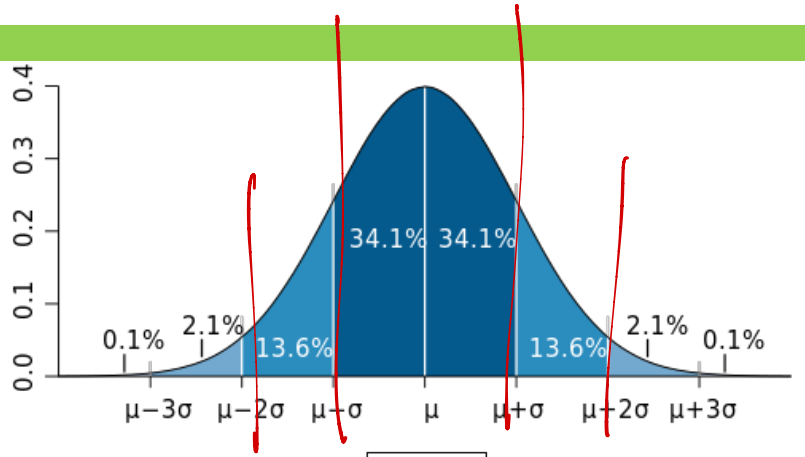
Unsupervised Learning

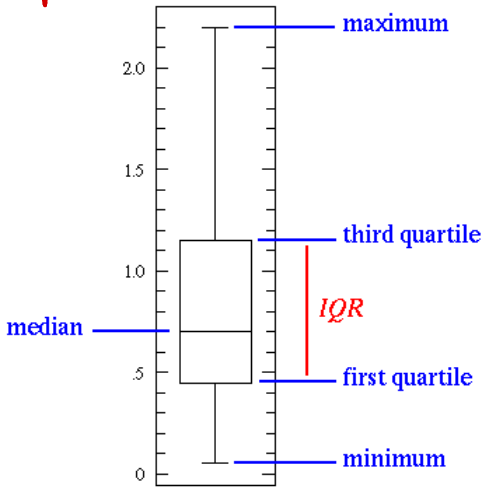
Outliers

Applications

Outliers

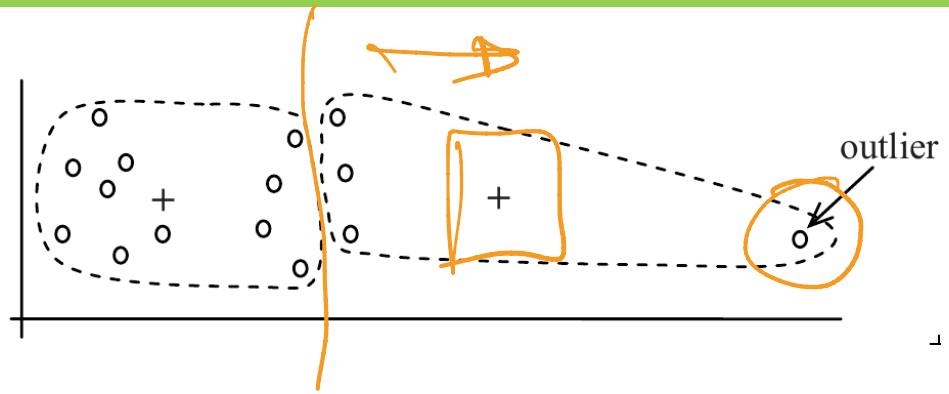
- Values outside the normal range
- Statistically, define in terms of deviation from mean or median
- Gaussian distribution – number of standard deviations from mean
- Box and whisker plots – outer and inner fences based on median, interquartile (IQR) value

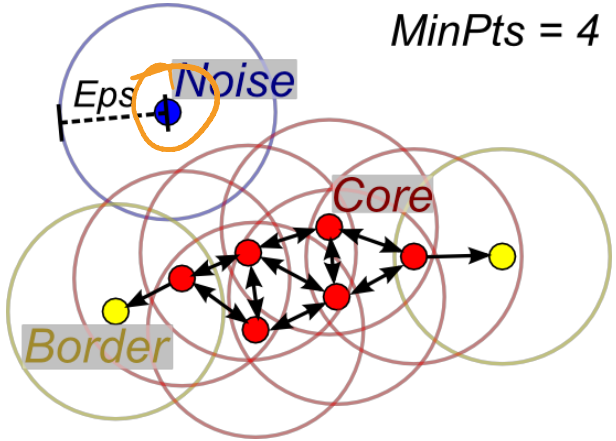




Outliers and clustering

- Outliers are points that lie outside natural clusters
- K Means – far away from all centroids
    - But outliers can distort the clustering process
- Density based clustering – not connected to any core point
    - But density is applied uniformly





