-->change back

- $$\begin{bmatrix} & & \\ & A & \\ & & \end{bmatrix} = \begin{bmatrix} | & | & | \\ U_1 & U_2 & U_3 \\ | & | & | \end{bmatrix} \begin{bmatrix} \lambda_1 & & \\ & \lambda_2 & \\ & & \lambda_3 \end{bmatrix} \begin{bmatrix} | & | & | \\ U_1 & U_2 & U_3 \\ | & | & | \end{bmatrix}^{-1}$$

Finding eigenvectors and eigenvalues

Recall: Eigenvectors and Eigenvalues

- Eigenvectors are vectors **u** that are only stretched by a linear transformation
- “Stretching” a vector is simply multiplying its coordinates by a scalar $Au = \lambda$u
- Here this scalar λ is called the eigenvalue, and u is called the eigenvector

How do we find eigenvectors/values?

- By definition $Au = \lambda \mathrm{I} \mathrm{u}$, then $(\lambda I - \mathrm{A})\mathrm{u} = 0$
- The columns of λI-A are not linearly independent
- This means that the matrix has determinant 0
- Thus p(λ)=det(λI-A)=0 ---this is called the **characteristic polynomial**

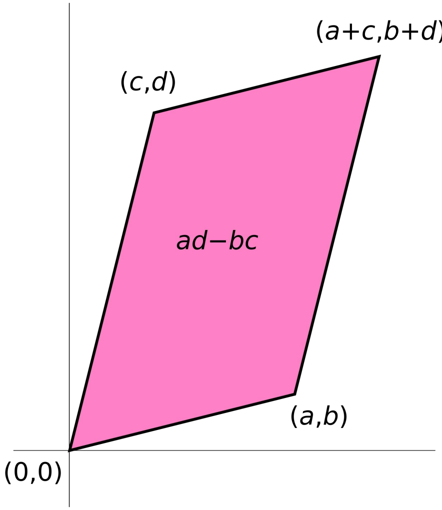
- Roots of the characteristic polynomial are eigenvalues

Finding eigenvectors

- Solving for $p(\lambda) = \det(\lambda I - A) = 0$ gives you eigenvalues λ_1, λ_2 ...
- Then the eigenvectors can be found by solving for each component of $\boldsymbol{\lambda}_i x - A = 0$

What is the determinant?

- It is the volume scaling factor of the matrix viewed as a linear transformation, i.e., what is the volume the transformation scales a unit cube by?
- Negative determinants signal change in orientation
- Determinant of a 2x2 matrix

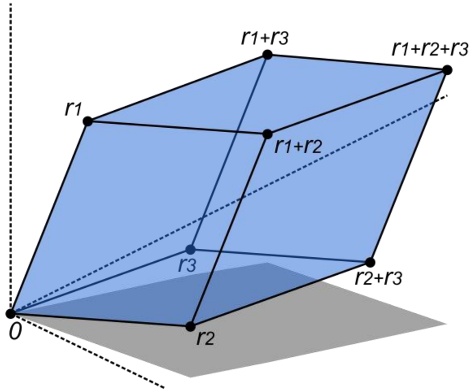


$$|A| = \begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc.$$

Area of the parallelogram that is formed by these vectors is the determinant

Higher dimensional determinants

Determinant of a 3x3 matrix



$$|A| = \begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} = a \begin{vmatrix} e & f \\ h & i \end{vmatrix} - b \begin{vmatrix} d & f \\ g & i \end{vmatrix} + c \begin{vmatrix} d & e \\ g & h \end{vmatrix}$$

$$\begin{vmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{vmatrix} = a \begin{vmatrix} f & g & h \\ j & k & l \\ n & o & p \end{vmatrix} - b \begin{vmatrix} e & g & h \\ i & k & l \\ m & o & p \end{vmatrix} + c \begin{vmatrix} e & f & h \\ i & j & l \\ m & n & p \end{vmatrix} - d \begin{vmatrix} e & f & g \\ i & j & k \\ m & n & o \end{vmatrix}.$$

Example 3: Finding Eigenvectors

$$\mathbf{A} = \begin{bmatrix} 3 & 2 \\ 0 & 2 \end{bmatrix} \qquad \lambda I - A = \begin{bmatrix} \lambda - 3 & -2 \\ 0 & \lambda - 2 \end{bmatrix}$$

$$p_A(\lambda) = \det(\lambda I - A) = (\lambda - 3)(\lambda - 2) = 0$$

Eigenvalues are $\lambda_1 = 3, \lambda_2 = 2$

$$3I - A = \begin{bmatrix} 0 & -2 \\ 0 & 3-2 \end{bmatrix} = \begin{bmatrix} 0 & -2 \\ 0 & 1 \end{bmatrix}$$

$\begin{bmatrix} 0 & -2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ ➔ Y is 0, X is anything

Eigenvector $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$

Example 3: Finding Eigenvectors

Eigenvalues are $\lambda_1 = 3, \lambda_2 = 2$

$$2I - A = \begin{bmatrix} 2-3 & -2 \\ 0 & 2-2 \end{bmatrix} = \begin{bmatrix} -1 & -2 \\ 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} -1 & -2 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$ ➔ -X=2Y

Eigenvector $\begin{bmatrix} 2 \\ -1 \end{bmatrix}$

Power iteration method

- To find the largest eigenvectors of a matrix A
- Start with a random vector b_0
- Then repeat until convergence

$$b_{k+1} = \frac{Ab_k}{||Ab_k||}$$

- Why does this work?
  - If A has a dominant eigenvector u, then random vector b_0 will have a component in the direction of u
  - This component gets stretched by application of A and becomes a larger part of the magnitude of b_1 and so forth ..

Power iteration: finding eigenvalues

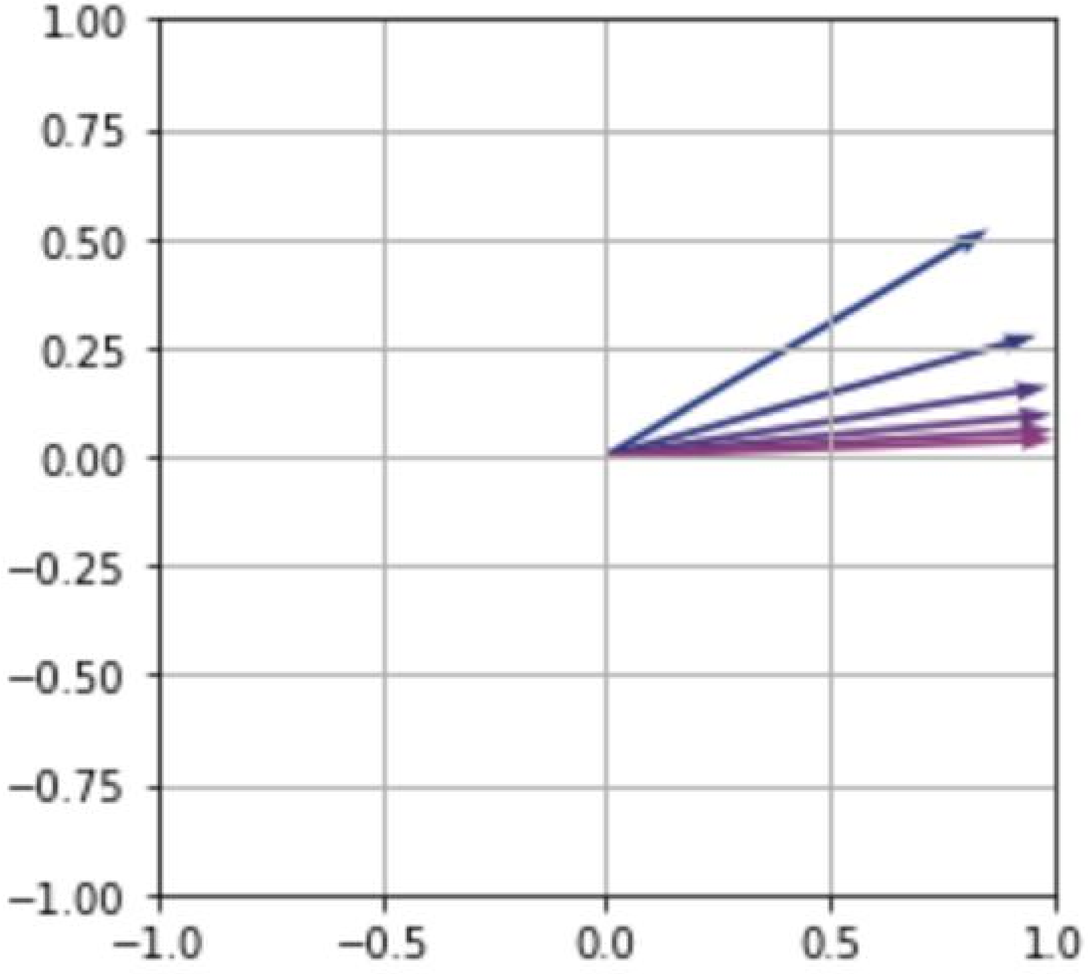
- The power iteration method finds the largest eigenvector
- How do we find its eigenvalue?
- Note that the eigenvalue $\lambda = \frac{A\,u^T u}{u^T u}$
- To see this, substitute $Au = \lambda u$

$$\lambda = \frac{\lambda(u^T u)}{(u^T u)}$$

- This is also called the **Rayleigh quotient**

Example

$$\mathbf{A} = \begin{bmatrix} 3 & 2 \\ 0 & 2 \end{bmatrix}$$



Next eigenvector

- To get the next eigenvector, effectively take out the largest eigenvector from the matrix

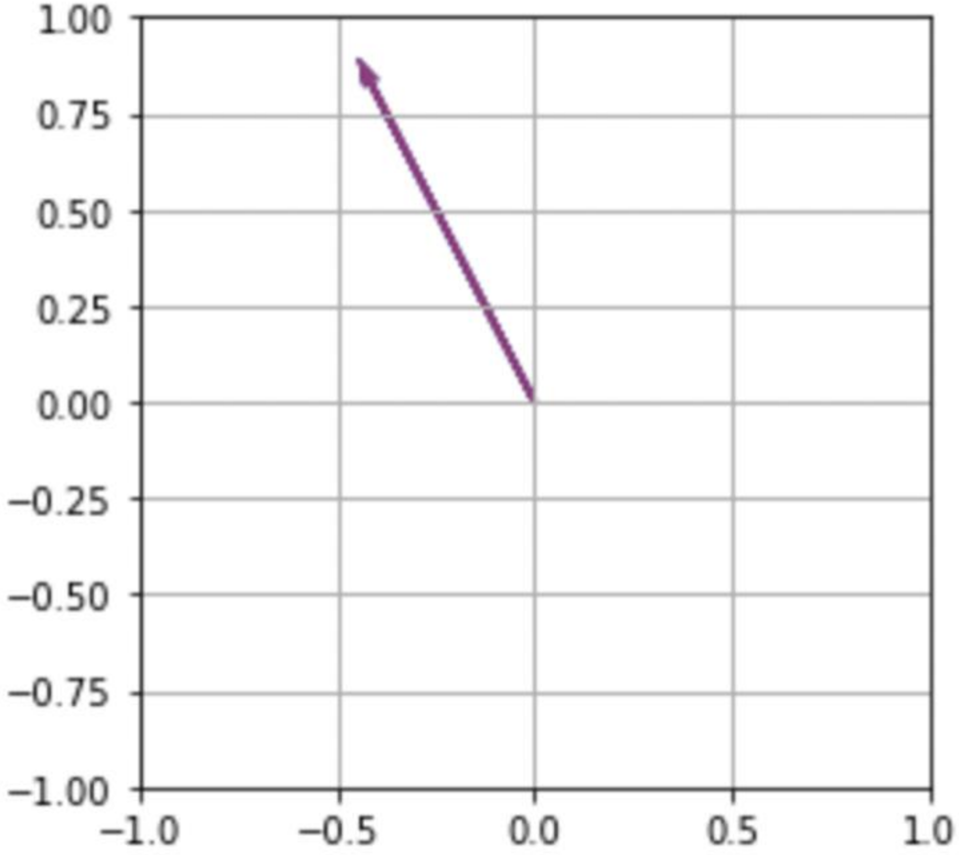
- $B = A - \lambda u u^T$

- The largest eigenvalue of B is the second largest eigenvalue of A

Example

$$B = A - 3\left(\begin{bmatrix}1\\0\end{bmatrix}\begin{bmatrix}1 & 0\end{bmatrix}\right)$$

