Introduction to Unsupervised Learning

Types of Machine Learning

Supervised	data points have known outcome
Unsupervised	data points have unknown outcome

Types of Machine Learning

Supervised — data points have known outcome

Unsupervised — data points have unknown outcome

Types of Unsupervised Learning

Clustering

identify unknown structure in data

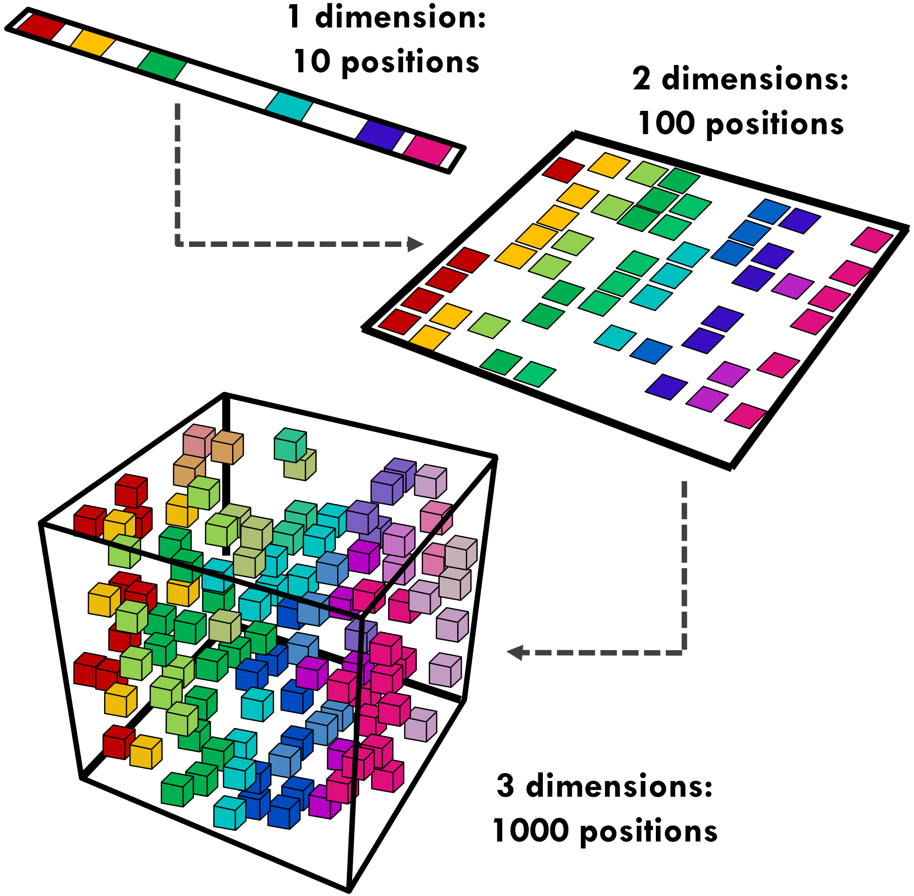
Types of Unsupervised Learning

Clustering	identify unknown structure in data
Dimensionality Reduction	use structural characteristics to simplify data

Dimensionality Reduction

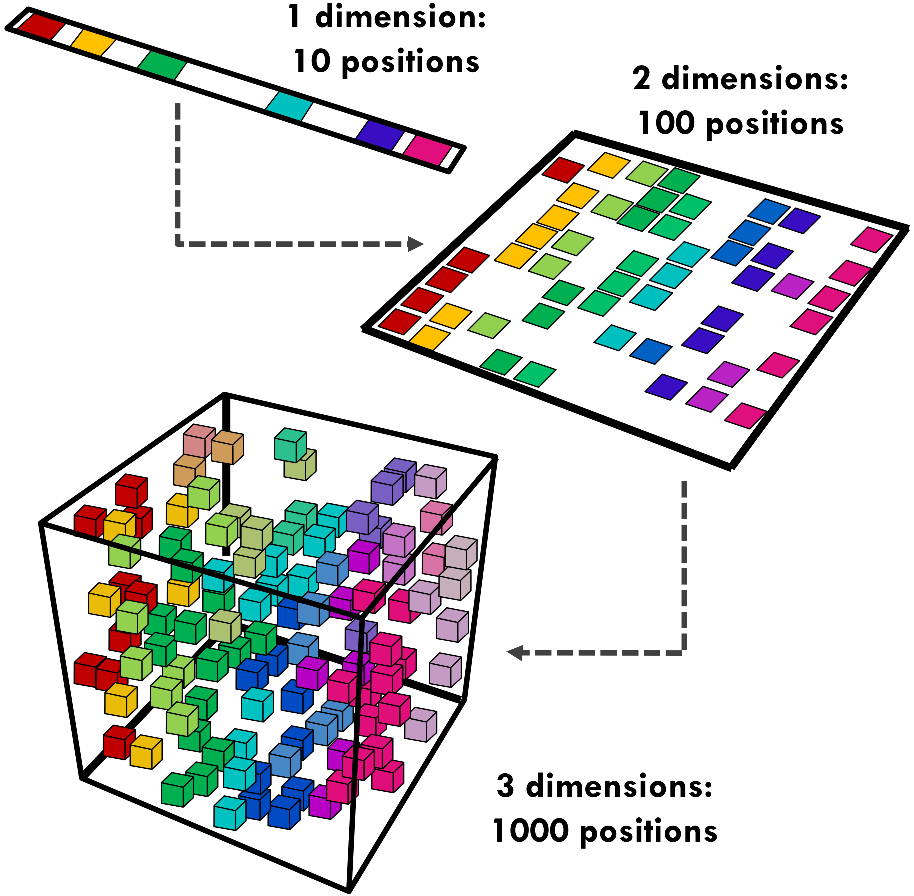
Curse of Dimensionality

- Theoretically, increasing features should improve performance



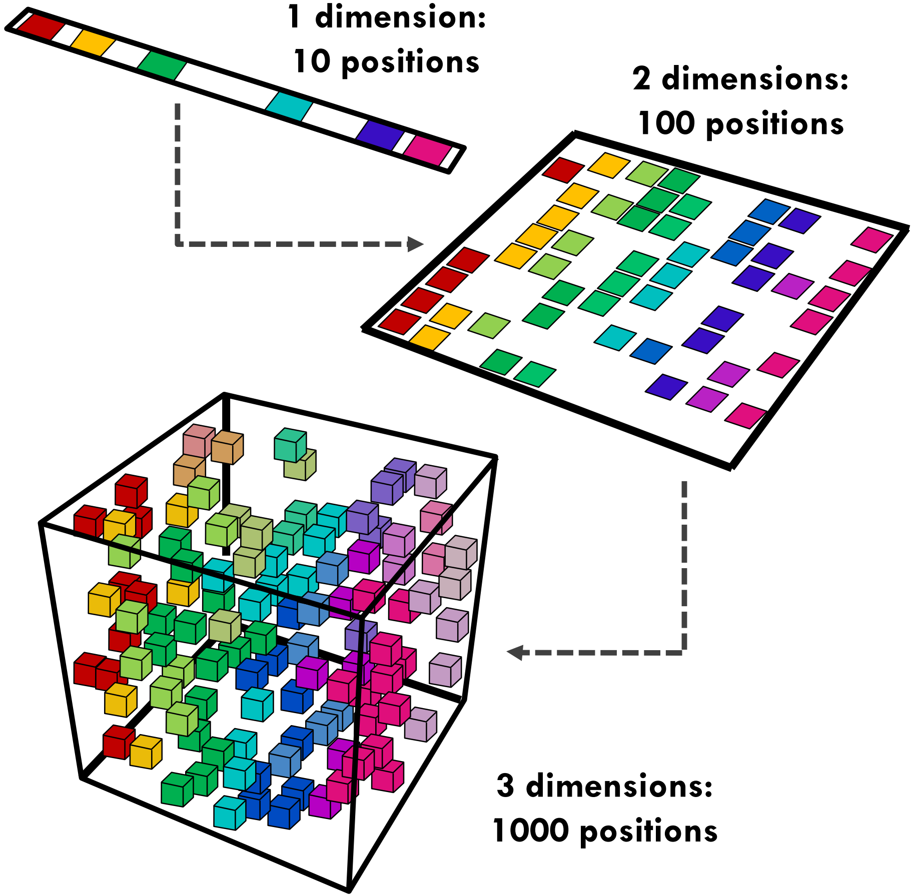
Curse of Dimensionality

- Theoretically, increasing features should improve performance
- In practice, too many features leads to worse performance



Curse of Dimensionality

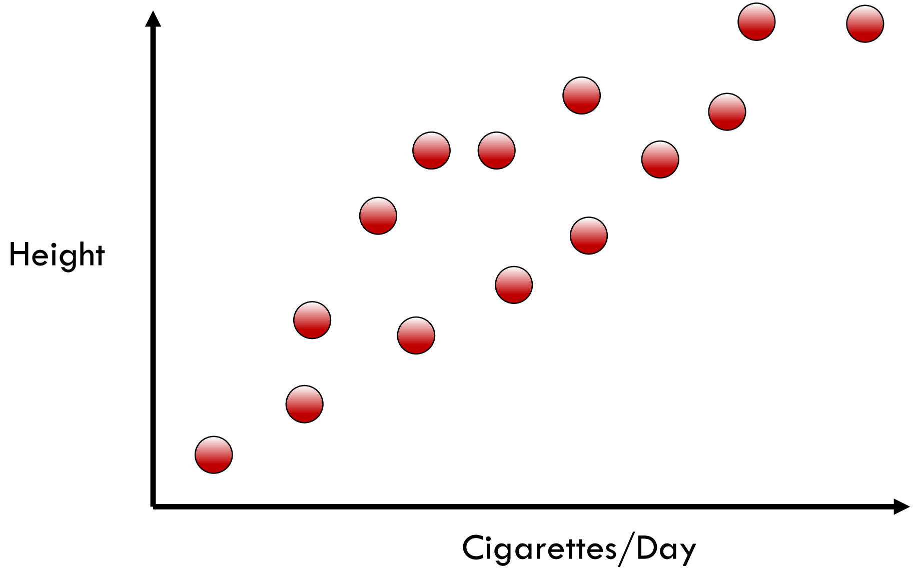
- Theoretically, increasing features should improve performance
- In practice, too many features leads to worse performance
- Number of training examples required increases exponentially with dimensionality