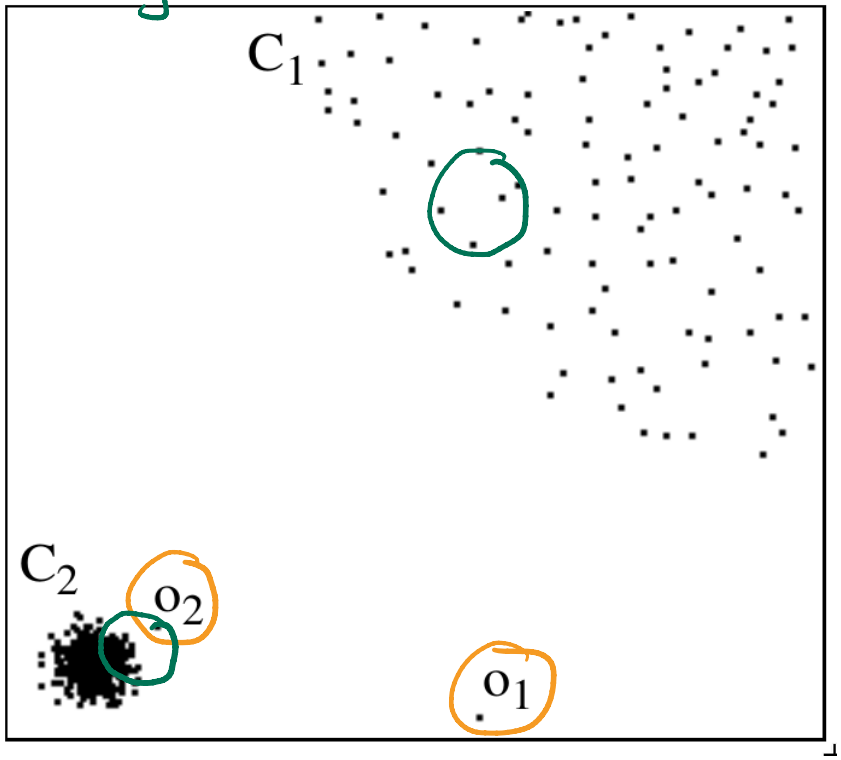
ε

MinPts inside ball

Outliers and density

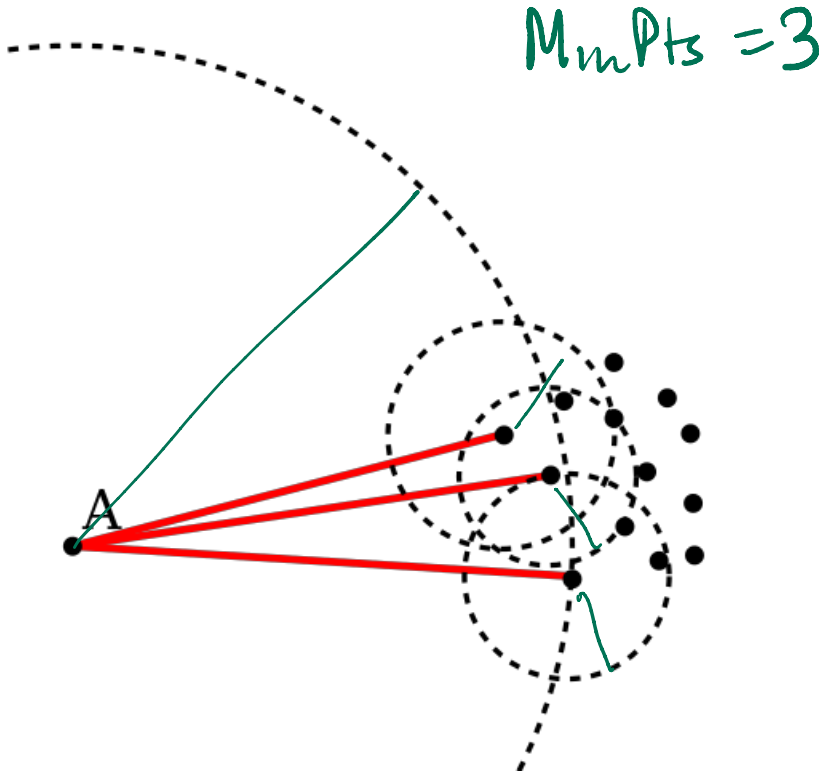
Is the density of a point similar to the density of its neighbours?

- An outlier is less dense than its nearest neighbours
- But difference in density may be local
- A distance metric to eliminate o_2 could make all of C_1 outliers
- C_1 has 400 points, C_2 has 100 points
- Larger distance would make all of C_2 outliers with respect to C_1



Outliers and density

- For clustering, we defined a radius *Eps* and looked for *MinPts* neighbours within that ball
- Instead, fix *MinPts* and find smallest ball with that many neighbours
- Compare *radius(p)* with radius of its neighours
- A is an outlier because its radius is much more than that of its neighbours