$$B = \begin{bmatrix}0 & -1\\0 & 1\end{bmatrix}$$



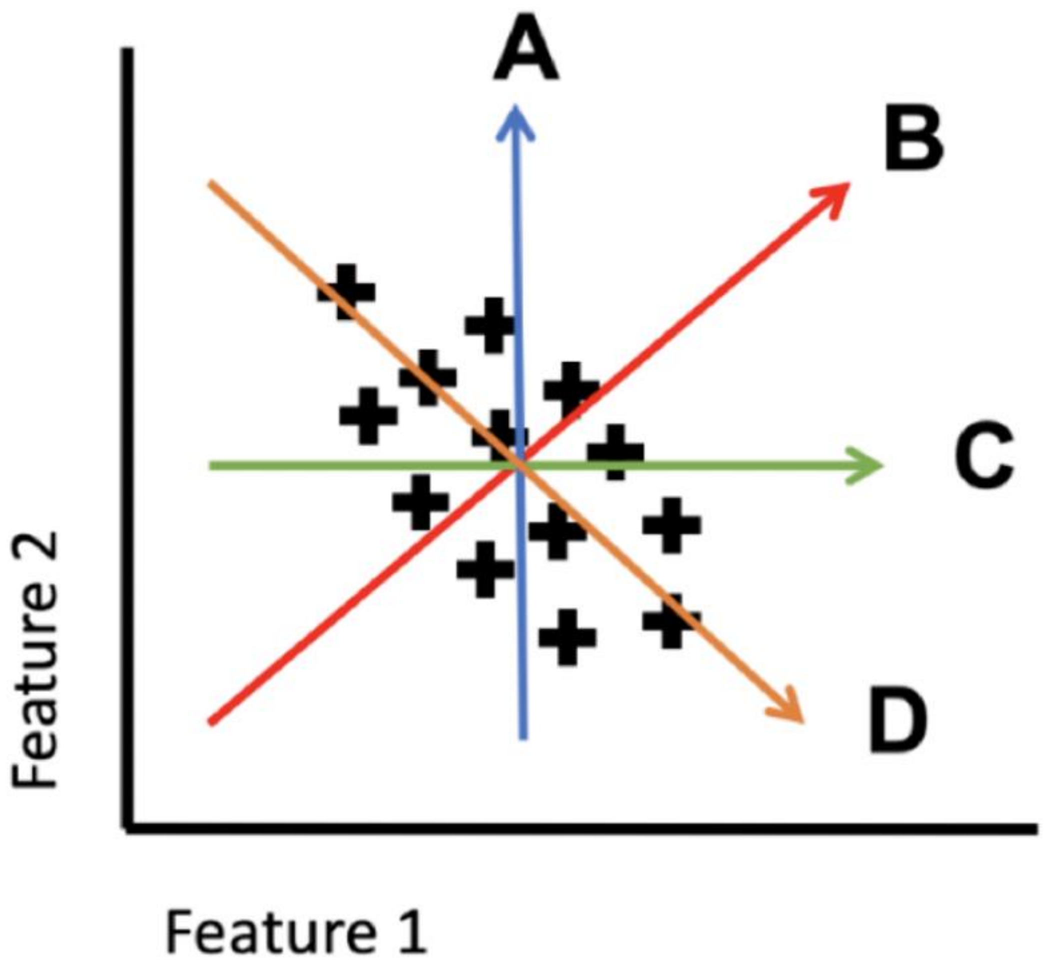
Undiagonalizable Matrices

- Matrices that have multiplicities in their eigenvalue but not equivalent multiplicity in their eigenvectors

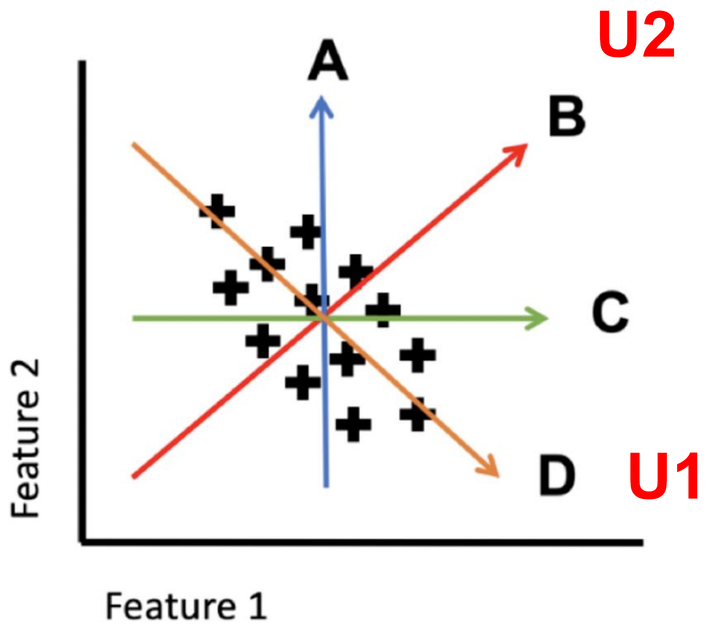
- Example:
- $A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$
- $\lambda I - A = \begin{bmatrix} \lambda - 1 & -1 \\ 0 & \lambda - 1 \end{bmatrix}$
- $p_A(\lambda) = (\lambda - 1)^2$

- This matrix has eigenvalue -1 with multiplicity 2
- but only one eigenvector: $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$

Variance, Covariance, PCA


A
B
C
D
Feature 2
Feature 1

PCA



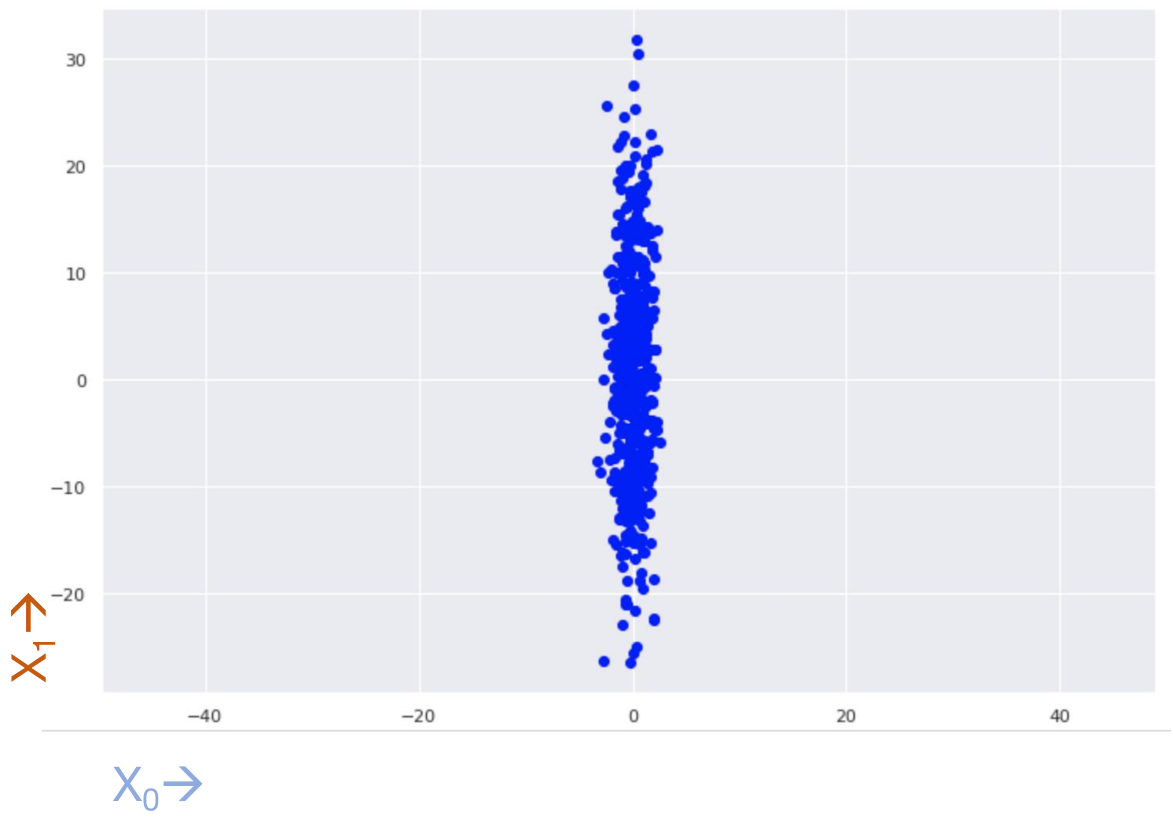
PCA finds directions in the data that explain the most variance

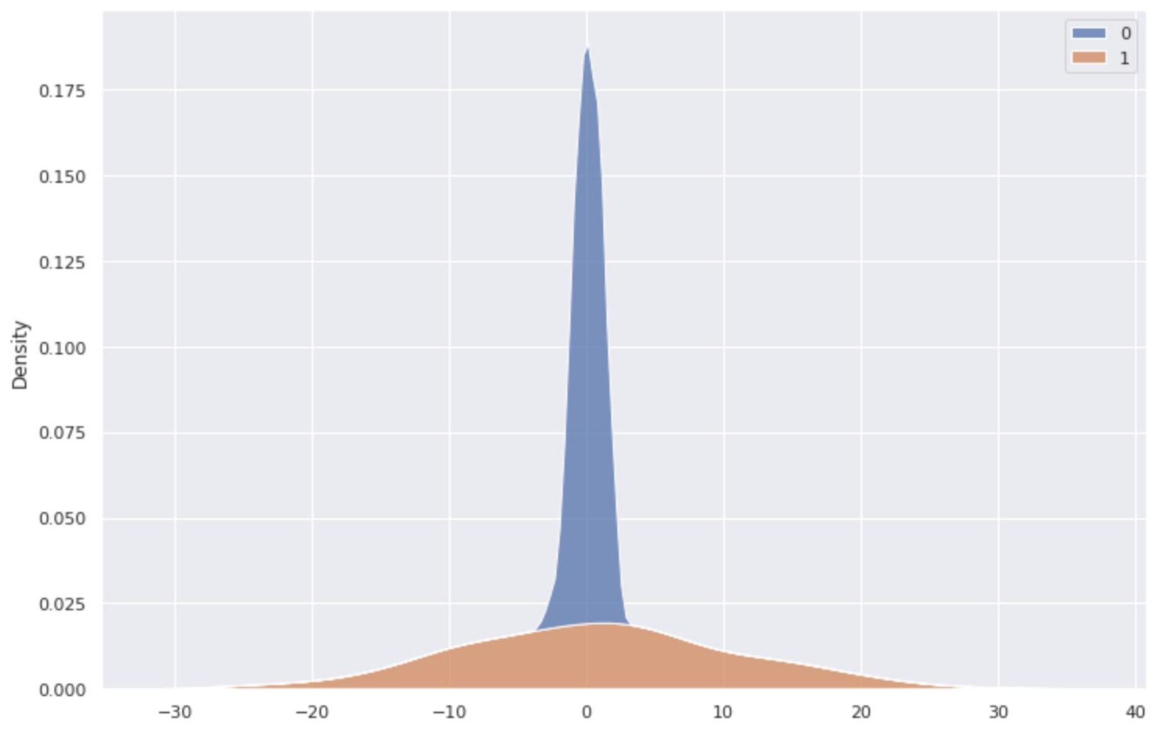
PC1 projects data to the line that explains the most variance

PC2 to the line that explains the next most (while being orthogonal)