Solution: Dimensionality Reduction

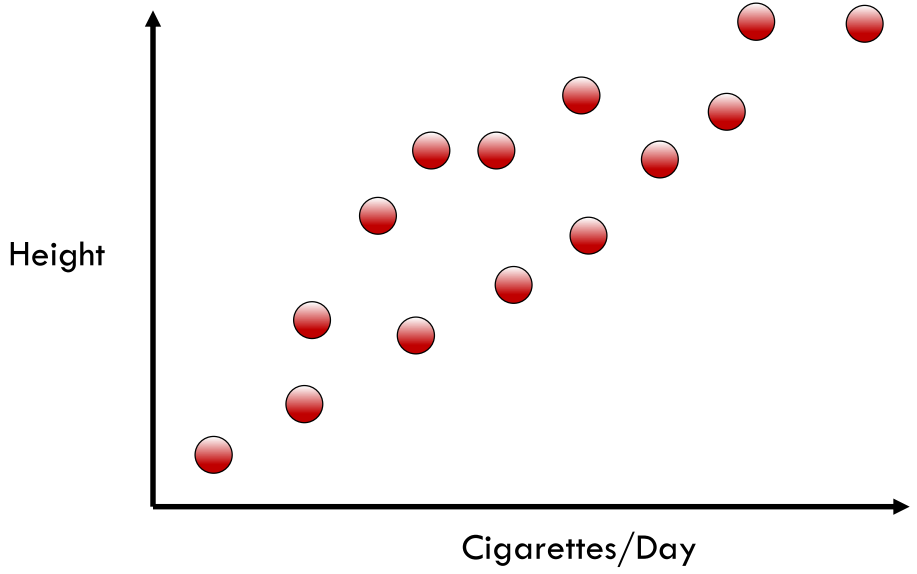
- Data can be represented by fewer dimensions (features)

- Reduce dimensionality by



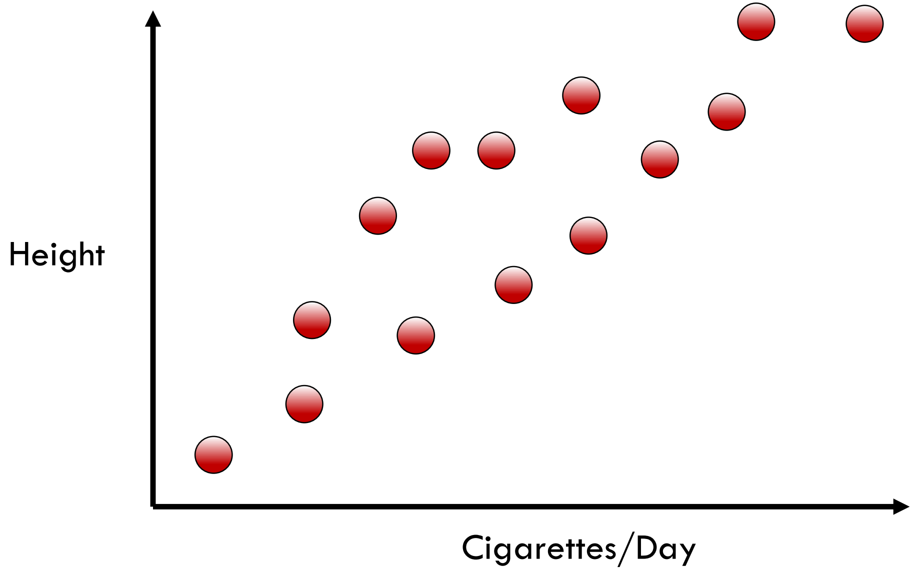
Solution: Dimensionality Reduction

- Data can be represented by fewer dimensions (features)
- Reduce dimensionality by selecting subset (feature elimination)
- Combine with linear and non-



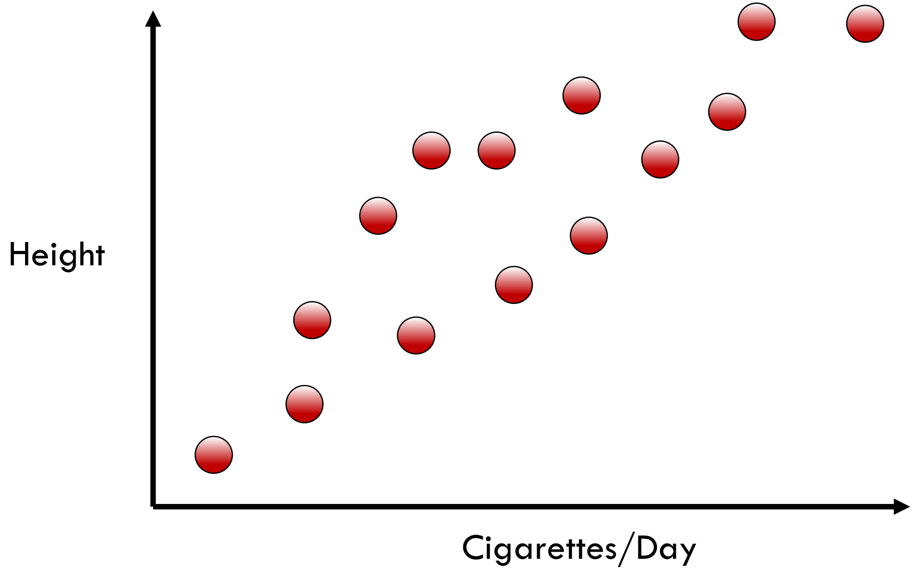
Solution: Dimensionality Reduction

- Data can be represented by fewer dimensions (features)
- Reduce dimensionality by selecting subset (feature elimination)
- Combine with linear and non-linear transformations



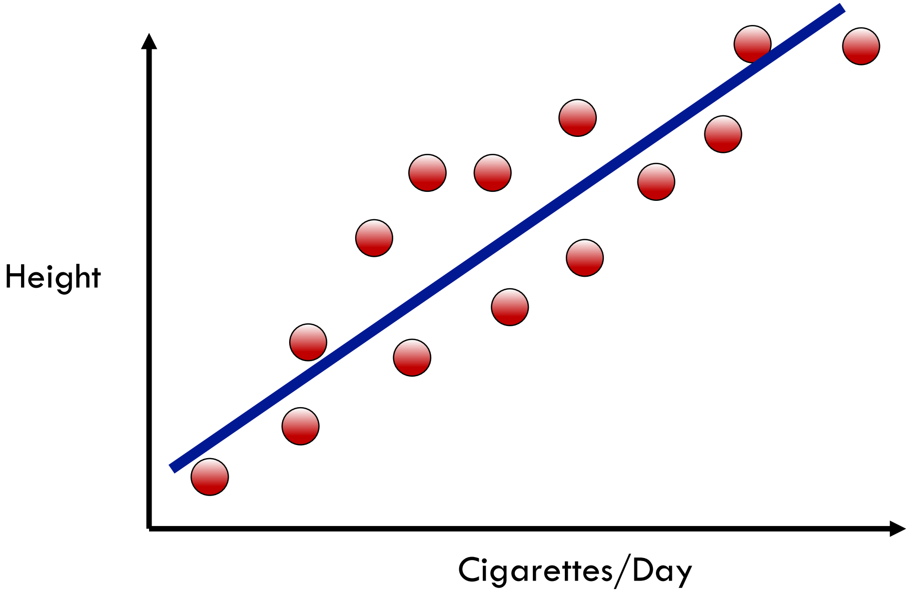
Solution: Dimensionality Reduction

- Two features: height and cigarettes per day
- Both features increase together (correlated)
- Can we reduce number of features to one?



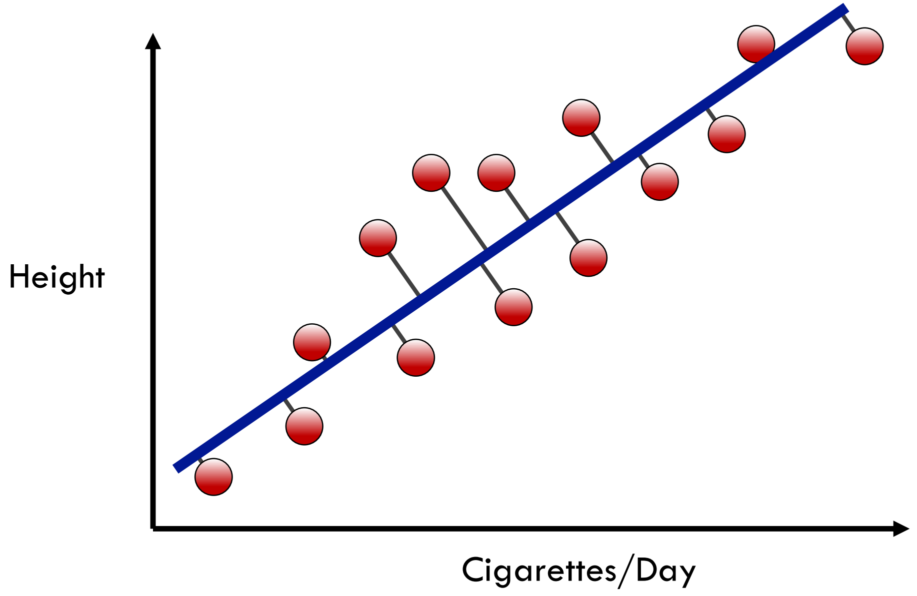
Solution: Dimensionality Reduction

- Two features: height and cigarettes per day
- Both features increase together (correlated)
- Can we reduce number of features to one?