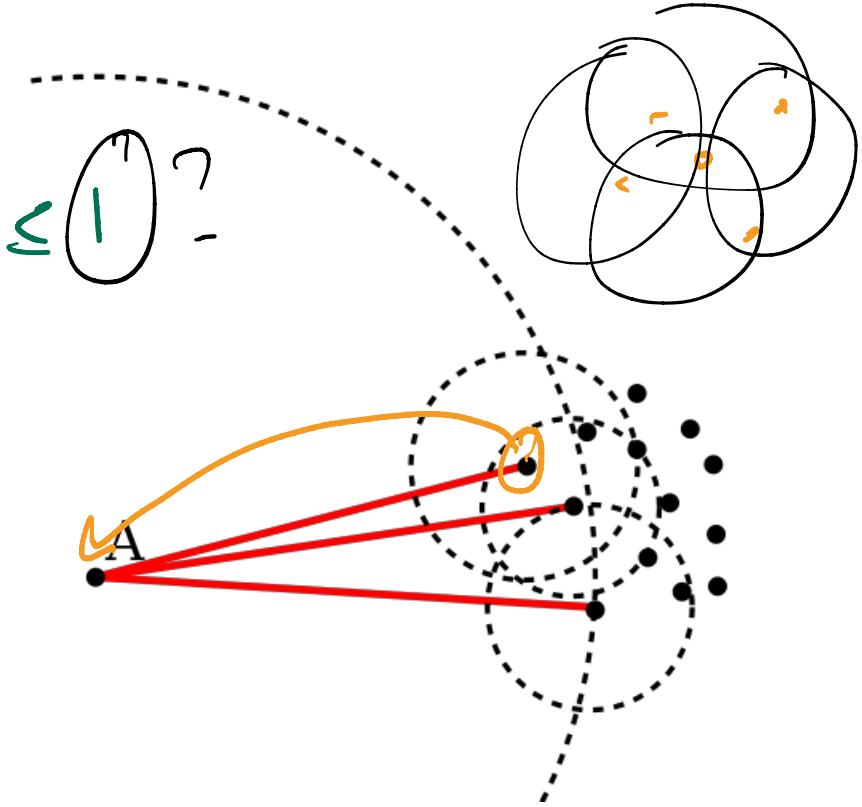


Outliers and density

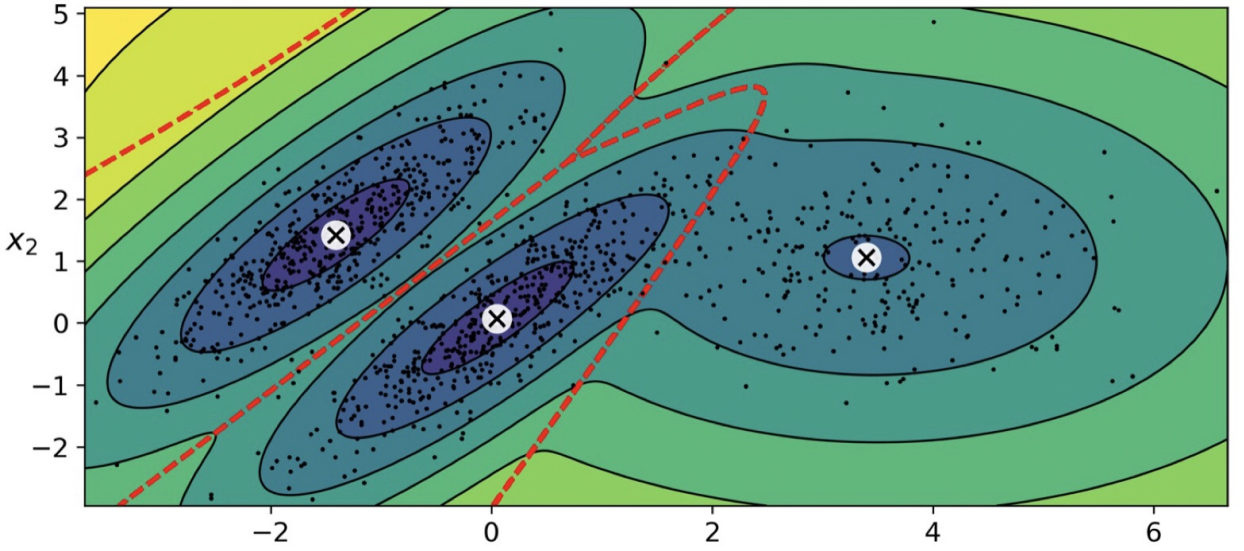
- Local outlier factor *LOF(p)*

$$0 \leq \frac{\text{Mean radius of } MinPts\text{-}neighbours(p)}{radius(p)}$$

- The smaller this ratio, the more likely that *p* is an outlier
- Comparison is local to neighbourhood, so this can deal with different densities across range of data



Clustering using Mixture of Gaussians

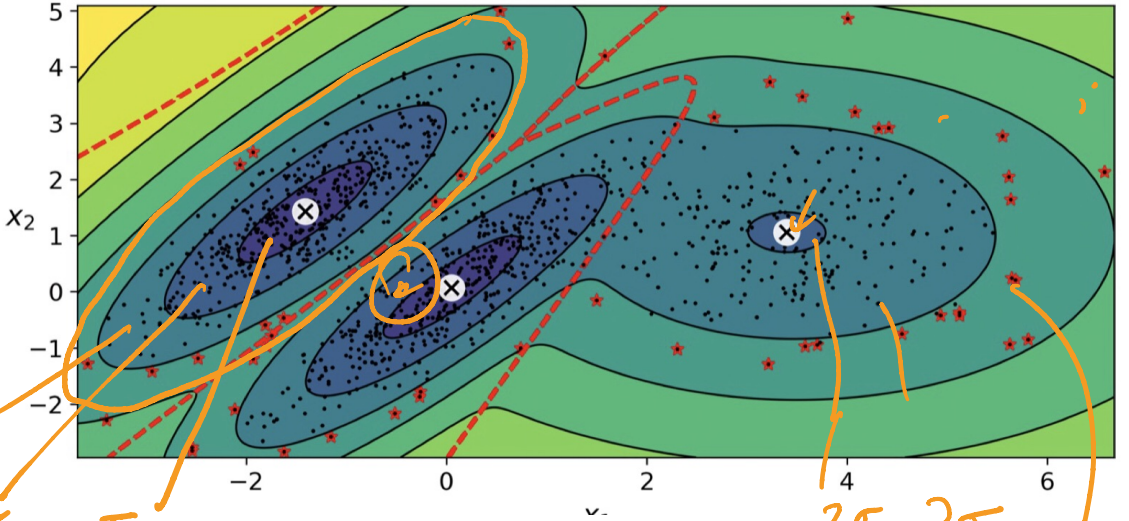


Examples :- Iris dataset

- Tshirt sizes - shoulder, chest, length

XS, S, M, L, XL

Outlier detection using mixture of Gaussians



3σ 2σ σ 2σ 3σ

Numeric data → (μ_1, σ_1), (μ_2, σ_2), (μ_3, σ_3)