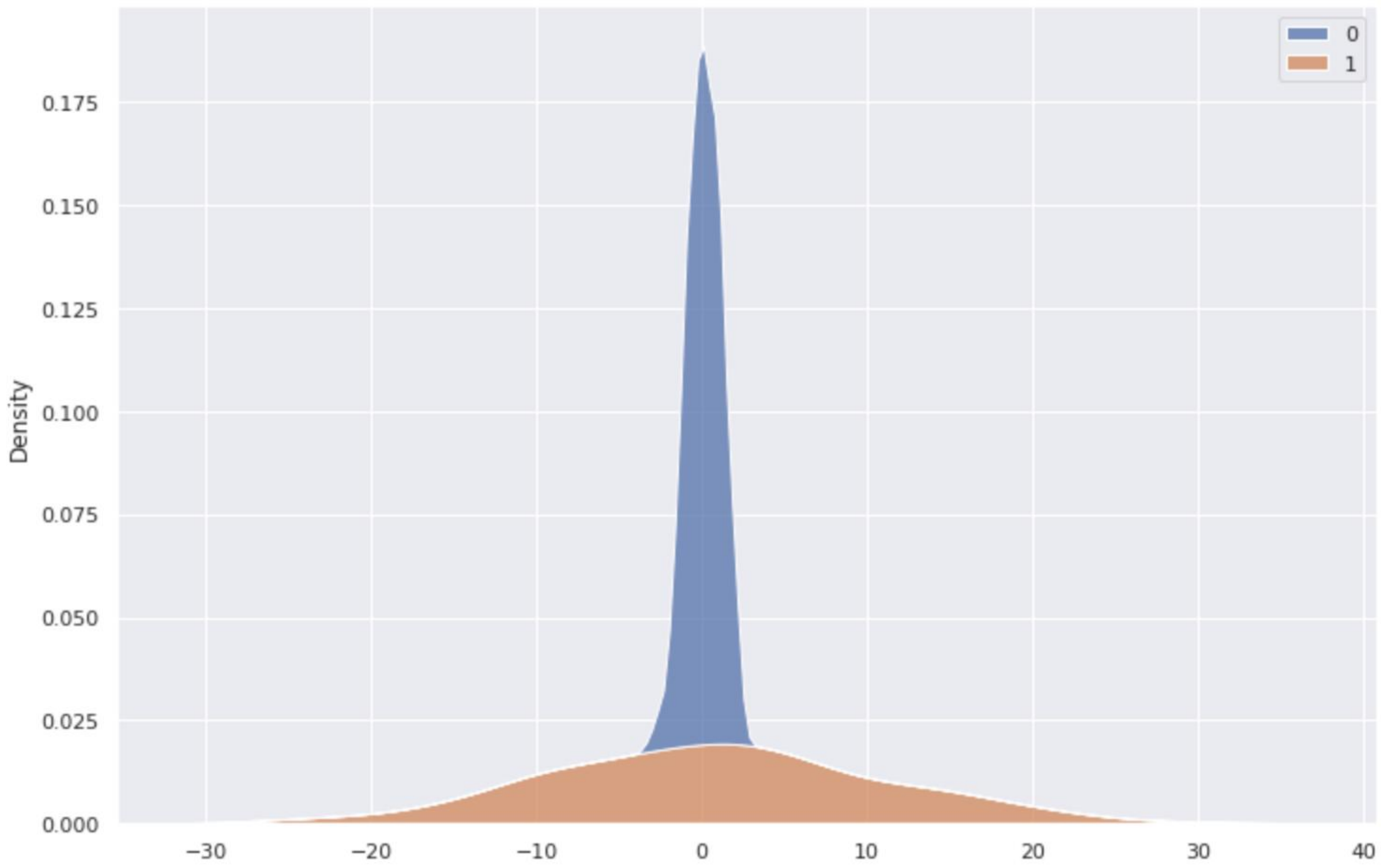
How do we find these directions?

Spread of data





Variance: A measure of spread



$$Var(X) = E[(X - E[X])^2]$$

$$Var(X_0) = 0.94$$

$$Var(X_1) = 86.22$$

Population variance: $\frac{1}{M}\sum_i (X_i - \bar{X})^2$

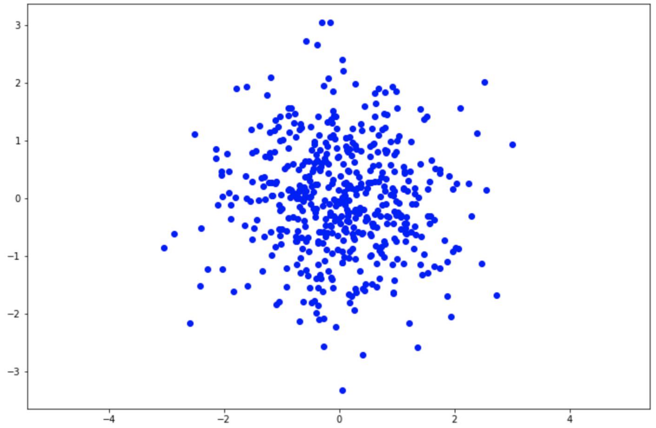
sample variance: $\frac{1}{M-1}\sum_i (X_i - \bar{X})^2$

Covariance: Joint Variability

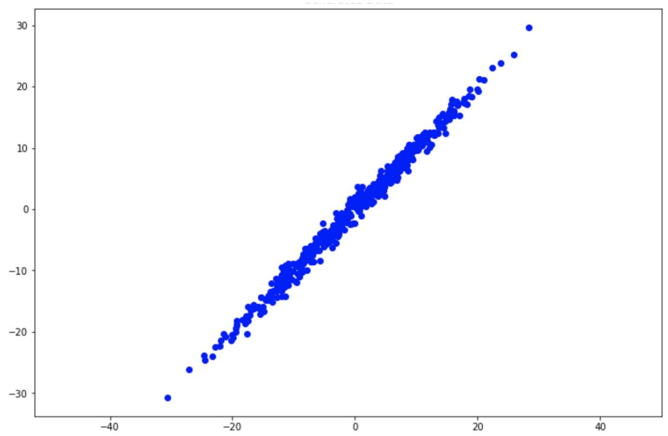
Population covariance: $\frac{1}{M}\sum_i (X_i - \bar{X})(Y_i - \bar{Y})$

$$\boldsymbol{cov(X, Y) = E[(X - E[X])(Y - E[Y])]}$$

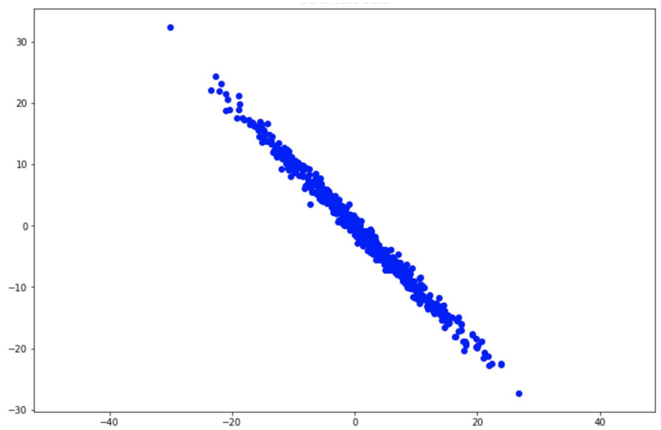
sample covariance: $\frac{1}{M-1}\sum_i (X_i - \bar{X})(Y_i - \bar{Y})$



$cov(X, Y) = 0.38$

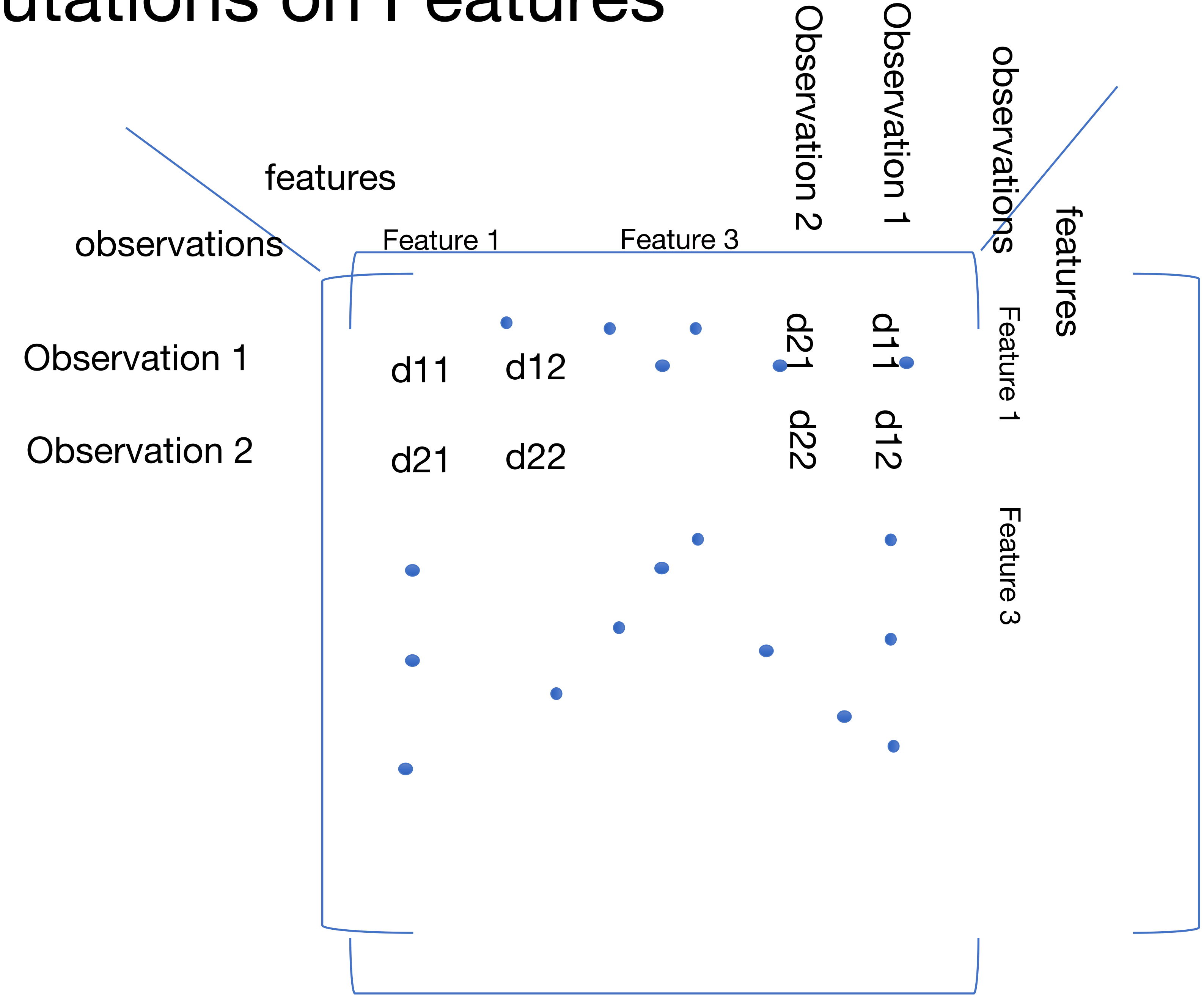


$cov(X, Y) = 99.91$



$cov(X, Y) = -93.31$

Computations on Features



Covariance on features

Observations

Features

Observation 1

$$\begin{array}{l} \text{Feature 1} \\ \text{Feature 2} \\ \\ \\ \\ \end{array} \begin{bmatrix} d11 & d12 & \cdots \\ d21 & d22 & \\ \vdots & \ddots & \\ & & \end{bmatrix}$$

Covariance Matrix

If there are n features in your data, you can compute a matrix of covariances

$$\Sigma = \begin{bmatrix} cov(X_1, X_1) & \cdots & cov(X_1, X_n) \\ \vdots & \ddots & \vdots \\ cov(X_n, X_1) & \cdots & cov(X_n, X_n) \end{bmatrix}$$

Easy way to compute Σ if $E(X_i) = 0$

$$\Sigma = \begin{bmatrix} E[(X_1 - E[X_1])(X_1 - E[X_1])] & \cdots & E[(X_1 - E[X_1])(X_n - E[X_n])] \\ \vdots & \ddots & \vdots \\ E[(X_n - E[X_n])(X_1 - E[X_1])] & \cdots & E[(X_n - E[X_n])(X_n - E[X_n])] \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} cov(X_1, X_1) & \cdots & cov(X_1, X_n) \\ \vdots & \ddots & \vdots \\ cov(X_n, X_1) & \cdots & cov(X_m, X_n) \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} E[(X_1)(X_1)] & \cdots & E[(X_1)(X_n)] \\ \vdots & \ddots & \vdots \\ E[(X_n)(X_1)] & \cdots & E[(X_n)(X_n)] \end{bmatrix}$$

$$\Sigma = \frac{1}{(M-1)} X X^T$$

For M datapoints

Covariance as a bilinear form

- The covariance matrix doesn't just "store" covariances
- Σ actually represents a linear transformation!
- Computes covariance of new variables $w,\ v$ which are combinations of variables on which Σ is computed

- Thus $cov(v, w) = v^T \Sigma w$

- For a single variable $cov(v, v) = var(v) = v^T \Sigma v$

Example

- Variance of variables $v = a_1X_1 + a_2X_2 \ldots\ a_nX_n$,

$$[a_1 \quad a_2 \quad a_3] \begin{bmatrix} cov(X_1, X_1) & cov(X_1, X_2) & cov(X_1, X_3) \\ cov(X_2, X_1) & cov(X_2, X_2) & cov(X_2, X_3) \\ cov(X_3, X_1) & cov(X_3, X_2) & cov(X_3, X_3) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

$$= \quad [a_1 \quad a_2 \quad a_3] \begin{bmatrix} a_1cov(X_1, X_1) + a_2cov(X_1, X_2) + a_3cov(X_1, X_3) \\ a_1cov(X_2, X_1) + a_2cov(X_2, X_2) + a_3cov(X_2, X_3) \\ a_1cov(X_3, X_1) + a_2cov(X_3, X_2) + a_3cov(X_3, X_3) \end{bmatrix}$$

$$= a_1{}^2cov(X_1, X_1) + a_1a_2cov(X_1, X_2) + a_1a_3cov(X_1, X_3) + a_2a_1cov(X_2, X_1) + a_2{}^2cov(X_2, X_2) + a_2a_3cov(X_2, X_3) + a_3a_1cov(X_3, X_1) + a_3a_2cov(X_3, X_2) + a_3{}^2cov(X_3, X_3)$$