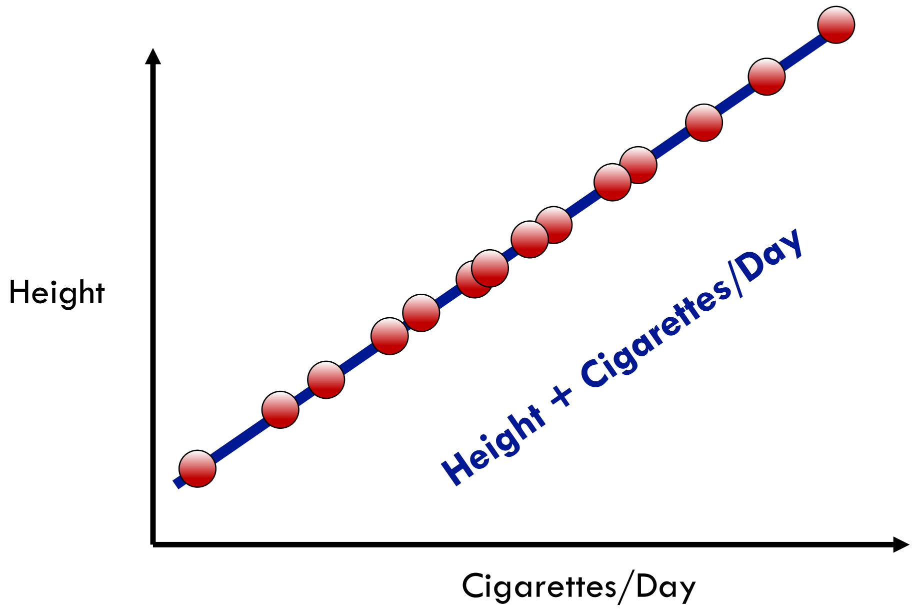
Solution: Dimensionality Reduction

- Two features: height and cigarettes per day
- Both features increase together (correlated)
- Can we reduce number of features to one?



Solution: Dimensionality Reduction

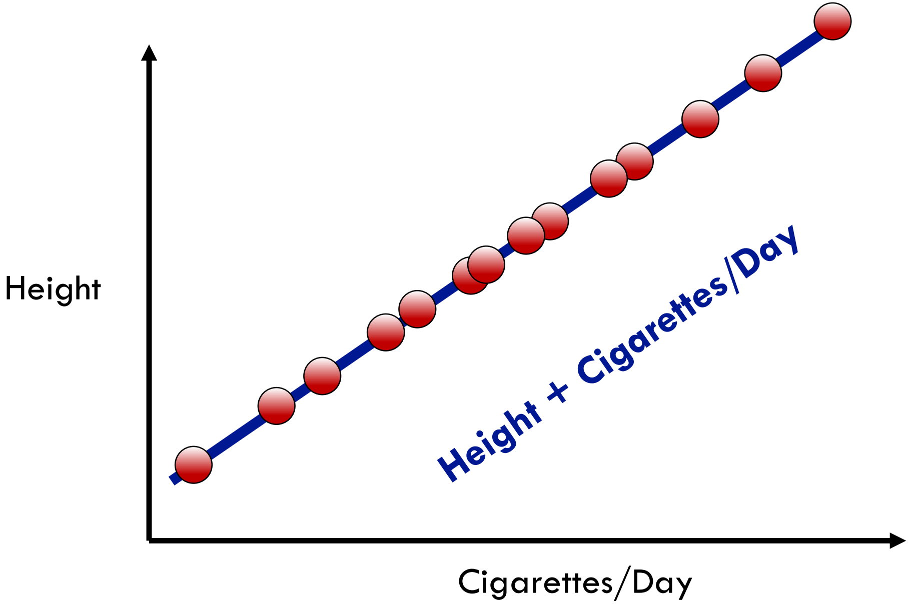
- Two features: height and cigarettes per day
- Both features increase together (correlated)
- Can we reduce number of features to one?



Solution: Dimensionality Reduction

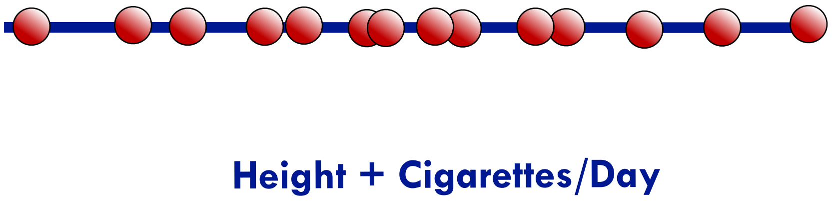
- Create single feature that is combination of height and cigarettes

- This is Principal Component Analysis (PCA)



Solution: Dimensionality Reduction

- Create single feature that is combination of height and cigarettes
- This is Principal Component Analysis (PCA)



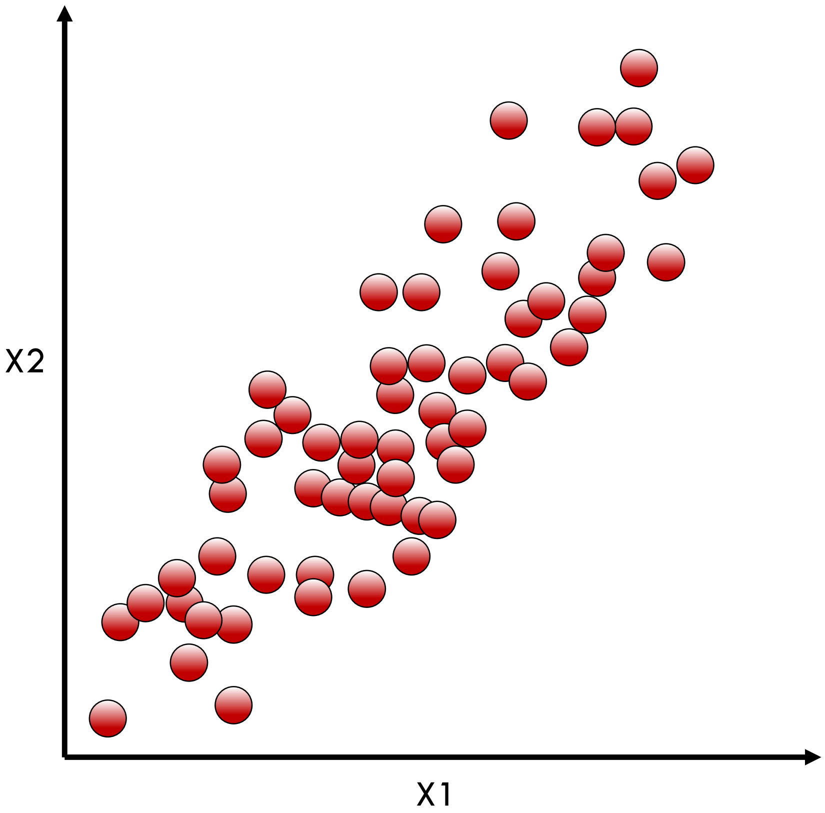
Dimensionality Reduction

Given an N-dimensional data set (x), find a $N \times K$ matrix (U):

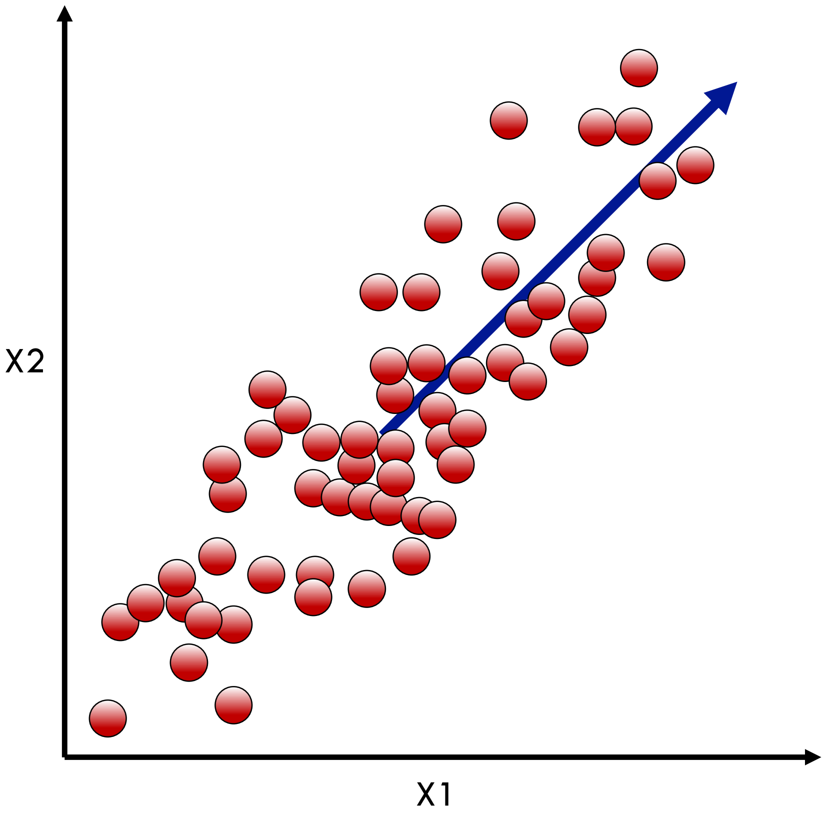
$y = U^T x$, where y has K dimensions and $K < N$