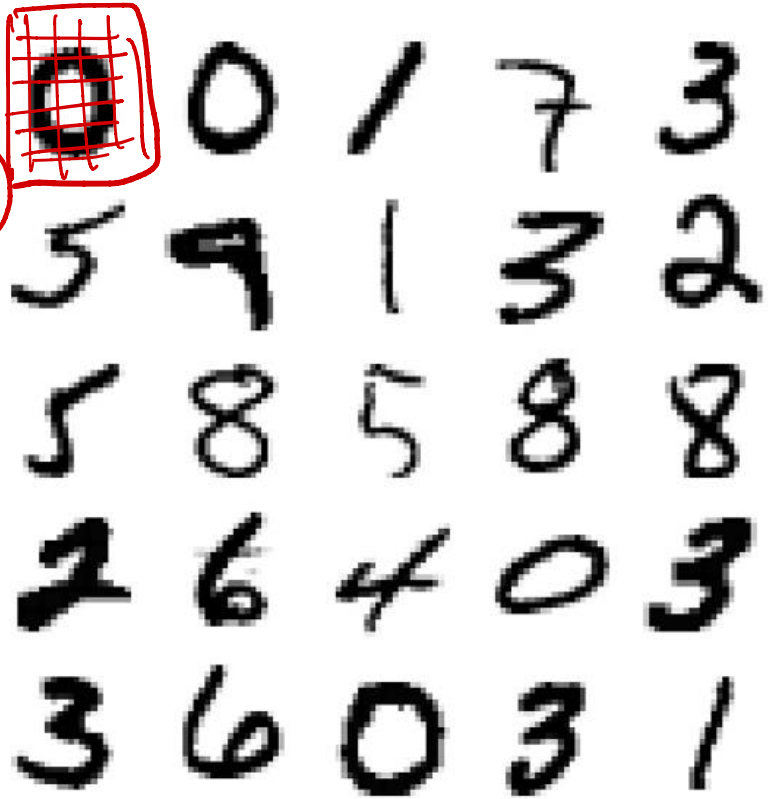
> 3σ for all Gaussians → Outliers

Semi-supervised learning

MNIST

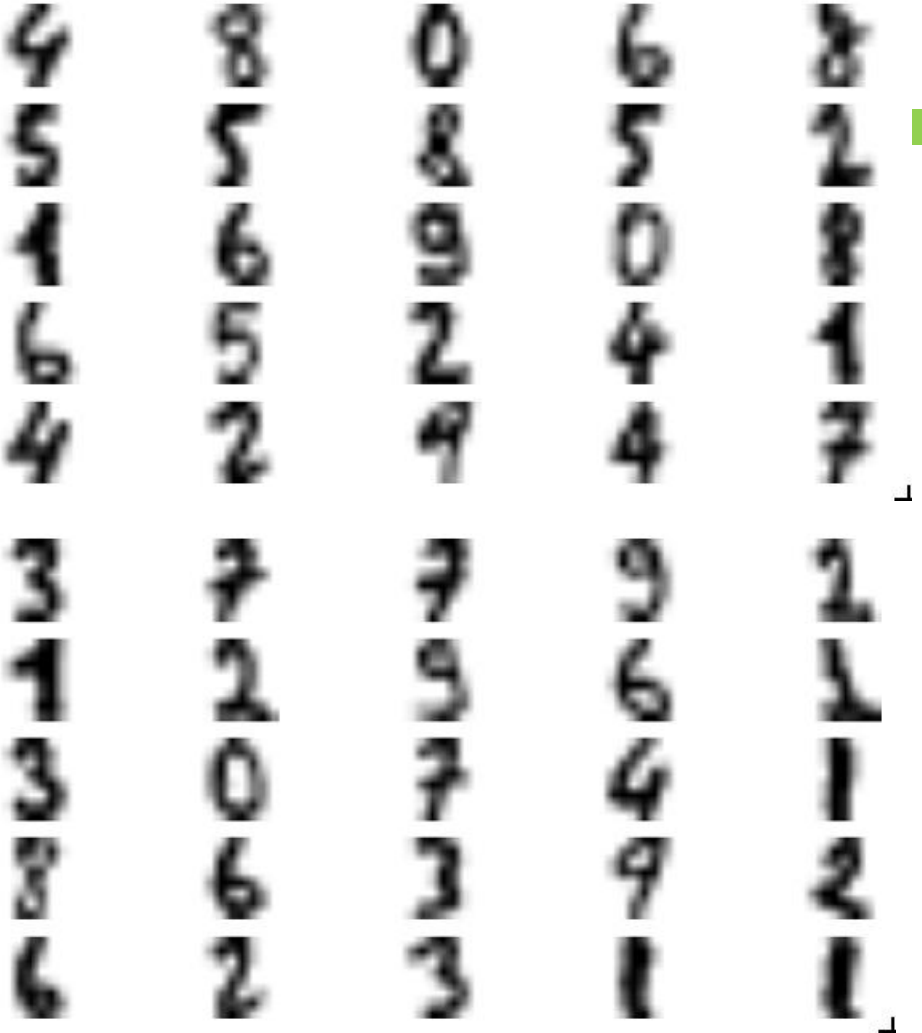
- Labelling training data is a bottleneck of supervised learning
- Handwritten digits 0,1,...,9
  - 1797 images
- Standard logistic regression model has 96.9% accuracy
- Suppose we take 50 random samples as training set
- Logistic regression gives 83.3%

$(x_1 \cdots x_{m \times n})$



Semi-supervised learning

- Instead of 50 random samples, 50 clusters using K means
- Use image nearest to each centroid as training set
    - 50 *representative images*
- Logistic regression accuracy jumps to 92.2%



Semi-supervised learning

- Propagate representative image label to entire cluster
- Logistic regression improves to 93.3%
- Propagage representive image label to only 20% items closest to centroid
- Logistic regression improves to 94%
- Only 50 actual labels used, about 5 per class!