Example

- Covariance of variables $v = a_1X_1 + a_2X_2 \dots a_nX_n$, $w = b_1X_1 + b_2X_2 \dots b_nX_n$

$$[b_1 \quad b_2 \quad b_3] \begin{bmatrix} cov(X_1, X_1) & cov(X_1, X_2) & cov(X_1, X_3) \\ cov(X_2, X_1) & cov(X_2, X_2) & cov(X_2, X_3) \\ cov(X_3, X_1) & cov(X_3, X_2) & cov(X_3, X_3) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

$$= \quad [b_1 \quad b_2 \quad b_3] \begin{bmatrix} a_1cov(X_1, X_1) + a_2cov(X_1, X_2) + a_3cov(X_1, X_3) \\ a_1cov(X_2, X_1) + a_2cov(X_2, X_2) + a_3cov(X_2, X_3) \\ a_1cov(X_3, X_1) + a_2cov(X_3, X_2) + a_3cov(X_3, X_3) \end{bmatrix}$$

$$= b_1a_1cov(X_1, X_1) + b_1a_2cov(X_1, X_2) + b_1a_3cov(X_1, X_3) + b_2a_1cov(X_2, X_1) + b_2a_2cov(X_2, X_2) + b_2a_3cov(X_2, X_3) + b_3a_1cov(X_3, X_1) + b_3a_2cov(X_3, X_2) + b_3a_3cov(X_3, X_3)$$

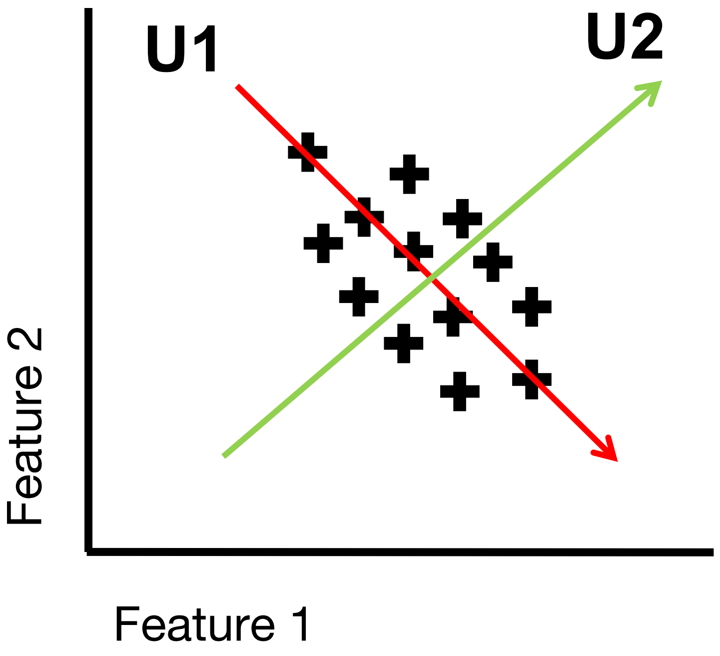
Properties of the covariance matrix

- Symmetric $\boldsymbol{cov}(\boldsymbol{X}, \boldsymbol{Y}) = \boldsymbol{cov}(\boldsymbol{Y}, \boldsymbol{X})$

- Positive semidefinite $v^T \Sigma v \geq 0$
  - Seen by the fact that the diagonal contains variance
  - Variance is always positive

- Positive semidefinite matrices have all positive, real eigenvalues

Principal Components Analysis



PCA finds directions in the data that explain the most variance

U1 is the eigenvector of Σ with *largest eigenvalue*

U2 is the eigenvector of Σ with the *second largest eigenvalue*…

Geometric Interpretation of the Covariance Matrix

Covariance Matrix

If there are n features in your data, you can compute a matrix of covariances

$$\Sigma = \begin{bmatrix} cov(X_1, X_1) & \cdots & cov(X_1, X_n) \\ \vdots & \ddots & \vdots \\ cov(X_n, X_1) & \cdots & cov(X_n, X_n) \end{bmatrix}$$

Eigendecomposing Σ

- Decomposing a matrix into a product of eigenvectors, eigenvalues, and the inverse of the eigenvectors

$$\Sigma = U \Lambda U^{-1}$$

- Note the U is a rotation matrix and Λ is a diagonal matrix

- $\begin{bmatrix} & & \\ & \Sigma & \\ & & \end{bmatrix} = \begin{bmatrix} | & | & | \\ U_1 & U_2 & U_3 \\ | & | & | \end{bmatrix} \begin{bmatrix} \lambda_1 & & \\ & \lambda_2 & \\ & & \lambda_3 \end{bmatrix} \begin{bmatrix} | & | & | \\ U_1 & U_2 & U_3 \\ | & | & | \end{bmatrix}^{-1}$

Rewriting Σ

- We can rewrite this matrix $\Sigma = U\Lambda U^{-1}$ =U$LLU^{-1}$where L = $\sqrt{\Lambda}$
  - This is just the square root of the values along the diagonals

$$L=\begin{bmatrix} \sqrt{\lambda_1} & & \\ & \sqrt{\lambda_2} & \\ & & \sqrt{\lambda_3} \end{bmatrix}$$

- Applying the matrix C=UL to *n* normal random variables X = [$N(0,1), N(0,1)$, … $N(0,1)$] results in rotated scaled data $\tilde{X}$, whose covariance matrix is Σ!

Proof

- Fact 1: For mean-centered data $\tilde{X}$ the covariance matrix $\tilde{\Sigma} = \tilde{X}\tilde{X}^T$
- Fact 2: N independent normal random variables $N(0,1)$ have variance 1, and 0 covariance, so their covariance matrix is the $n \times n$ identity matrix I_n
- Fact 3: For an orthonormal matrix $U^{-1} = U^T$
- Given this, new data $\tilde{X} = ULX$
- $\tilde{\Sigma} = \tilde{X}\tilde{X}^T = (ULX)(X^T L^T U^T) = ULI_n L^T U^T = ULL^T U^T$
  - But L is just a diagonal matrix so $L^T = L$ and $LL = \Lambda$
- $\tilde{\Sigma} = ULXX^T L^T U^T = ULI_n L^T U^T = ULL^T U^T = U\Lambda U^T = U\Lambda U^{-1} = \Sigma$