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix} \xrightarrow{U^T} y = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_k \end{bmatrix} (K < N)$$

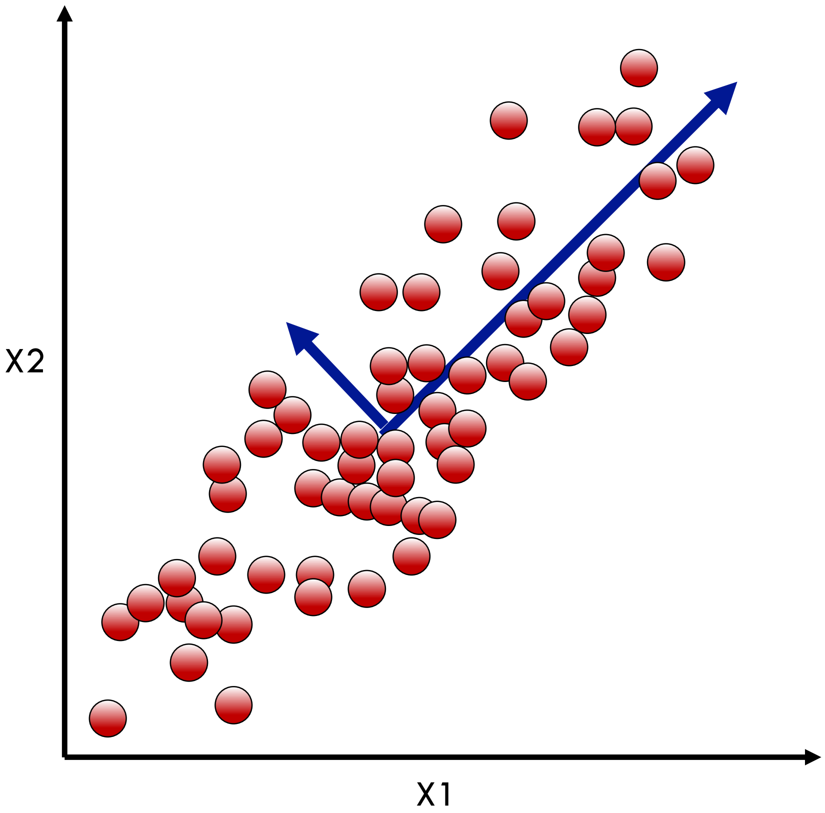
Principal Component Analysis (PCA)



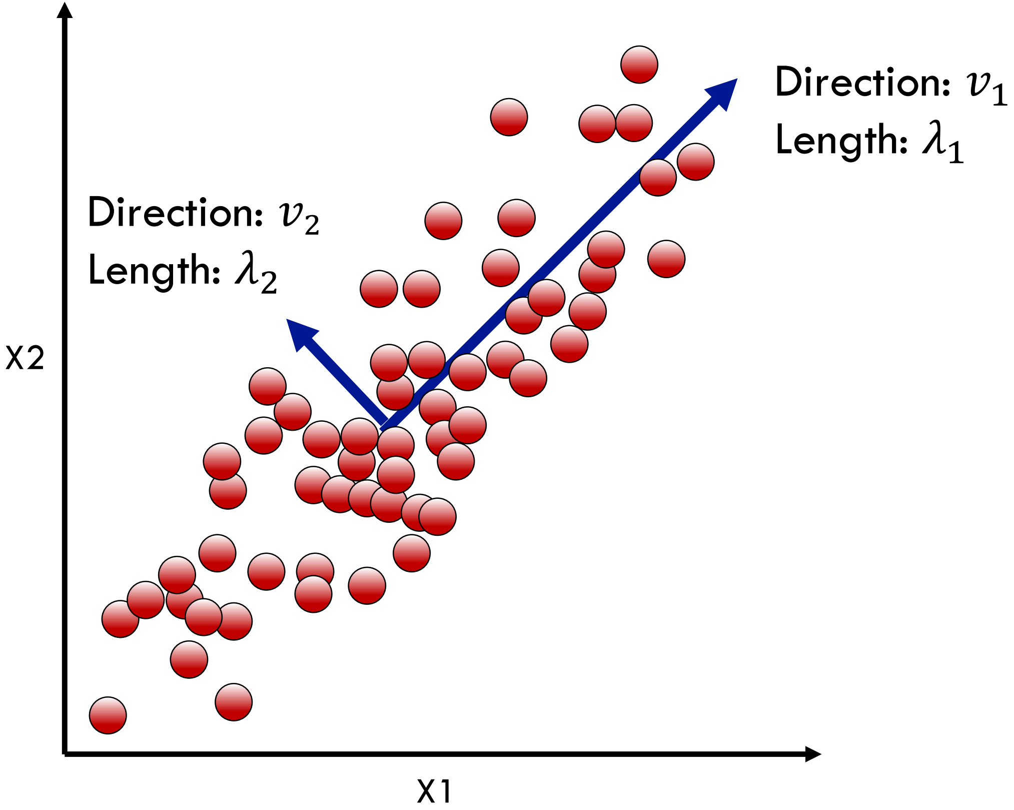
Principal Component Analysis (PCA)



Principal Component Analysis (PCA)



Principal Component Analysis (PCA)



Single Value Decomposition (SVD)

- SVD is a matrix factorization method normally used for PCA
- Does not require a square data set
- SVD is used by Scikit-learn for PCA

$$\begin{bmatrix} \star & \star & \star \\ \star & \star & \star \\ \star & \star & \star \\ \star & \star & \star \\ \star & \star & \star \end{bmatrix} = \begin{bmatrix} \star & \star & \star & \star & \star \\ \star & \star & \star & \star & \star \\ \star & \star & \star & \star & \star \\ \star & \star & \star & \star & \star \\ \star & \star & \star & \star & \star \end{bmatrix} \begin{bmatrix} \star & 0 & 0 \\ 0 & \star & 0 \\ 0 & 0 & \star \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \star & \star & \star \\ \star & \star & \star \\ \star & \star & \star \end{bmatrix}$$

$$A_{m\times n} \qquad U_{m\times m} \qquad S_{m\times n} \qquad V^T_{n\times n}$$

Truncated Single Value Decomposition

- How can SVD be used for dimensionality reduction?
- Principal components are calculated from US
- "Truncated SVD" used for dimensionality reduction ($n \rightarrow k$)

$$\underset{A_{m\times n}}{\begin{bmatrix} \star & \star & \star \\ \star & \star & \star \\ \star & \star & \star \\ \star & \star & \star \\ \star & \star & \star \end{bmatrix}} \approx \underset{U_{m\times k}}{\begin{bmatrix} \star & \star & \star & \star & \star \\ \star & \star & \star & \star & \star \\ \star & \star & \star & \star & \star \\ \star & \star & \star & \star & \star \\ \star & \star & \star & \star & \star \end{bmatrix}} \underset{S_{k\times k}}{\begin{bmatrix} \mathbf{9} & 0 & 0 \\ 0 & \mathbf{7} & 0 \\ 0 & 0 & \mathbf{1} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}} \underset{V^T_{k\times n}}{\begin{bmatrix} \star & \star & \star \\ \star & \star & \star \\ \star & \star & \star \end{bmatrix}}$$