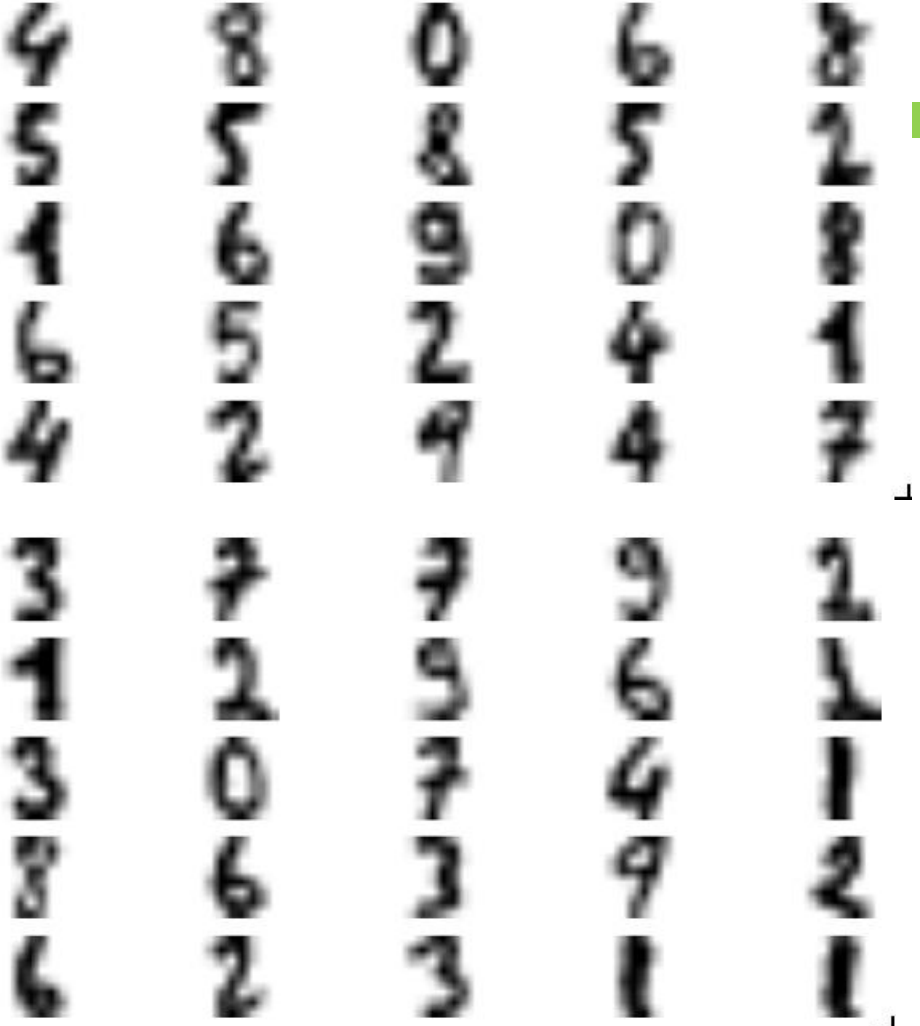


Image segmentation

- An image is a matrix of pixels
- Each pixel has (R,G,B) values
- K means clustering on these values merges colours

Image segmentation

- An image is a matrix of pixels
- Each pixel has (R,G,B) values
- K means clustering on these values merges colours
- With 10 clusters, not much change

Image segmentation

- An image is a matrix of pixels
- Each pixel has (R,G,B) values
- K means clustering on these values merges colours
- With 10 clusters, not much change
- Same with 8

Image segmentation

- An image is a matrix of pixels
- Each pixel has (R,G,B) values
- K means clustering on these values merges colours
- With 10 clusters, not much change
- Same with 8
- At 6 colours, ladybug red goes

Image segmentation

- An image is a matrix of pixels
- Each pixel has (R,G,B) values
- K means clustering on these values merges colours
- With 10 clusters, not much change
- Same with 8
- At 6 colours, ladybug red goes
- 4 colours

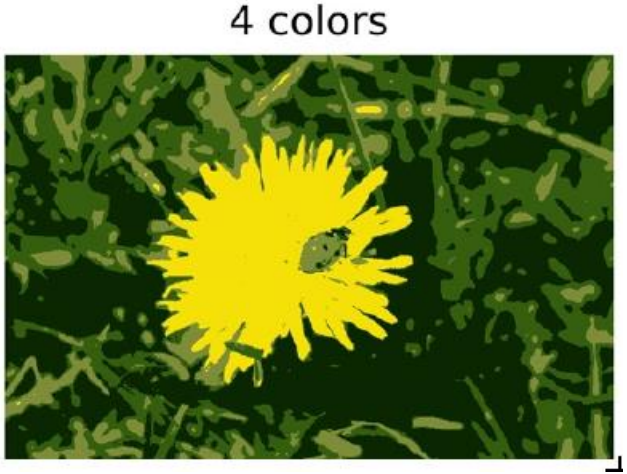
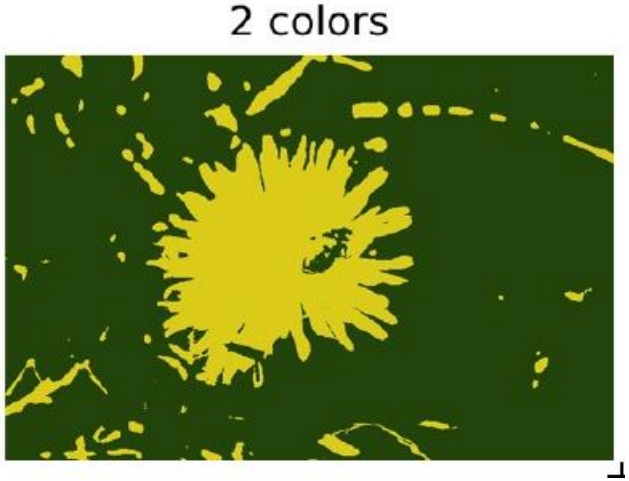


Image segmentation

- An image is a matrix of pixels
- Each pixel has (R,G,B) values
- K means clustering on these values merges colours
- With 10 clusters, not much change
- Same with 8
- At 6 colours, ladybug red goes
- 4 colours
- Finally 2 colours, flower and rest



Dimensionality reduction

Madhavan Mukund

https://www.cmi.ac.in/~madhavan

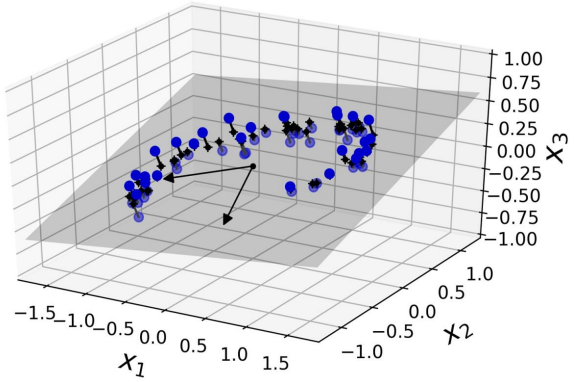
Data Mining and Machine Learning
August–December 2020

Dimensionality reduction

- Principal Component Anaylsis — transform d-dimensional input to k-dimensional input, preserving essential features

Dimensionality reduction

- Principal Component Anaylsis — transform d-dimensional input to k-dimensional input, preserving essential features
- Example: PCA projection of blue points in 3D to black points in 2D

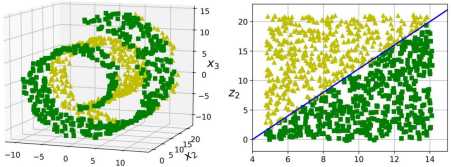


Dimensionality reduction . . .