Covariance geometry

$C = UL$

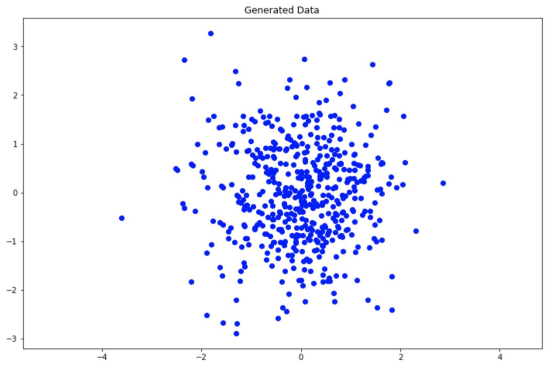
C^{-1}

X

$\tilde{X} = ULX$

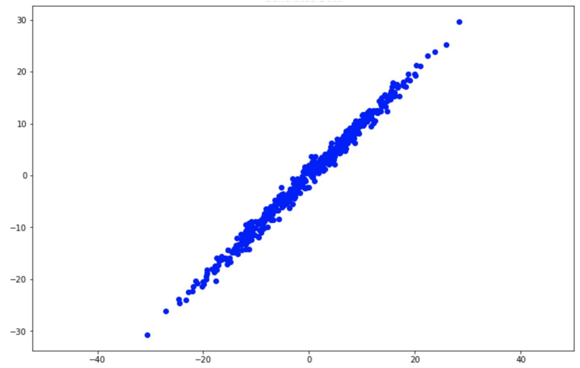
$X^{'} = (UL^{-1})\tilde{X} = X$

$N(0,1)$



$N(0,1)$

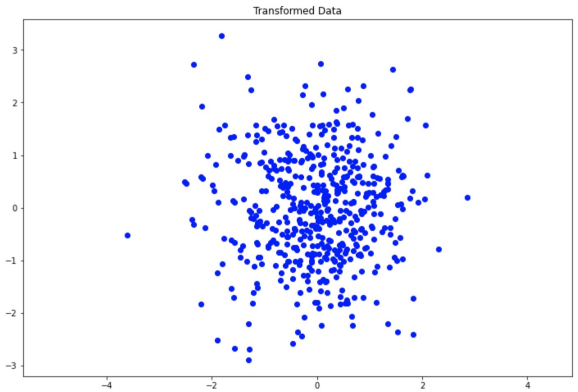
Uncorrelated data



Correlated data

$\text{Cov}(\tilde{X}) = \Sigma$

Σ is covariance matrix of another correlated data

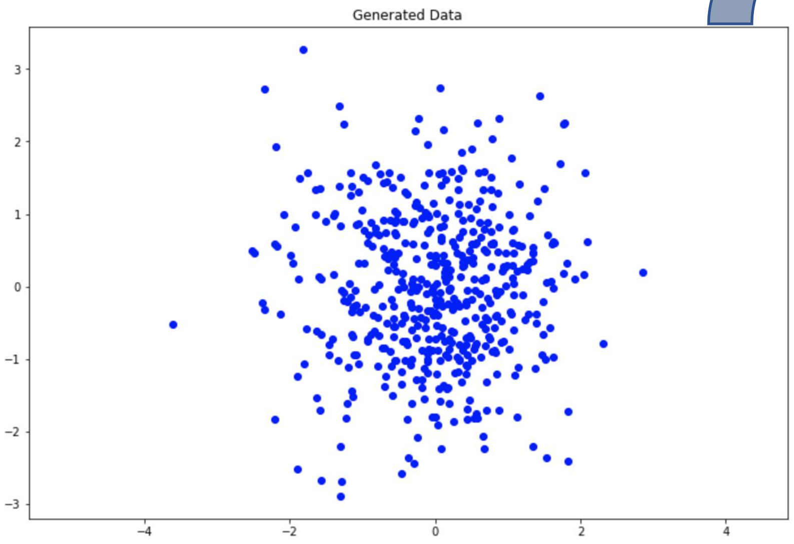


Back to uncorrelated data again

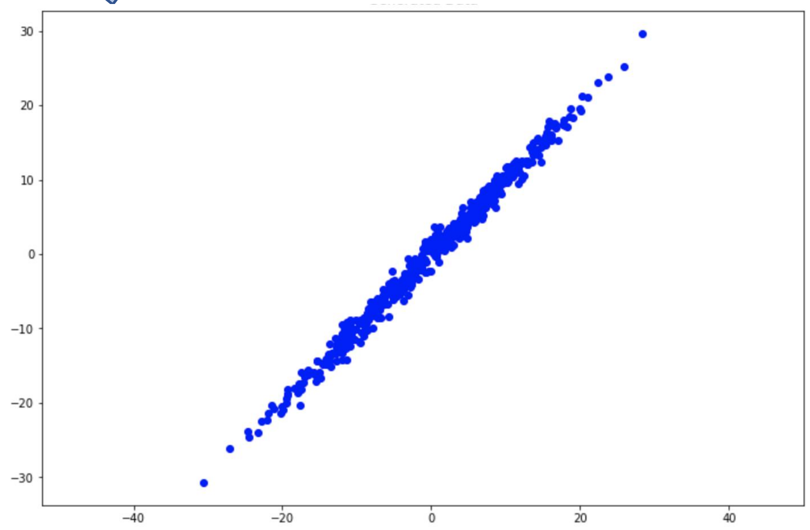
Covariance geometry

C=UL

$N(0,1)$



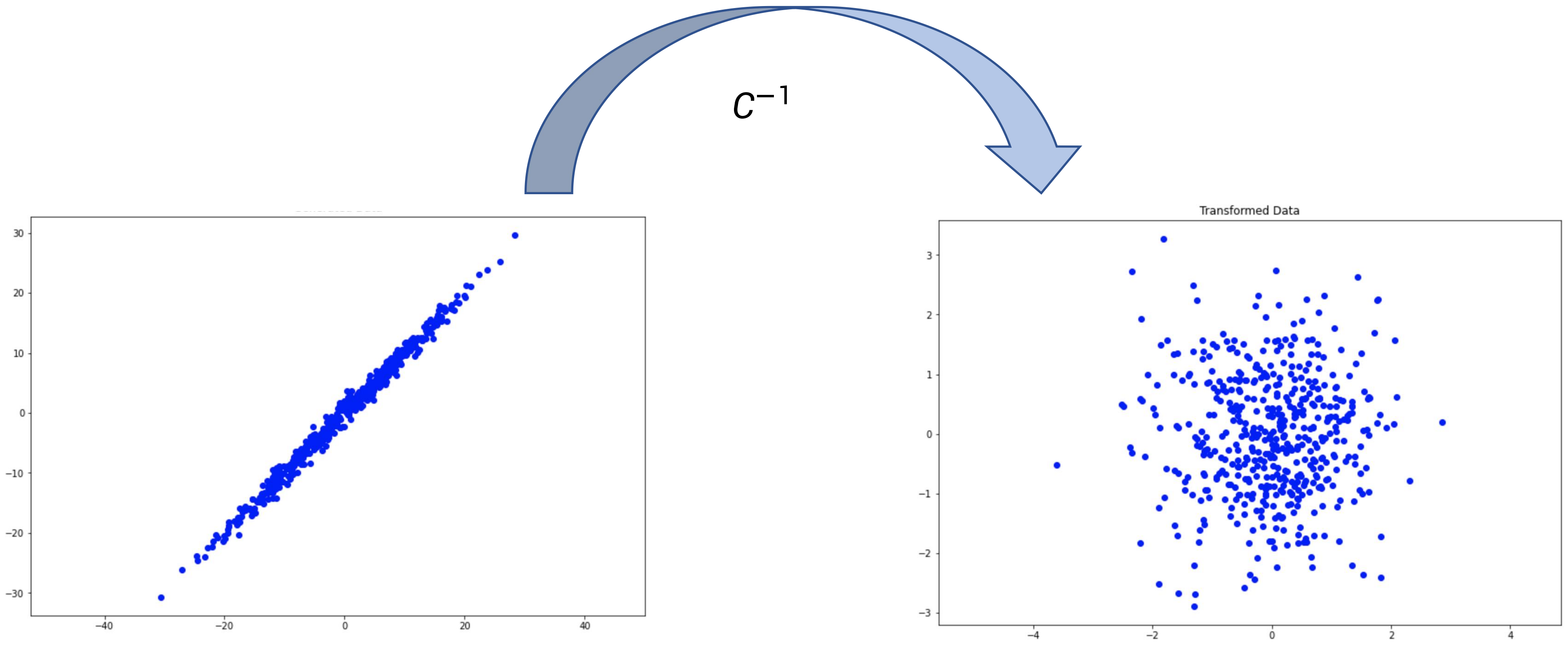
$N(0,1)$



Data with Σ as covariance

Inverse Covariance as an operator

- Inverse of C can be used to whiten or decorrelate data

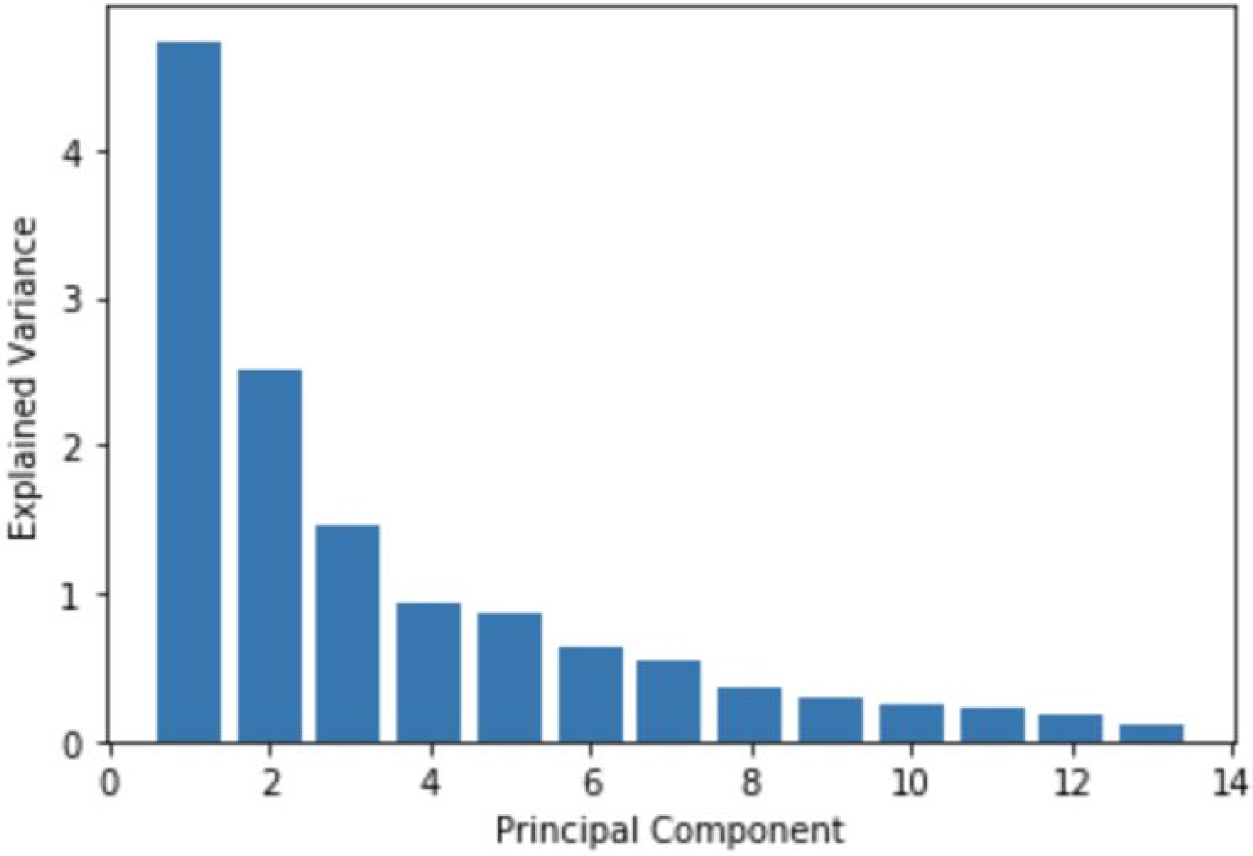


Denoising and Low Rank Approximation with SVD

Dimensionality Reduction

- One of the key applications of PCA is dimensionality reduction
- This is useful for:
  - Visualization *
  - Data Compression
  - Denoising

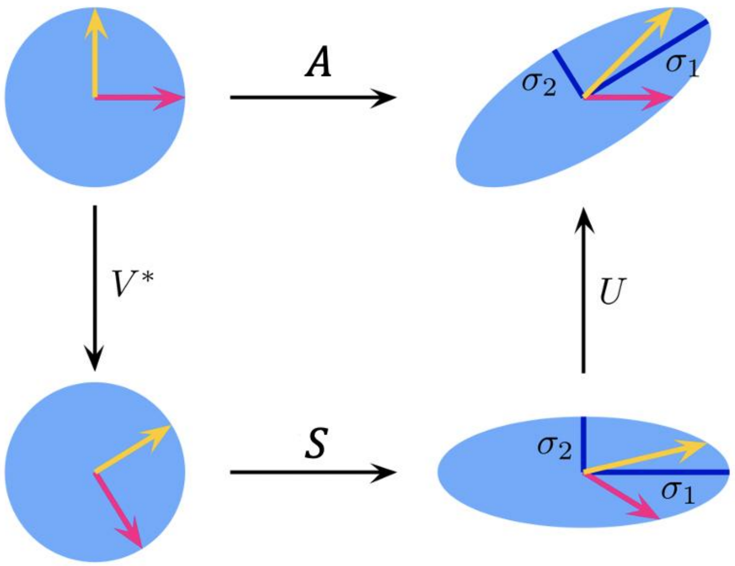
Intrinsic Dimensionality Estimation



Suggests that we can recreate the data using fewer features

Singular Value Decomposition

- Generalization of eigendecomposition to non-square matrices



$$A = USV^T$$

Singular Value Decomposition

- ANY real matrix has a singular value decomposition
- $A = USV^T$
- V^T is the transpose of V
- U, V are orthogonal matrices
- Orthogonal matrices have orthonormal column vectors
- These matrices have the property that $UU^T = I$

SVD and Eigendecomposition

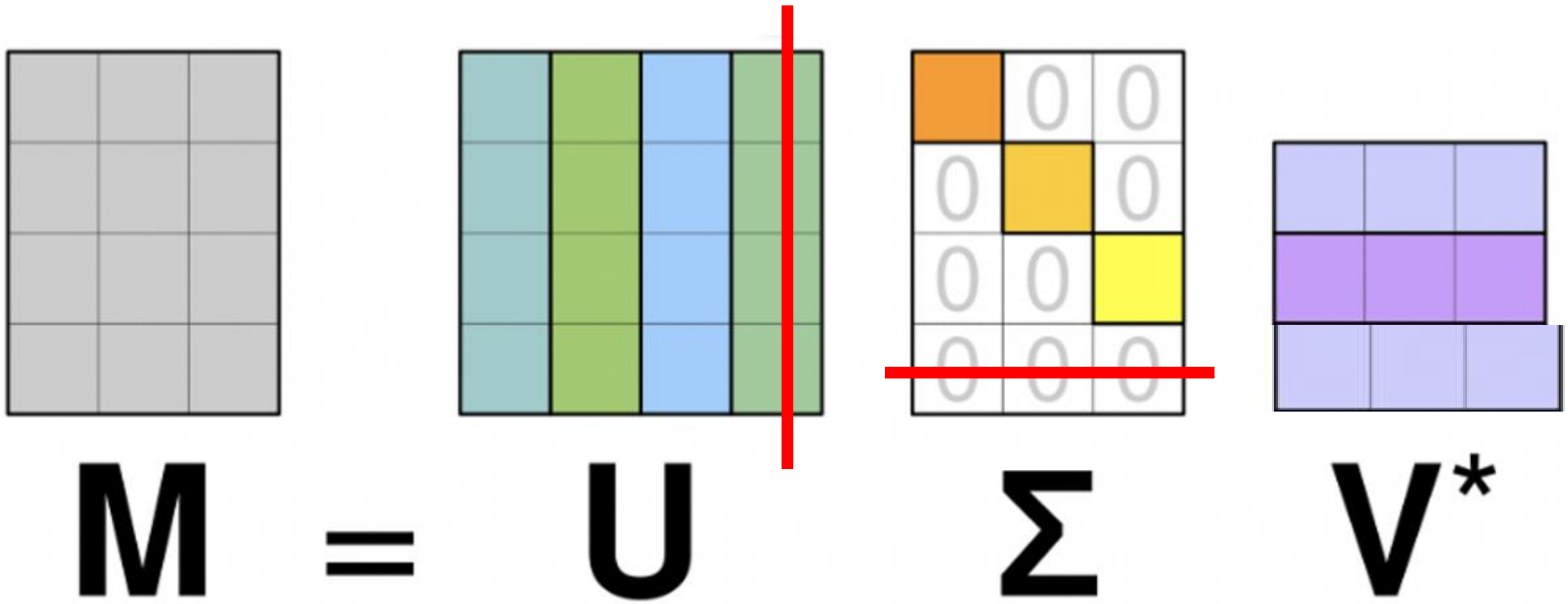
- $A = USV^T$
- $AA^T = USV^T VSU^T = U\, S^2\, U^T = US^2U^{-1}$
- $A^TA = VSU^T \mathrm{U}\, SV^T = VS^2V^T = VS^2V^{-1}$
- The left singular vectors are eigenvectors of AA^T
- The right singular vectors are eigenvectors of A^TA
- Eigenvalues of A^TA are the squar