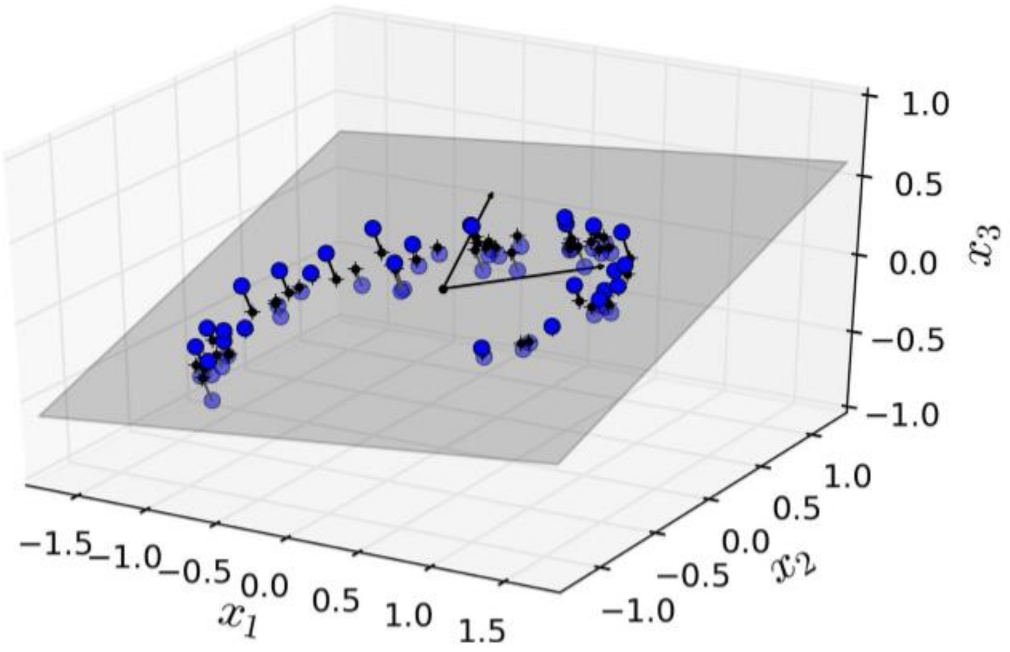


Figure 8-2. A 3D dataset lying close to a 2D subspace

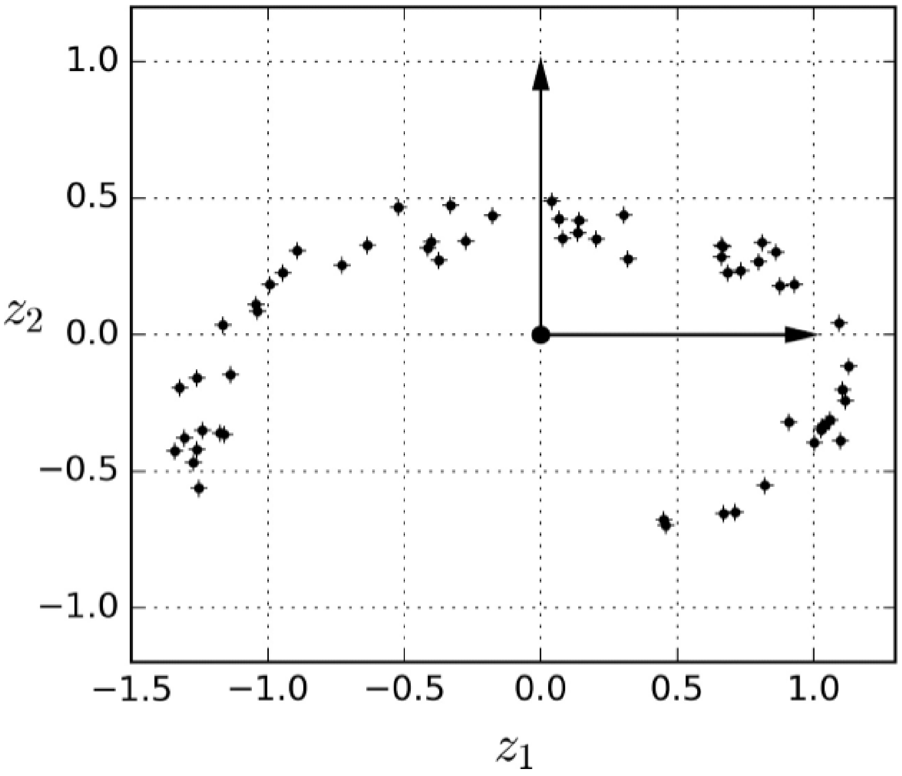
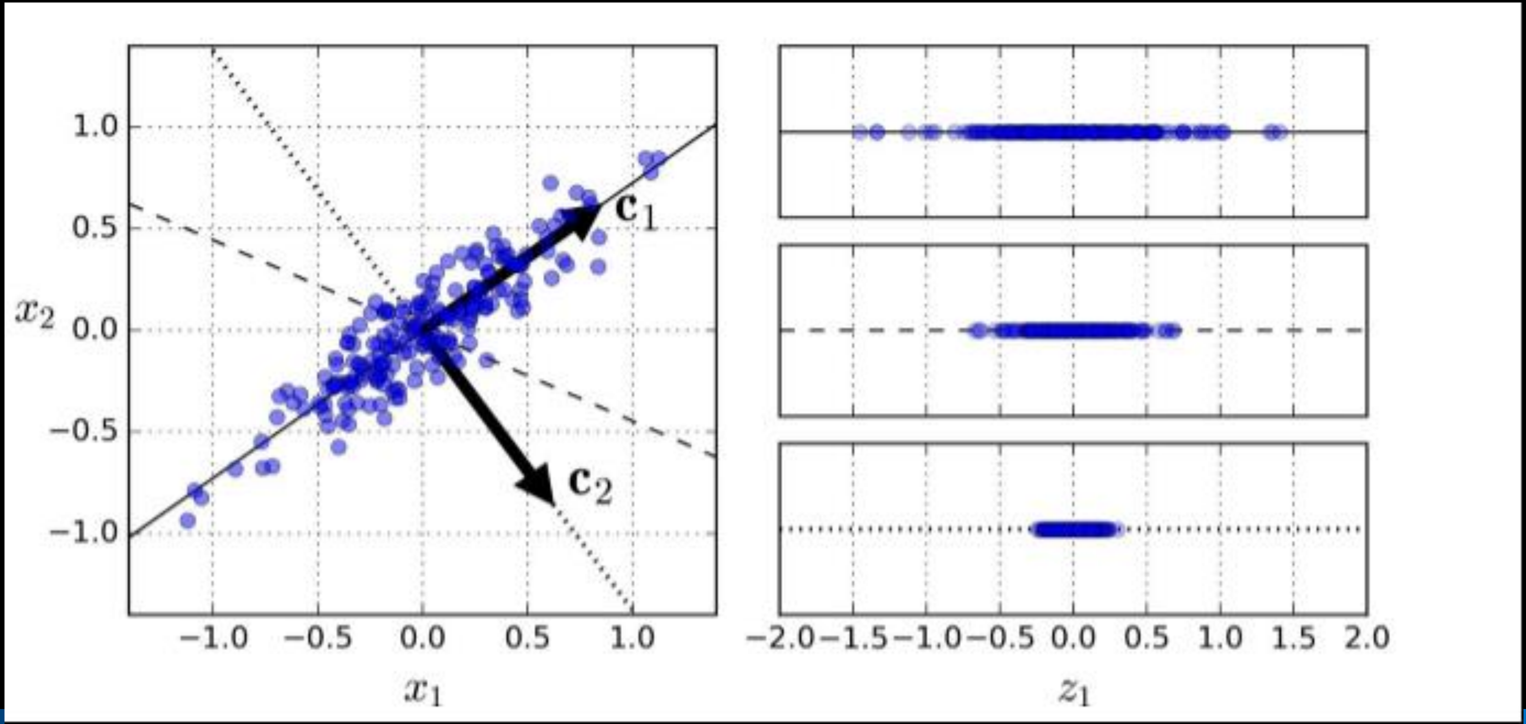


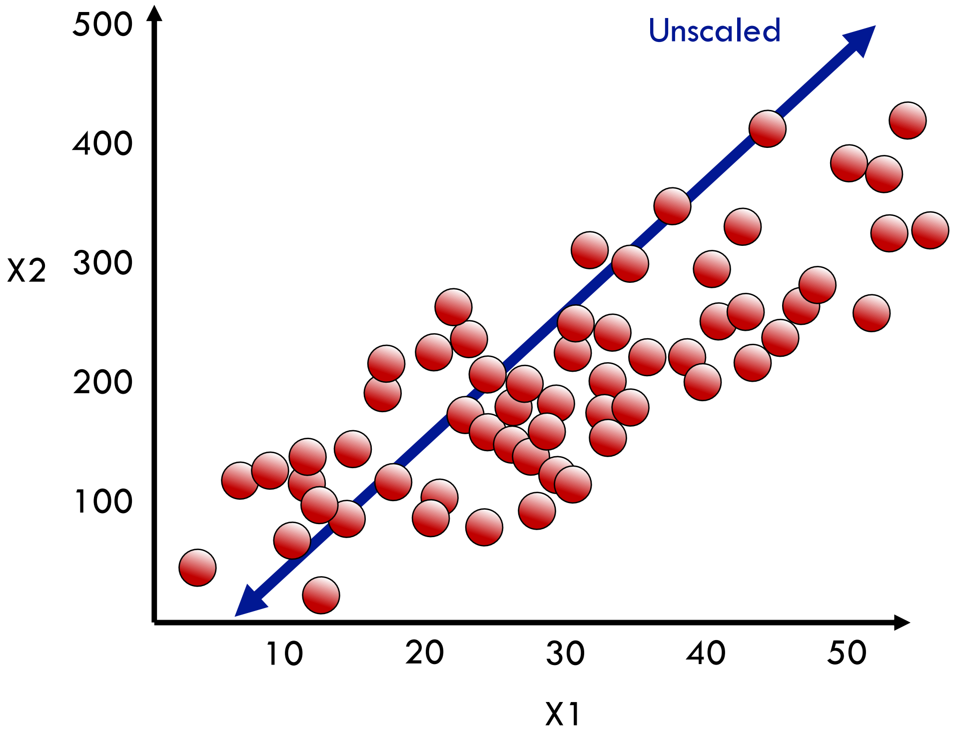
Figure 8-3. The new 2D dataset after projection