- Unsupervised preprocessing technique — may make later steps easier, like simplifying classification boundaries

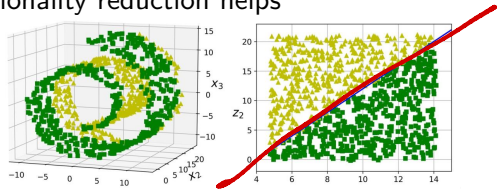
Dimensionality reduction . . .

- Unsupervised preprocessing technique — may make later steps easier, like simplifying classification boundaries
- Swiss roll dataset: dimensionality reduction helps

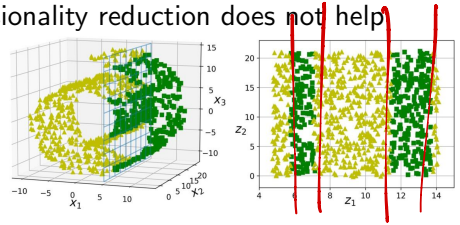


Dimensionality reduction . . .

- Unsupervised preprocessing technique — may make later steps easier, like simplifying classification boundaries
- Swiss roll dataset: dimensionality reduction helps

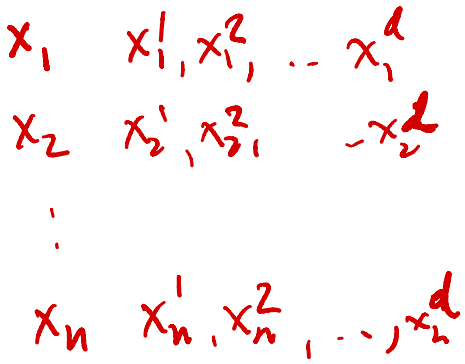


- Swiss roll dataset: dimensionality reduction does not help



Singular Value Decomposition (SVD)

- Input matrix M, dimensions $n \times d$
  - Rows are items, columns are features



Singular Value Decomposition (SVD)

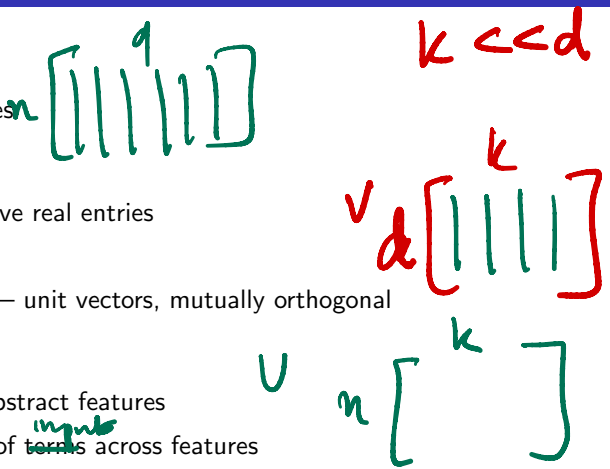
- Input matrix M, dimensions $n \times d$
  - Rows are items, columns are features
- Decompose M as $UDV^\top$
  - D is a $k \times k$ diagonal matrix, positive real entries
  - U is $n \times k$, V is $d \times k$
  - Columns of U, V are orthonormal — unit vectors, mutually orthogonal

Singular Value Decomposition (SVD)

- Input matrix M, dimensions $n \times d$
  - Rows are items, columns are features
- Decompose M as $UDV^\top$
  - D is a $k \times k$ diagonal matrix, positive real entries
  - U is $n \times k$, V is $d \times k$
  - Columns of U, V are orthonormal — unit vectors, mutually orthogonal
- Interpretation
  - Columns of V correspond to new abstract features

Singular Value Decomposition (SVD)

- Input matrix M, dimensions $n \times d$
  - Rows are items, columns are features
- Decompose M as $UDV^\top$
  - D is a $k \times k$ diagonal matrix, positive real entries
  - U is $n \times k$, V is $d \times k$
  - Columns of U, V are orthonormal — unit vectors, mutually orthogonal
- Interpretation
  - Columns of V correspond to new abstract features
  - Rows of U describe decomposition of ~~terms~~ inputs across features



Singular Value Decomposition (SVD)

- Input matrix M, dimensions $n \times d$
  - Rows are items, columns are features
- Decompose M as $UDV^\top$
  - D is a $k \times k$ diagonal matrix, positive real entries
  - U is $n \times k$, V is $d \times k$
  - Columns of U, V are orthonormal — unit vectors, mutually orthogonal
- Interpretation
  - Columns of V correspond to new abstract features
  - Rows of U describe decomposition of terms across features
  - For columns $\boldsymbol{u}_i$ of U and $\boldsymbol{v}_i$ of V, $\boldsymbol{u}_i \cdot \boldsymbol{v}_i^\top$ is an $n \times d$ matrix, like M

$$M = \sum u_i v_i^T$$

Singular Value Decomposition (SVD)

- Input matrix M, dimensions $n \times d$
  - Rows are items, columns are features
- Decompose M as $UDV^\top$
  - D is a $k \times k$ diagonal matrix, positive real entries
  - U is $n \times k$, V is $d \times k$
  - Columns of U, V are orthonormal — unit vectors, mutually orthogonal
- Interpretation
  - Columns of V correspond to new abstract features
  - Rows of U describe decomposition of terms across features
  - For columns $\boldsymbol{u}_i$ of U and $\boldsymbol{v}_i$ of V, $\boldsymbol{u}_i \cdot \boldsymbol{v}_i^\top$ is an $n \times d$ matrix, like M
  - $\boldsymbol{u}_i \cdot \boldsymbol{v}_i^\top$ describes components of rows of M along direction $\boldsymbol{v}_i$