Recall fact: $(\mathbf{AB})^T = \mathbf{B}^T\mathbf{A}^T$

PCA via SVD

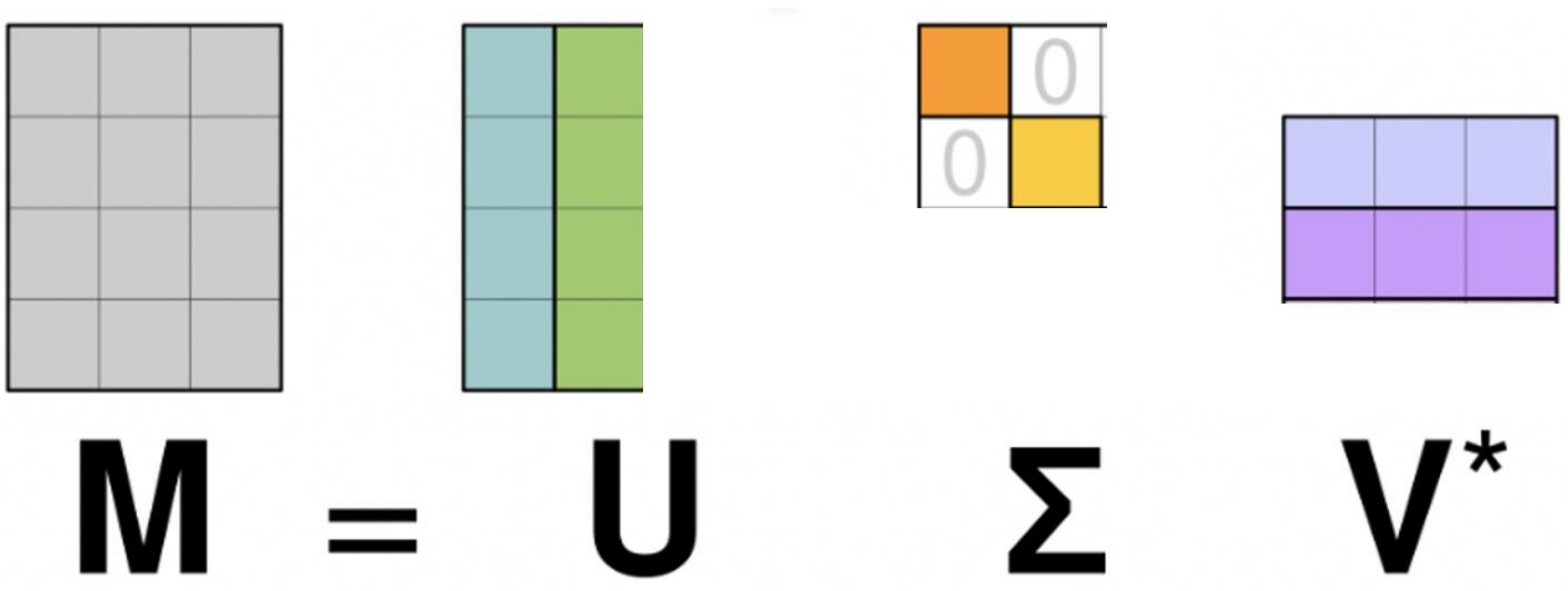
- Suppose our data matrix has SVD $X = USV^T$
- Cov(X) $= E([\mathrm{X} - \bar{X}][X^T - \bar{X^T}]) = \frac{[\mathrm{X} - \bar{X}]\,[X^T - \bar{X^T}]}{n}$
  - Where X is an n-dimensional vector
  - Suppose X is mean centered
- Reduces to $\frac{XX^T}{n}$ but since n is just a scalar we can drop it,
- Here we had flipped the rows and columns of this matrix
- Eigenvectors of XX^T are columns of V
- The principal components are $\tilde{X} = XV = USV^TV = US$!

Low rank approximation



	M	U	Σ	V*
Original	$m \times n$	$m \times m$	$m \times n$	$n \times n$
low rank	$m \times n$	$m \times r$	$r \times n$	$n \times n$

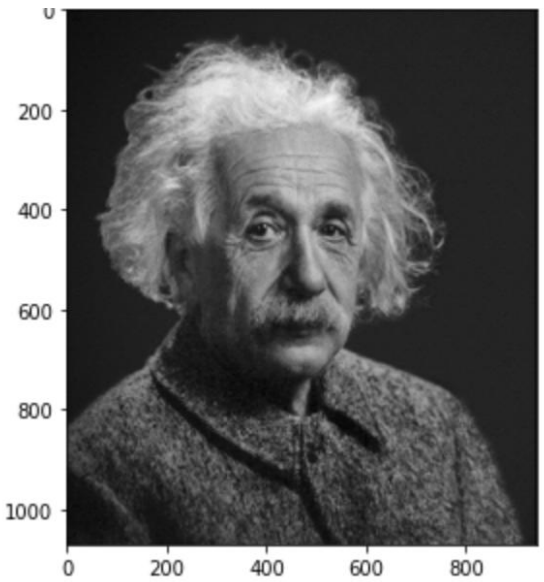
Low rank approximation



$m \times n$ $m \times q$ $q \times r$ $r \times n$

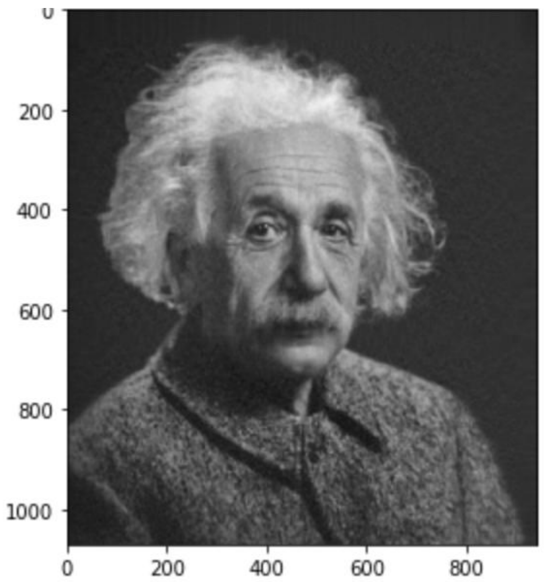
Low rank reconstruction

Here the columns are taken to be features, and rows are observations



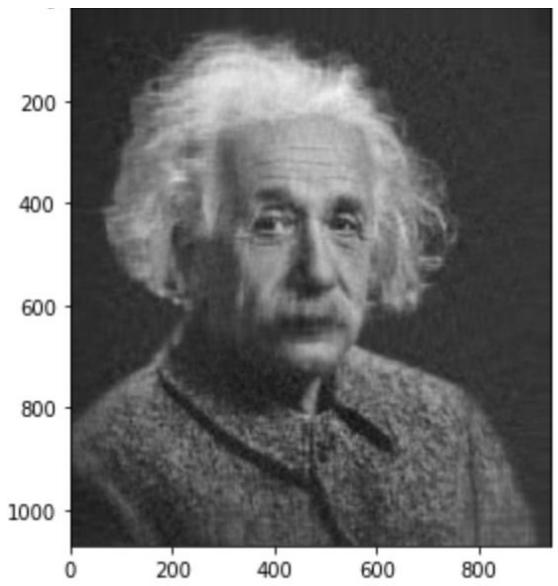
Original

`(1070, 942)`



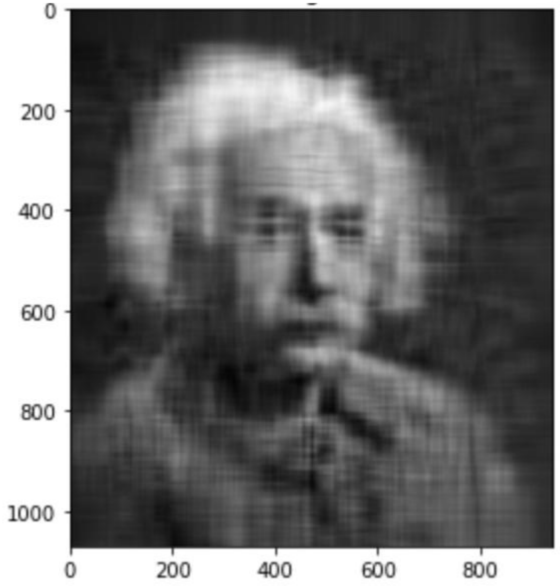
80 vectors

No obvious degradation



40 vectors

Still no obvious degradation

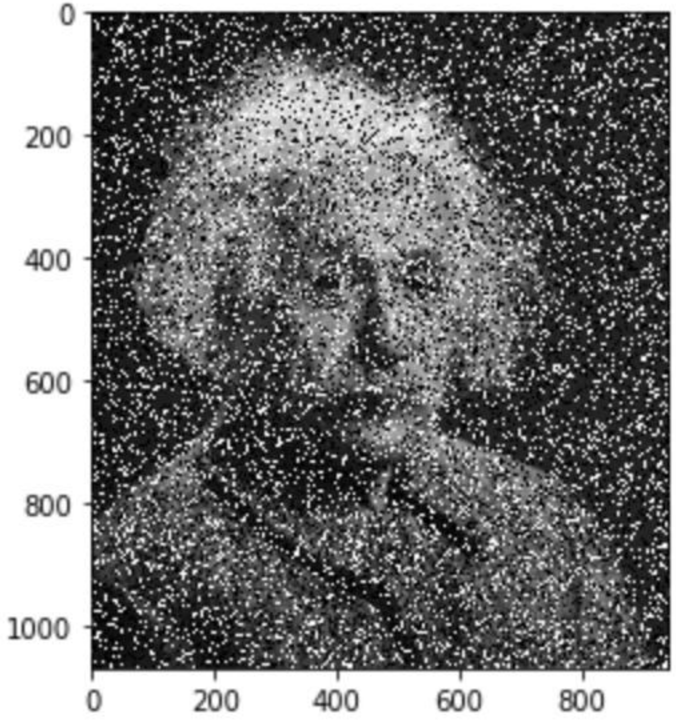


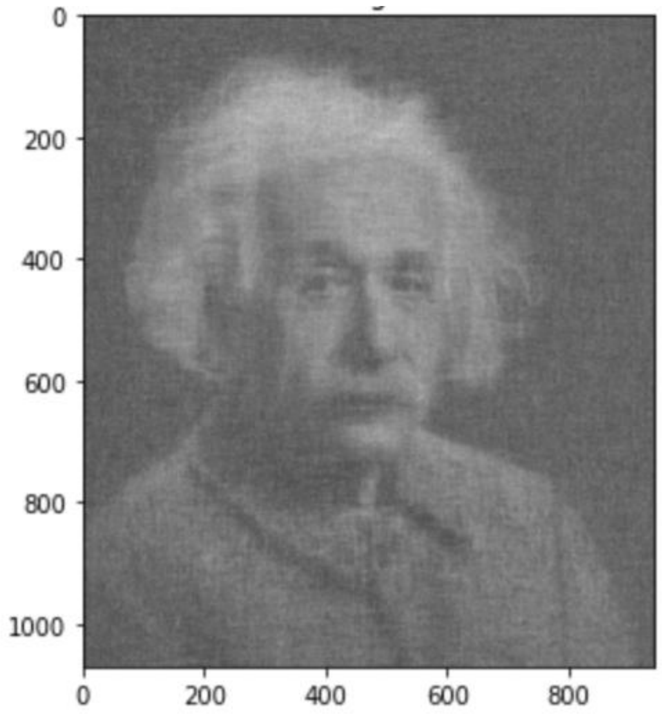
10 vectors

Blurred but can
still recognize is who

Denoising Data

- Low rank decomposition naturally keeps smooth signals and takes off jumpy or “high frequency” signals that don’t explain much variation





100 components

PCA Derivation and Method

PCA motivation

- How do we find directions in the data that preserve the most ***variance***?
- How do we pack ***maximal information*** in just a few dimensions?
- Assumption: we are looking for linear combinations of the original features

Setting up an optimization

- Recall that the variance of a variable is $v^T \Sigma v$
- Here Σ is the data covariance matrix
- Note: we could make variance arbitrarily large by increasing the magnitude of v
- Require solutions that have unit norm $v^T v = 1$