Projections along various axis



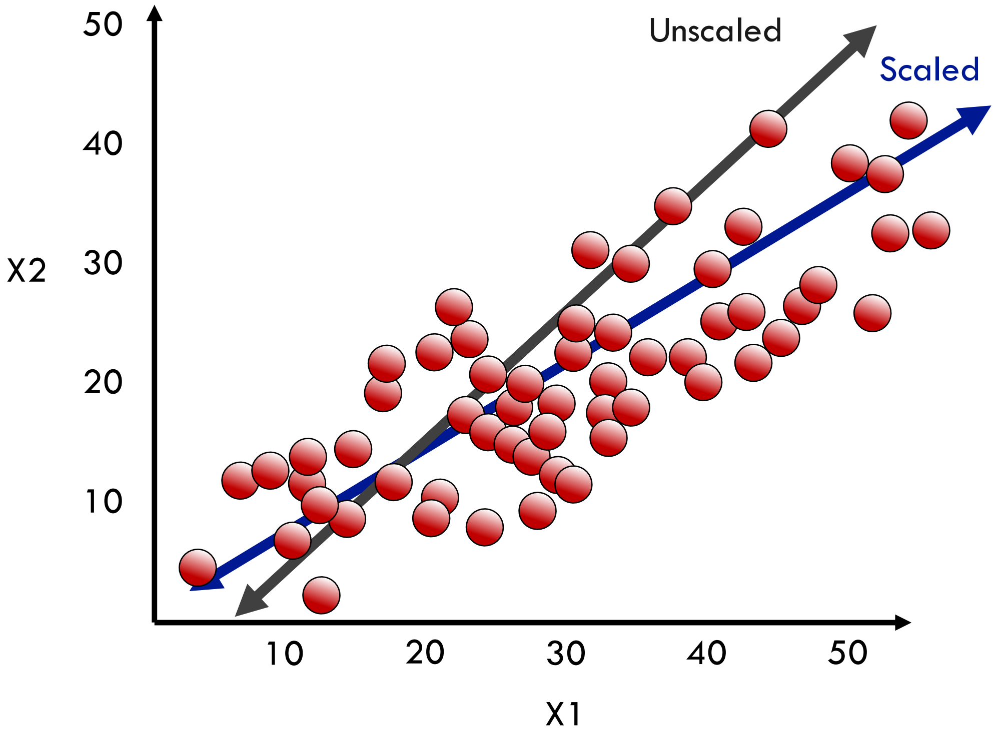
Importance of Feature Scaling

- PCA and SVD seek to find the vectors that capture the most variance
- Variance is sensitive to axis scale



Importance of Feature Scaling

- PCA and SVD seek to find the vectors that capture the most variance
- Variance is sensitive to axis scale
- Must scale data!



PCA: The Syntax

Import the class containing the dimensionality reduction method

```
from sklearn.decomposition import PCA
```

PCA: The Syntax

Import the class containing the dimensionality reduction method

```
from sklearn.decomposition import PCA
```

Create an instance of the class

```
PCAinst = PCA(n_components=3, whiten=True)
```

PCA: The Syntax

Import the class containing the dimensionality reduction method

```
from sklearn.decomposition import PCA
```

Create an instance of the class

```
PCAinst = PCA(n_components=3, whiten=True)
```

final number of dimensions

PCA: The Syntax

Import the class containing the dimensionality reduction method

```
from sklearn.decomposition import PCA
```

Create an instance of the class

```
PCAinst = PCA(n_components=3, whiten=True)
```

whiten = scale and center data

PCA: The Syntax

Import the class containing the dimensionality reduction method

```
from sklearn.decomposition import PCA
```

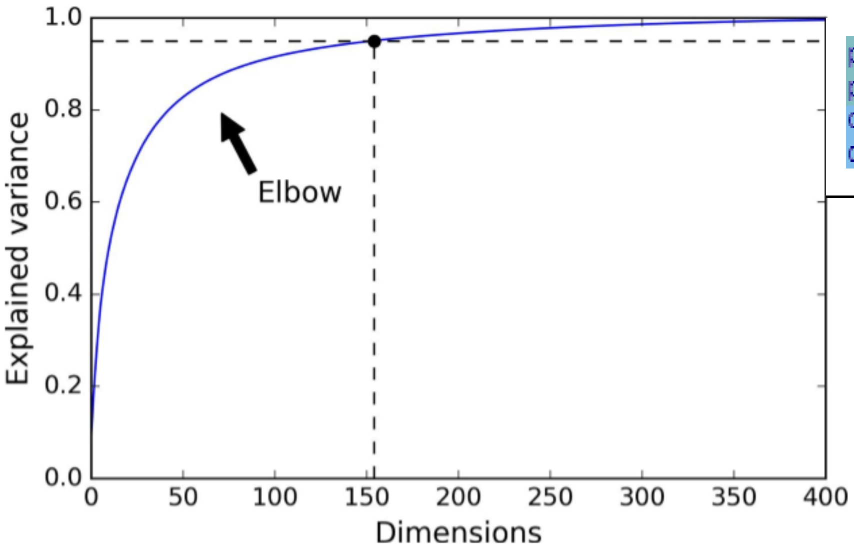
Create an instance of the class

```
PCAinst = PCA(n_components=3, whiten=True)
```

Fit the instance on the data and then transform the data

```
X_trans = PCAinst.fit_transform(X_train)
```

Explained variance




```
pca = PCA()
pca.fit(X)
cumsum = np.cumsum(pca.explained_variance_ratio_)
d = np.argmax(cumsum >= 0.95) + 1
```

PCA: The Syntax

Import the class containing the dimensionality reduction method

```
from sklearn.decomposition import PCA
```

Create an instance of the class

```
PCAinst = PCA(n_components=3, whiten=True)
```

Fit the instance on the data and then transform the data

```
X_trans = PCAinst.fit_transform(X_train)
```

Does not work with sparse matrices

Truncated SVD: The Syntax

Import the class containing the dimensionality reduction method

```
from sklearn.decomposition import TruncatedSVD
```

Create an instance of the class

```
SVD = TruncatedSVD(n_components=3)
```

Fit the instance on the data and then transform the data

```
X_trans = SVD.fit_transform(X_sparse)
```

Works with sparse matrices—used with text data for Latent Semantic Analysis (LSA)

Truncated SVD: The Syntax

Import the class containing the dimensionality reduction method

```
from sklearn.decomposition import TruncatedSVD
```

Create an instance of the class

```
SVD = TruncatedSVD(n_components=3)
```

does not center data

Fit the instance on the data and then transform the data

```
X_trans = SVD.fit_transform(X_sparse)
```

Works with sparse matrices—used with text data for Latent Semantic Analysis (LSA)

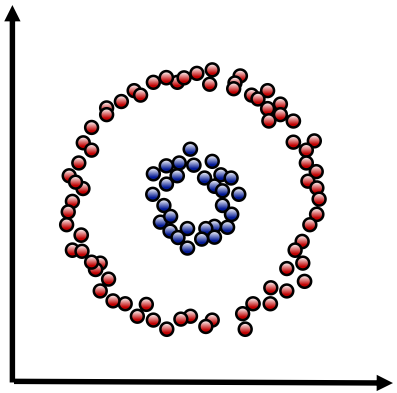
Moving Beyond Linearity

- Transformations calculated with PCA/SVD are linear

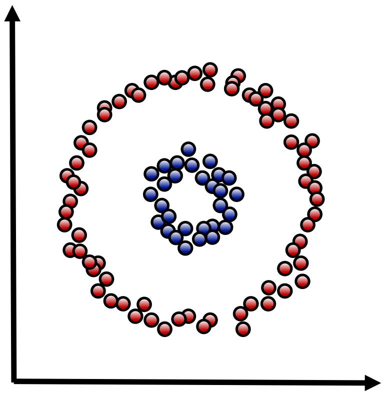
- Data can have non-linear features

- This can cause dimensionality

Original Space



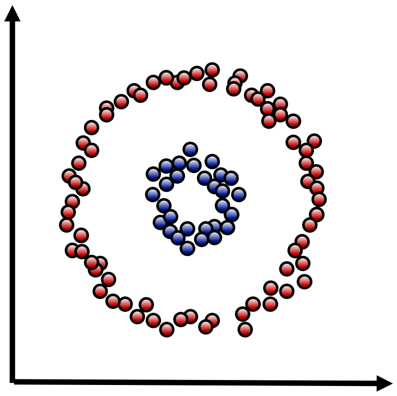
Projection by PCA



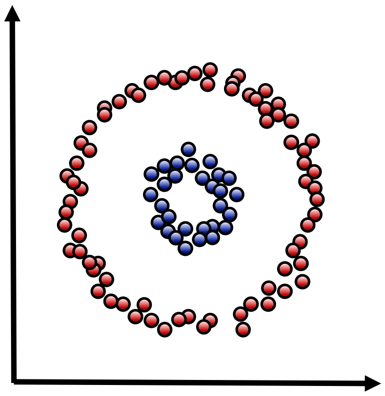
Moving Beyond Linearity

- Transformations calculated with PCA/SVD are linear
- Data can have non-linear features
- This can cause dimensionality reduction to fail