Singular vectors

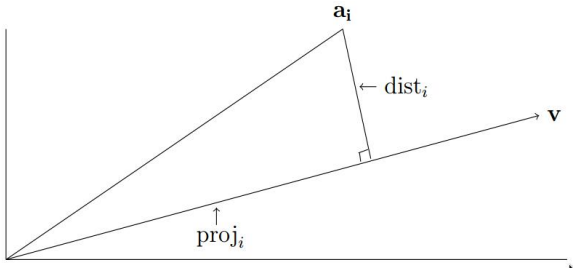
- Unit vectors passing through the origin

Singular vectors

- Unit vectors passing through the origin
- Want to find "best" k singular vectors to represent feature space

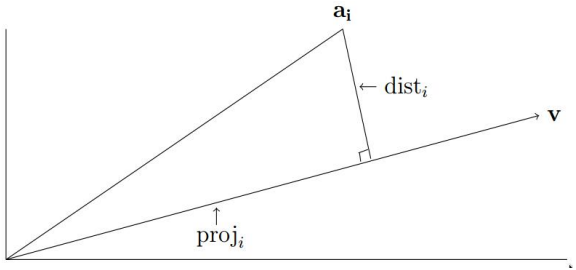
Singular vectors

- Unit vectors passing through the origin
- Want to find "best" k singular vectors to represent feature space
- Suppose we project $\boldsymbol{a}_i = (a_{i1}, a_{i2}, \ldots, a_{id})$ onto $\boldsymbol{v}$ through origin



Singular vectors

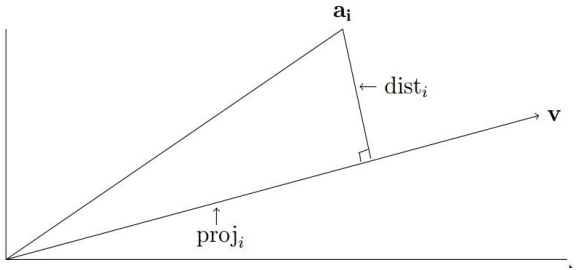
- Unit vectors passing through the origin
- Want to find "best" k singular vectors to represent feature space
- Suppose we project $\boldsymbol{a}_i = (a_{i1}, a_{i2}, \ldots, a_{id})$ onto $\boldsymbol{v}$ through origin



- Minimizing distance of $\boldsymbol{a}_i$ from v is equivalent to maximizing the projection of $\boldsymbol{a}_i$ onto v

Singular vectors

- Unit vectors passing through the origin
- Want to find "best" k singular vectors to represent feature space
- Suppose we project $\boldsymbol{a}_i = (a_{i1}, a_{i2}, \ldots, a_{id})$ onto $\boldsymbol{v}$ through origin



- Minimizing distance of $\boldsymbol{a}_i$ from v is equivalent to maximizing the projection of $\boldsymbol{a}_i$ onto v
- Length of the projection is $\boldsymbol{a}_i \cdot \boldsymbol{v}$

Singular vectors . . .

- Sum of squares of lengths of projections of all rows in M onto $\boldsymbol{v}$ — $|M\boldsymbol{v}|^2$

Singular vectors . . .

- Sum of squares of lengths of projections of all rows in M onto $\boldsymbol{v}$ — $|M\boldsymbol{v}|^2$
- First singular vector — unit vector through origin that maximizes the sum of projections of all rows in M

$$\boldsymbol{v}_1 = \arg\max_{|\boldsymbol{v}|=1} |M\boldsymbol{v}|$$

Singular vectors . . .

- Sum of squares of lengths of projections of all rows in M onto $\boldsymbol{v}$ — $|M\boldsymbol{v}|^2$
- First singular vector — unit vector through origin that maximizes the sum of projections of all rows in M

$$\boldsymbol{v}_1 = \arg\max_{|\boldsymbol{v}|=1} |M\boldsymbol{v}|$$

- Second singular vector — unit vector through origin, perpendicular to $\boldsymbol{v}_1$, that maximizes the sum of projections of all rows in M

$$\boldsymbol{v}_2 = \arg\max_{\boldsymbol{v} \perp \boldsymbol{v}_1;\ |\boldsymbol{v}|=1} |M\boldsymbol{v}|$$

Singular vectors . . .

- Sum of squares of lengths of projections of all rows in M onto $\boldsymbol{v}$ — $|M\boldsymbol{v}|^2$
- First singular vector — unit vector through origin that maximizes the sum of projections of all rows in M

$$\boldsymbol{v}_1 = \arg\max_{|\boldsymbol{v}|=1} |M\boldsymbol{v}|$$

$|Mv_1| > |Mv_2| > |Mv_3|$

- Second singular vector — unit vector through origin, perpendicular to $\boldsymbol{v}_1$, that maximizes the sum of projections of all rows in M

$$\boldsymbol{v}_2 = \arg\max_{\boldsymbol{v}\perp\boldsymbol{v}_1;\ |\boldsymbol{v}|=1} |M\boldsymbol{v}|$$

- Third singular vector — unit vector through origin, perpendicular to $\boldsymbol{v}_1$, $\boldsymbol{v}_2$, that maximizes the sum of projections of all rows in M

$$\boldsymbol{v}_3 = \arg\max_{\boldsymbol{v}\perp\boldsymbol{v}_1,\boldsymbol{v}_2;\ |\boldsymbol{v}|=1} |M\boldsymbol{v}|$$

Singular vectors . . .

- With each singular vector $\boldsymbol{v}_j$, associated singular value is $\sigma_j = |M\boldsymbol{v}_j|$

Singular vectors . . .