PC1 optimization

$$\max_v v^T \Sigma v$$

subject to $v^T v = 1$

Lagrange Multipliers

- Maximization: $f(x, y)$

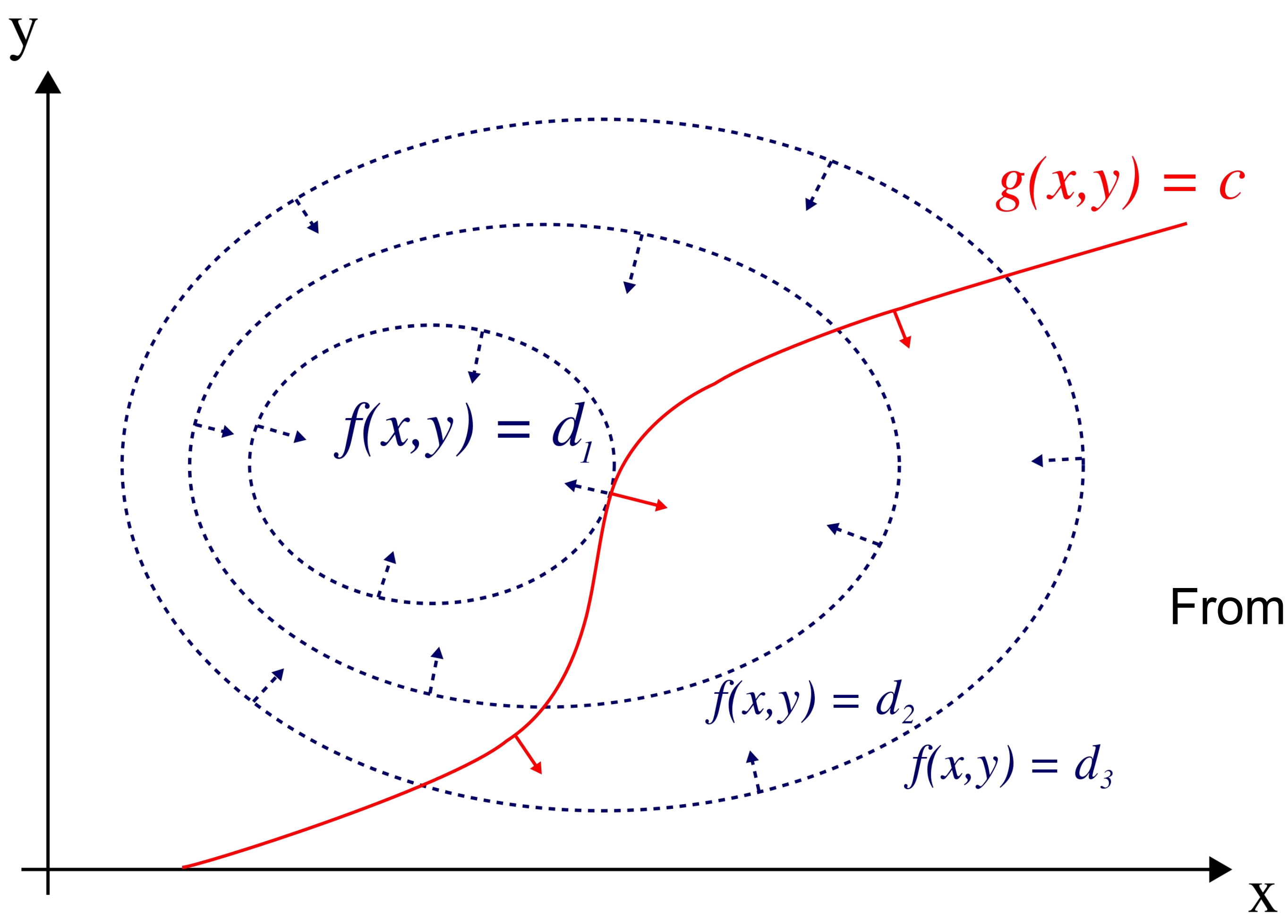
- Constraint: $g(x, y) = 0$

- Lagrangian function :

$$L(x, y, \lambda) = f(x, y) - \lambda g(x, y)$$

.

The **Lagrange multiplier theorem** says that at any local maxima $f(x, y)$ under the equality constraints, the gradient of $f(x, y)$ can be expressed as a linear combination of the gradient of the constraints (only $g(x, y)$ here) with the multiplier(s) λ being the coefficients.



From wikipedia

At the optimum the gradients of the multiplier and function are parallel to each other (they are tangent to each other)

Modified Optimization

PC1 optimization

$$\max_{v} L(v,\ \lambda) = v^T \Sigma v - \lambda(v^T v - 1)$$

To solve: Differentiate and set derivative equal to 0

$$\frac{d}{dv}(v^T \Sigma v - \lambda(v^T v - 1)) = 0$$

$$2\Sigma v - 2\lambda v = 0$$

$\Sigma v = \lambda v$ → v is the eigenvector with largest eigenvalue !

$$v = u_1$$

Other PCs

- To obtain the next vector:
- We maximize the same optimization, but with the constraint that it is orthogonal to u_1
- This gives us the second 2^{nd} largest eigenvector u_2
- Recall that eigenvectors are orthogonal to each other
- This means that the PC components will not contain redundant information, and will be uncorrelated

PCA Method

1. Compute the data covariance matrix Σ

2. Eigendecompose the matrix $\Sigma = U \Lambda U^{-1}$

3. The “loadings” are found in U, i.e., U is a weighted linear combination of the other feature dimensions

4. The new projected data “pc components” are $\tilde{X} = XU$

Variance explained

- Suppose u is a principal direction
- Recall the **Rayleigh quotient** gives the eigenvalue corresponding to the eigenvector

$$\lambda = \frac{u^T A u}{u^T u}$$

- Substituting Σ for A gives $\lambda = \frac{u^T \Sigma u}{u^T u} = u^T \Sigma u = var(u)$

- Thus, the *variance explained is equal to the corresponding eigenvalue*!

PCA for Data Visualization

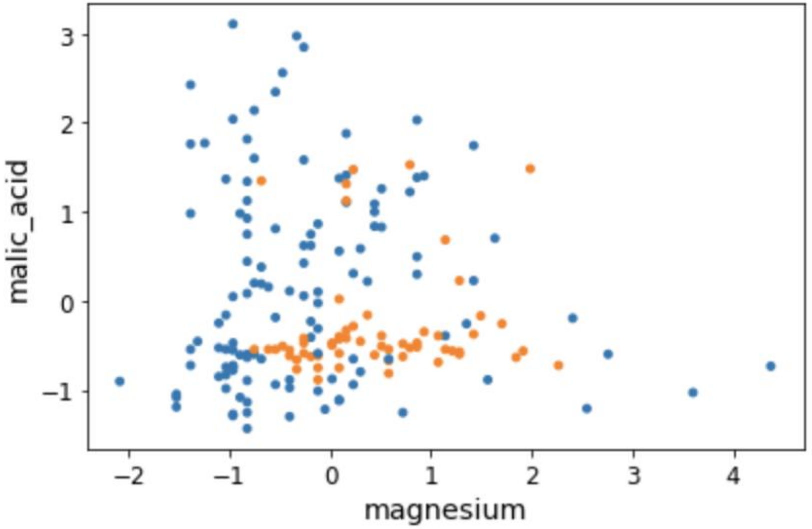
Wines Dataset

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoid_phenols	proanthocyanins	color_intensity
Wine0	1.518613	-0.562250	0.232053	-1.169593	1.913905	0.808997	1.034819	-0.659563	1.224884	0.251717
Wine1	0.246290	-0.499413	-0.827996	-2.490847	0.018145	0.568648	0.733629	-0.820719	-0.544721	-0.293321
Wine2	0.196879	0.021231	1.109334	-0.268738	0.088358	0.808997	1.215533	-0.498407	2.135968	0.269020
Wine3	1.691550	-0.346811	0.487926	-0.809251	0.930918	2.491446	1.466525	-0.981875	1.032155	1.186068
Wine4	0.295700	0.227694	1.840403	0.451946	1.281985	0.808997	0.663351	0.226796	0.401404	-0.319276

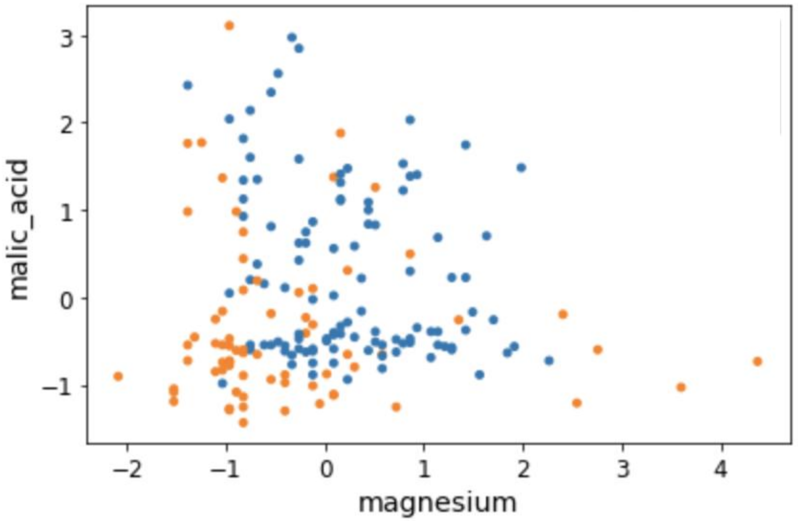
178 wine varieties, 13 features

Randomly selected features

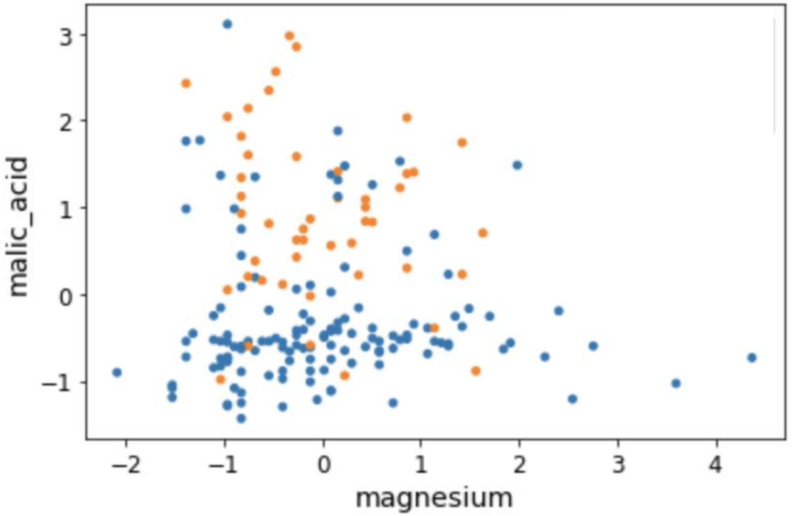
Cultivar 1



Cultivar 2



Cultivar 3

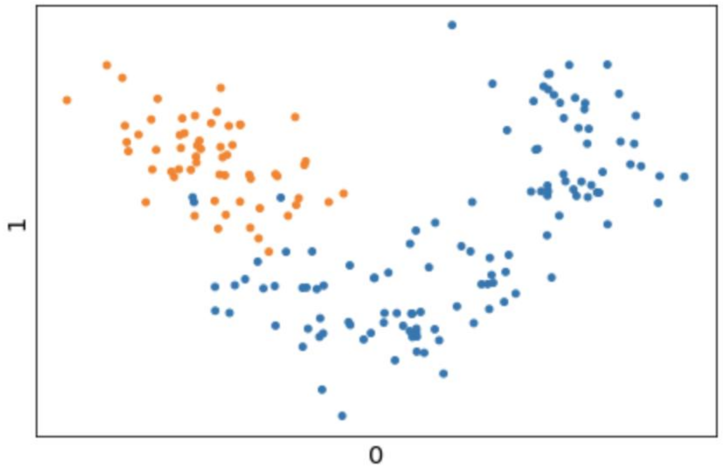


These features do not separate data groupings, nor do they fully reveal heterogeneity

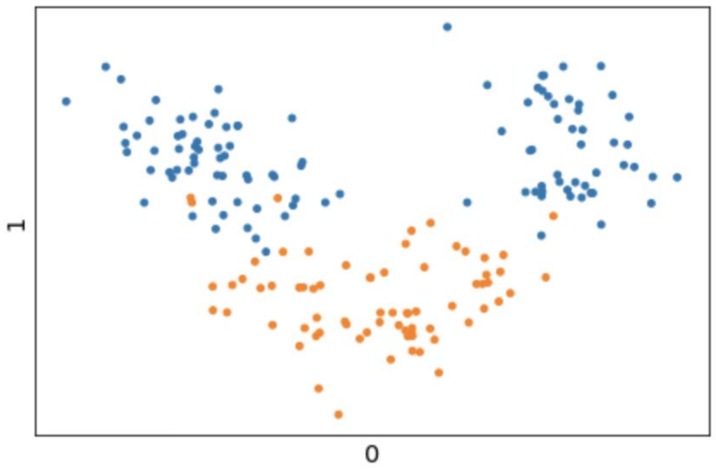
Visualization with PCA

- $\Sigma = U \Lambda U^{-1}$ is the eigendecomposition of Σ
- Recall that the principal axes are U, the eigenvectors
- The principal components of the data are $\tilde{X} = UX$
- Visualizing data via PCA involves drawing a scatterplot using the first two principal components of the data, PC1 vs PC2