Original Space

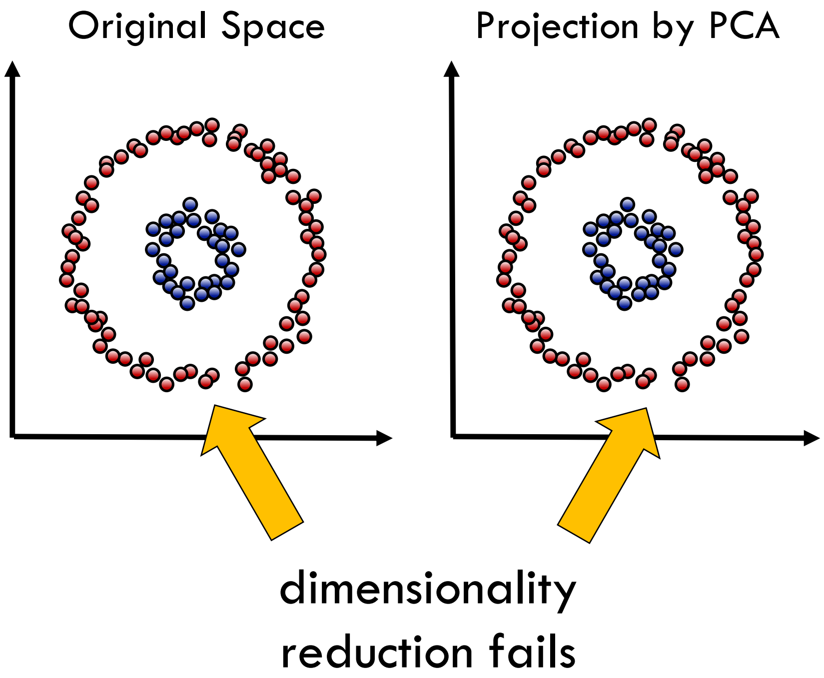


Projection by PCA



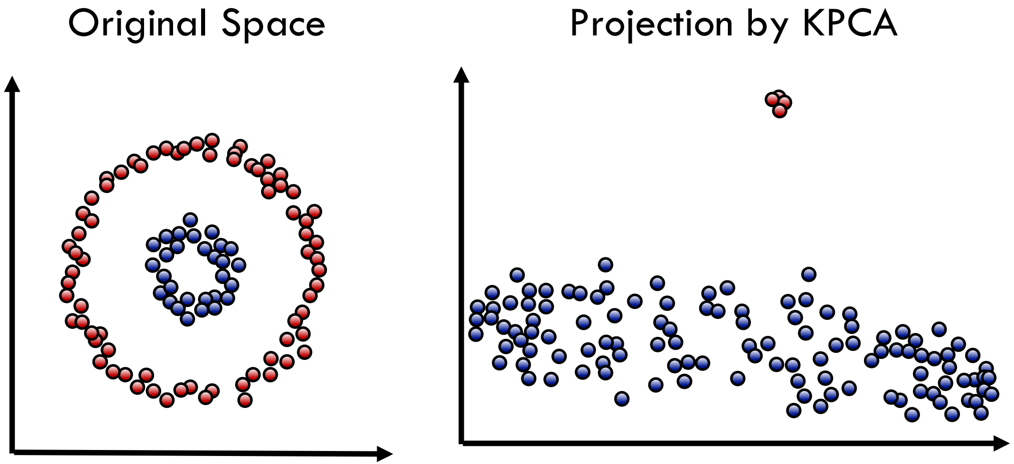
Moving Beyond Linearity

- Transformations calculated with PCA/SVD are linear
- Data can have non-linear features
- This can cause dimensionality reduction to fail



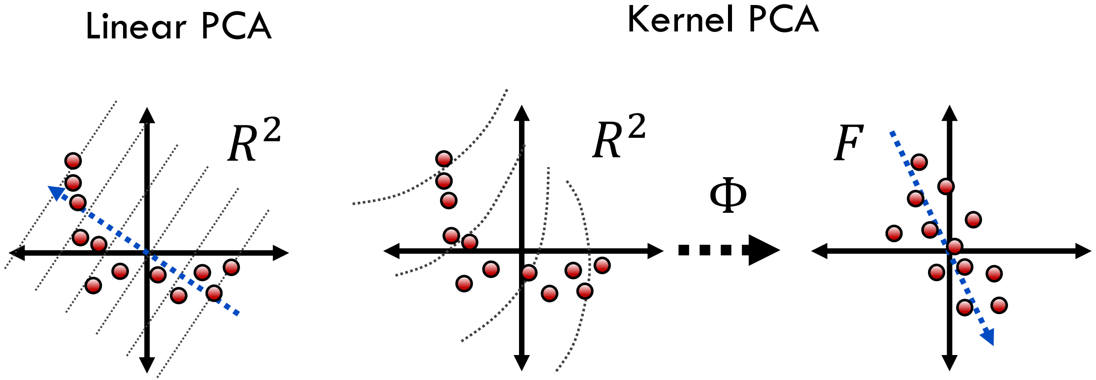
Kernel PCA

- **Solution:** kernels can be used to perform non-linear PCA

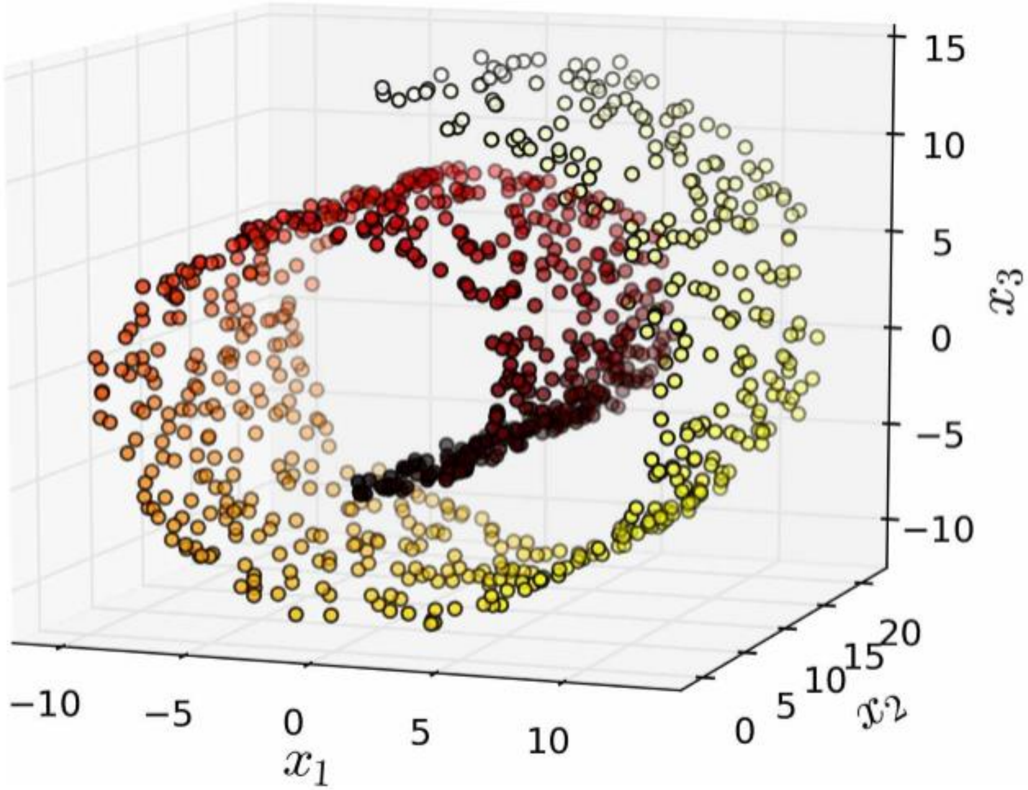


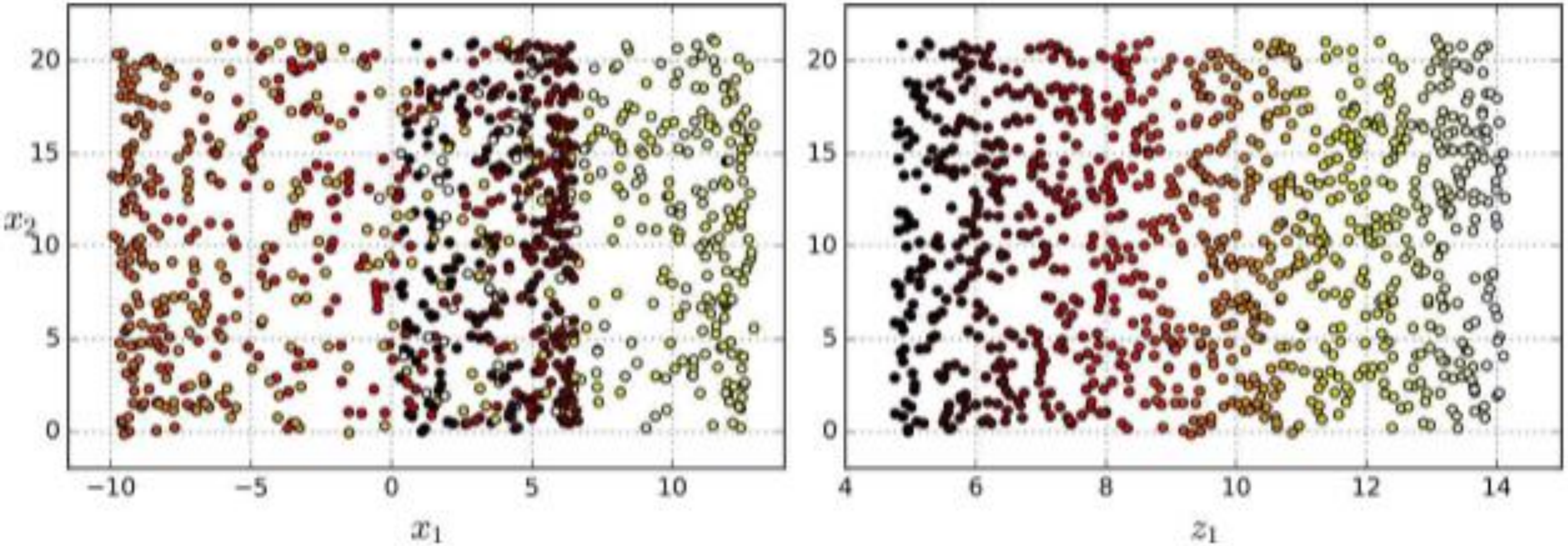
Kernel PCA

- **Solution:** kernels can be used to perform non-linear PCA
- Like the kernel trick introduced for SVMs