- With each singular vector $\boldsymbol{v}_j$, associated singular value is $\sigma_j = |M\boldsymbol{v}_j|$
- Repeat r times till $\max_{\boldsymbol{v} \perp \boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_r;\ |\boldsymbol{v}|=1} |M\boldsymbol{v}| = 0$
  - r turns out to be the rank of M
  - Vectors $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_r\}$ are orthonormal right singular vectors

Singular vectors . . .

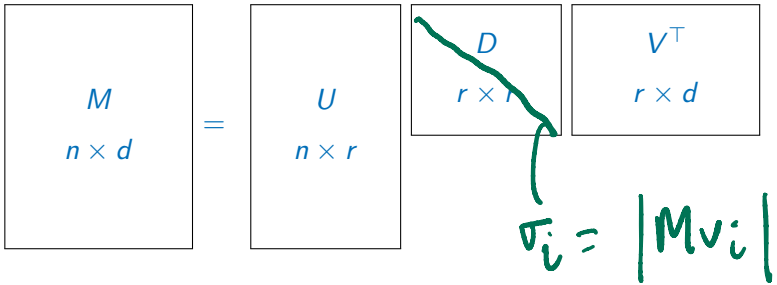
- With each singular vector $\boldsymbol{v}_j$, associated singular value is $\sigma_j = |M\boldsymbol{v}_j|$
- Repeat r times till $\max_{\boldsymbol{v} \perp \boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_r;\ |\boldsymbol{v}|=1} |M\boldsymbol{v}| = 0$
  - r turns out to be the rank of M
  - Vectors $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_r\}$ are orthonormal right singular vectors
- Our greedy strategy provably produces "best-fit" dimension r subspace for M
  - Dimension r subspace that maximizes content of M projected onto it

Singular vectors . . .

- With each singular vector $\boldsymbol{v}_j$, associated singular value is $\sigma_j = |M\boldsymbol{v}_j|$
- Repeat r times till $\max_{\boldsymbol{v} \perp \boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_r;\ |\boldsymbol{v}|=1} |M\boldsymbol{v}| = 0$
  - r turns out to be the rank of M
  - Vectors $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_r\}$ are orthonormal right singular vectors
- Our greedy strategy provably produces "best-fit" dimension r subspace for M
  - Dimension r subspace that maximizes content of M projected onto it
- Corresponding left singular vectors are given by $\boldsymbol{u}_i = \frac{1}{\sigma_i} M\boldsymbol{v}_i$
- Can show that $\{\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_r\}$ are also orthonormal

Singular Value Decomposition

- M, dimension $n \times d$, of rank r uniquely decomposes as $M = UDV^\top$
  - $V = [\boldsymbol{v}_1 \ \boldsymbol{v}_2 \ \cdots \ \boldsymbol{v}_r]$ are the right singular vectors
  - D is a diagonal matrix with $D[i,i] = \sigma_i$, the singular values
  - $U = [\boldsymbol{u}_1 \ \boldsymbol{u}_2 \ \cdots \ \boldsymbol{u}_r]$ are the left singular vectors



Rank-k approximation

- M has rank r, SVD gives rank r decomposition

Rank-k approximation

- M has rank r, SVD gives rank r decomposition
- Singular values are non-increasing — $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$

Rank-k approximation

- M has rank r, SVD gives rank r decomposition
- Singular values are non-increasing — $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$
- Suppose we retain only k largest ones
- We have
  - Matrix of first k right singular vectors $V_k = [\boldsymbol{v}_1 \ \boldsymbol{v}_2 \ \cdots \ \boldsymbol{v}_k]$,
  - Corresponding singular values $\sigma_1, \sigma_2, \ldots, \sigma_k$
  - Matrix of k left singular vectors $U_k = [\boldsymbol{u}_1 \ \boldsymbol{u}_2 \ \cdots \ \boldsymbol{u}_k]$

Rank-k approximation

- M has rank r, SVD gives rank r decomposition
- Singular values are non-increasing — $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$
- Suppose we retain only k largest ones
- We have
  - Matrix of first k right singular vectors $V_k = [\boldsymbol{v}_1 \ \boldsymbol{v}_2 \ \cdots \ \boldsymbol{v}_k]$,
  - Corresponding singular values $\sigma_1, \sigma_2, \ldots, \sigma_k$
  - Matrix of k left singular vectors $U_k = [\boldsymbol{u}_1 \ \boldsymbol{u}_2 \ \cdots \ \boldsymbol{u}_k]$
- Let D_k be the $k \times k$ diagonal matrix with entries $\sigma_1, \sigma_2, \ldots, \sigma_k$
- Then $U_k D_k V_k^\top$ is the best fit rank-k approximation of M

Rank-k approximation

- M has rank r, SVD gives rank r decomposition
- Singular values are non-increasing — $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$
- Suppose we retain only k largest ones
- We have
  - Matrix of first k right singular vectors $V_k = [\boldsymbol{v}_1 \ \boldsymbol{v}_2 \ \cdots \ \boldsymbol{v}_k]$,
  - Corresponding singular values $\sigma_1, \sigma_2, \ldots, \sigma_k$
  - Matrix of k left singular vectors $U_k = [\boldsymbol{u}_1 \ \boldsymbol{u}_2 \ \cdots \ \boldsymbol{u}_k]$
- Let D_k be the $k \times k$ diagonal matrix with entries $\sigma_1, \sigma_2, \ldots, \sigma_k$
- Then $U_k D_k V_k^\top$ is the best fit rank-k approximation of M
- In other words, by truncating the SVD, we can focus on k most significant features implicit in M

Summary

- Singular Value Decomposition (SVD) finds best fit k-dimensional subspace for any matrix M
- Principal Component Analysis uses SVD for dimensionality reduction
- Unsupervised technique — often helps simplify the problem, but may not
- SVD/PCA can only compress features that have a linear relationship
- More general techniques based on neural networks — autoencoders