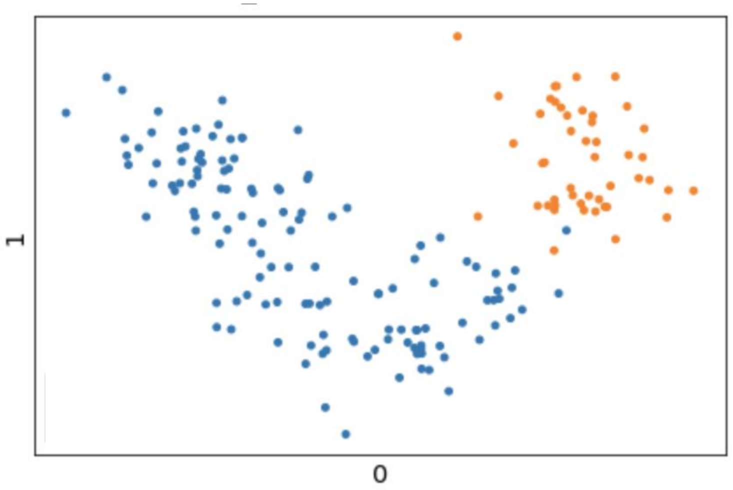
PC1 vs PC2 maximize spread



Cultivar1



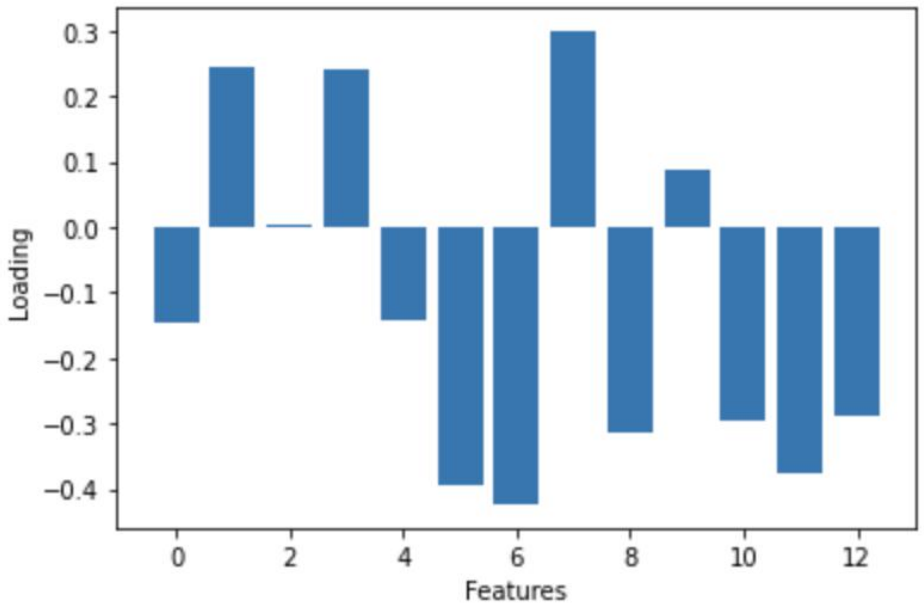
Cultivar2

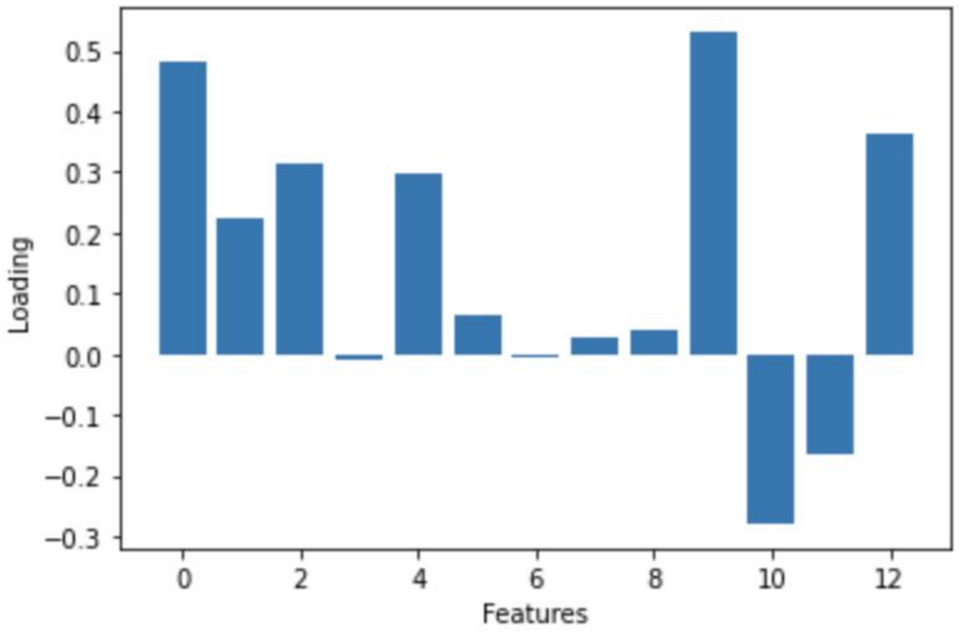


Cultivar3

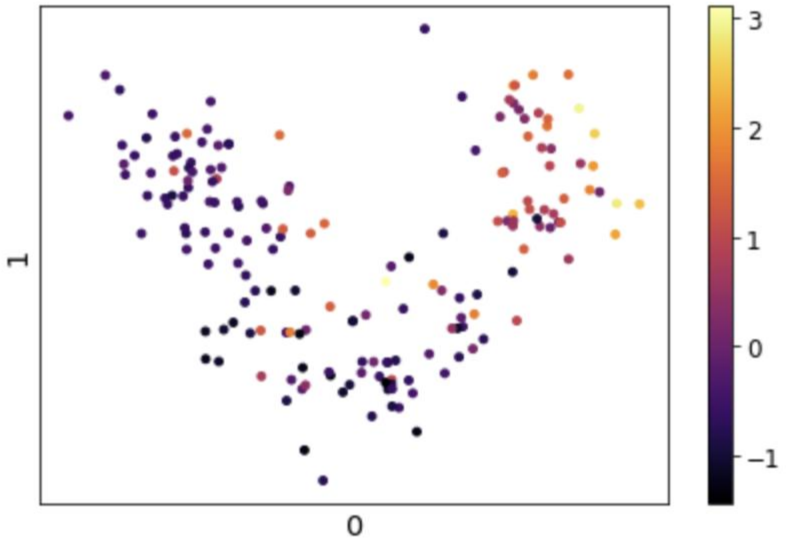
Loadings Explain PCs

	alcohol	malic_acid	ash
Wine0	1.518613	-0.562250	0.232053
Wine1	0.246290	-0.499413	-0.827996
Wine2	0.196879	0.021231	1.109334
Wine3	1.691550	-0.346811	0.487926
Wine4	0.295700	0.227694	1.840403

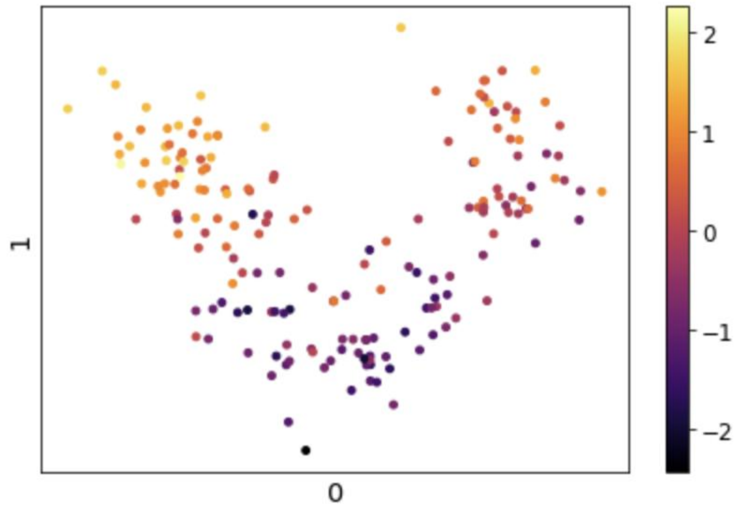




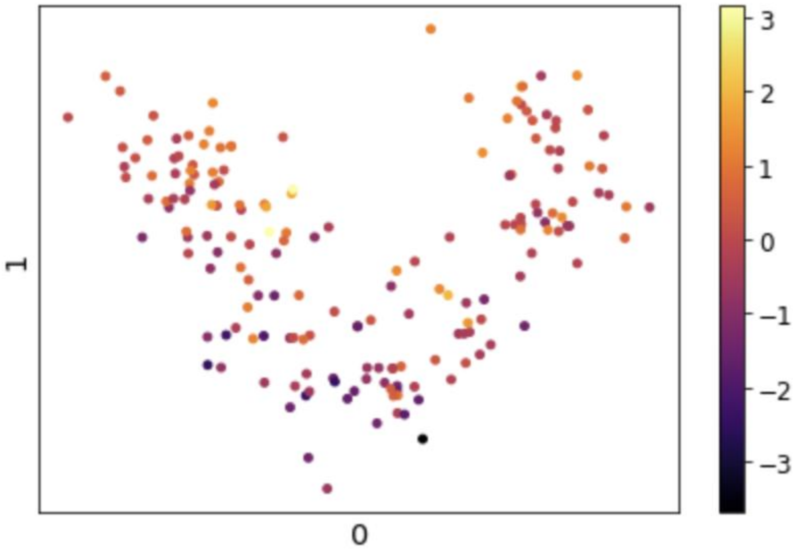
Malic Acid



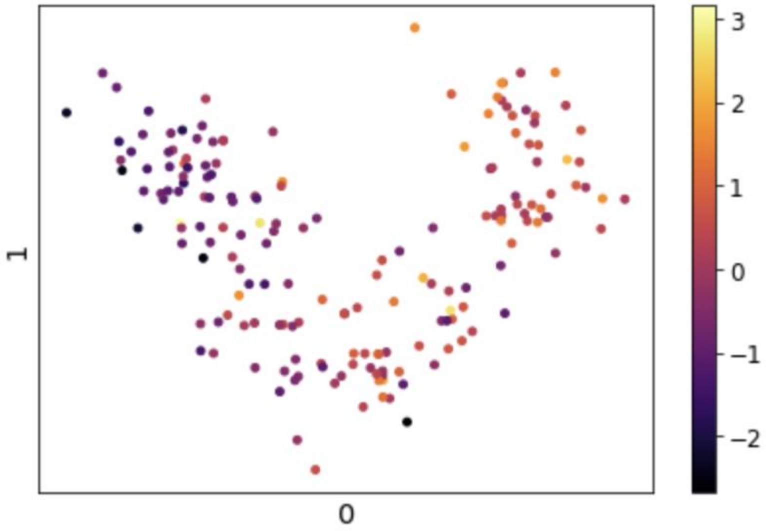
Alcohol



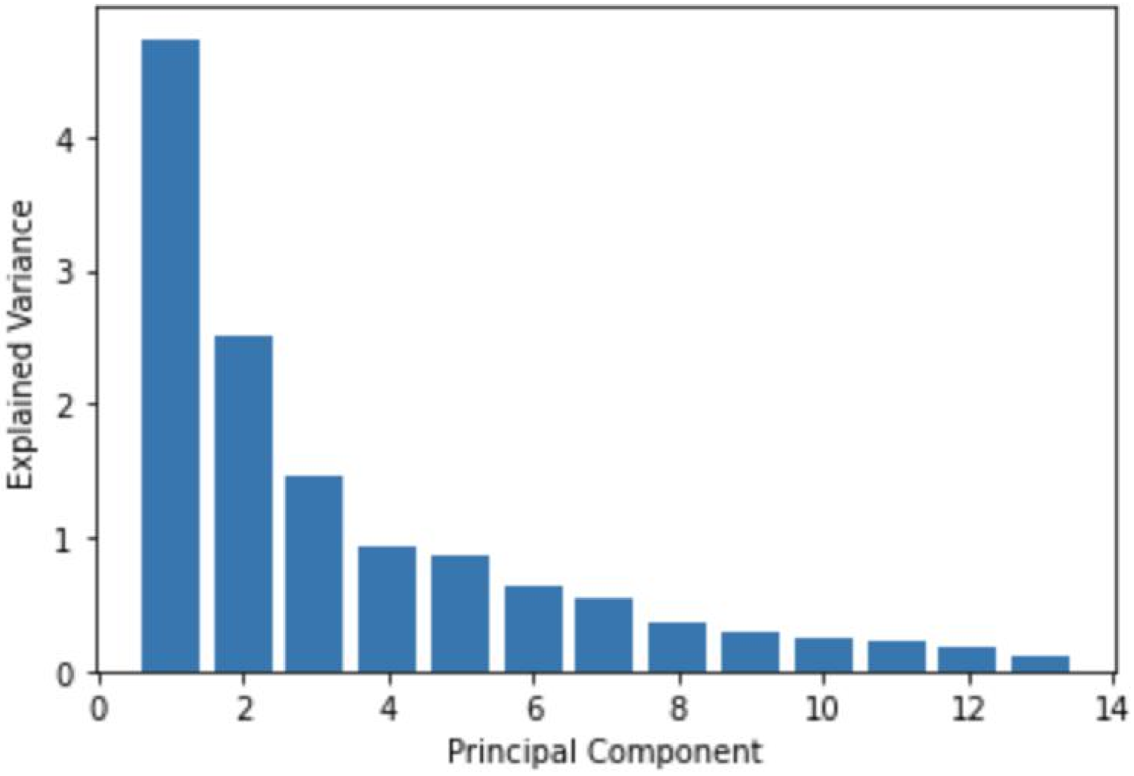
Ash



Alcalinity of Ash



Other PCs: Scree Plot



Explained variance is the eigenvalue

Other pcs may explain other data groups