Swiss roll Example





Squashing by projecting onto a plane (left) versus unrolling the Swiss roll (right)

Kernel PCA: The Syntax

Import the class containing the dimensionality reduction method

```
from sklearn.decomposition import KernelPCA
```

Create an instance of the class

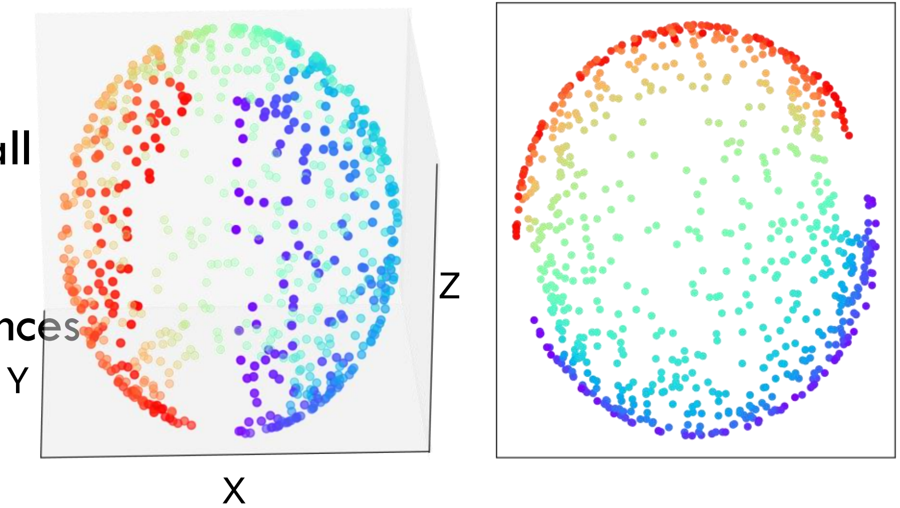
```
kPCA = KernelPCA(n_components=3, kernel='rbf', gamma=1.0)
```

Fit the instance on the data and then transform the data

```
X_trans = kPCA.fit_transform(X_train)
```

Multi-Dimensional Scaling (MDS)

- Non-linear transformation
- Doesn't focus on maintaining overall variance
- Instead, maintains geometric distances between points



MDS: The Syntax

Import the class containing the dimensionality reduction method

```
from sklearn.manifold import MDS
```

Create an instance of the class

```
mdsMod = MDS(n_components=2)
```

Fit the instance on the data and then transform the data

```
X_trans = mdsMod.fit_transform(X_sparse)
```

Many other manifold dimensionality methods exist: Isomap, TSNE.

Uses of Dimensionality Reduction

- Frequently used for high dimensionality data
- Natural language processing (NLP)—many word combinations
- Image-based data sets—pixels are features

Image Source: https://commons.wikimedia.org/wiki/File:Monarch_In_May.jpg

Uses of Dimensionality Reduction

- Divide image into 12 x 12 pixel sections

Image Source: https://commons.wikimedia.org/wiki/File:Monarch_In_May.jpg

Uses of Dimensionality Reduction

- Divide image into 12 x 12 pixel sections

- Flatten section to create row of data with 144 features

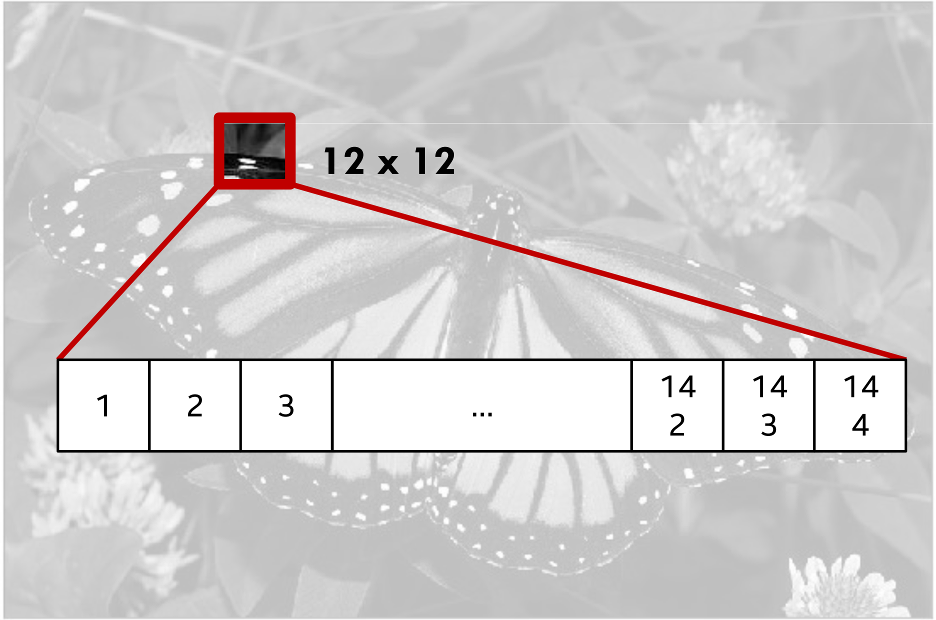


Image Source: https://commons.wikimedia.org/wiki/File:Monarch_In_May.jpg

Uses of Dimensionality Reduction

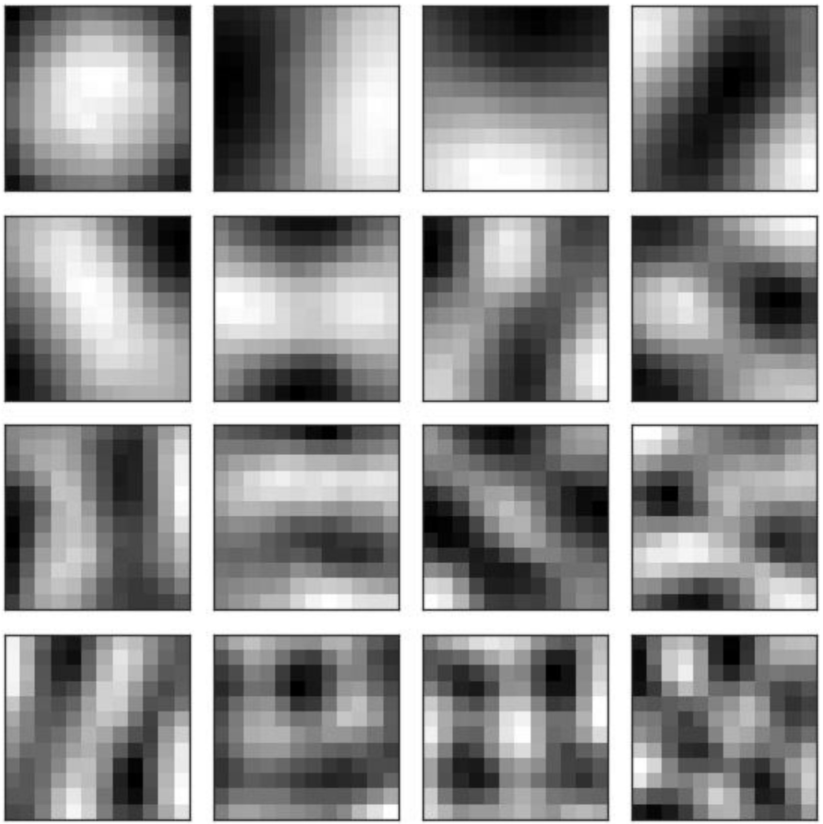
- Divide image into 12 x 12 pixel sections
- Flatten section to create row of data with 144 features
- Perform PCA on all data points

Image Source: https://commons.wikimedia.org/wiki/File:Monarch_In_May.jpg

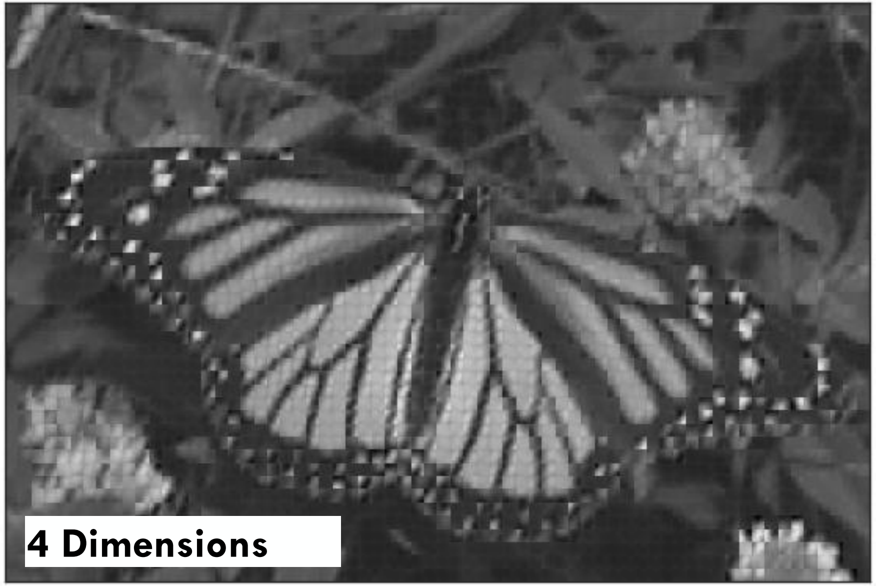
PCA Compression: 144 → 60 Dimensions

PCA Compression: 144 → 16 Dimensions

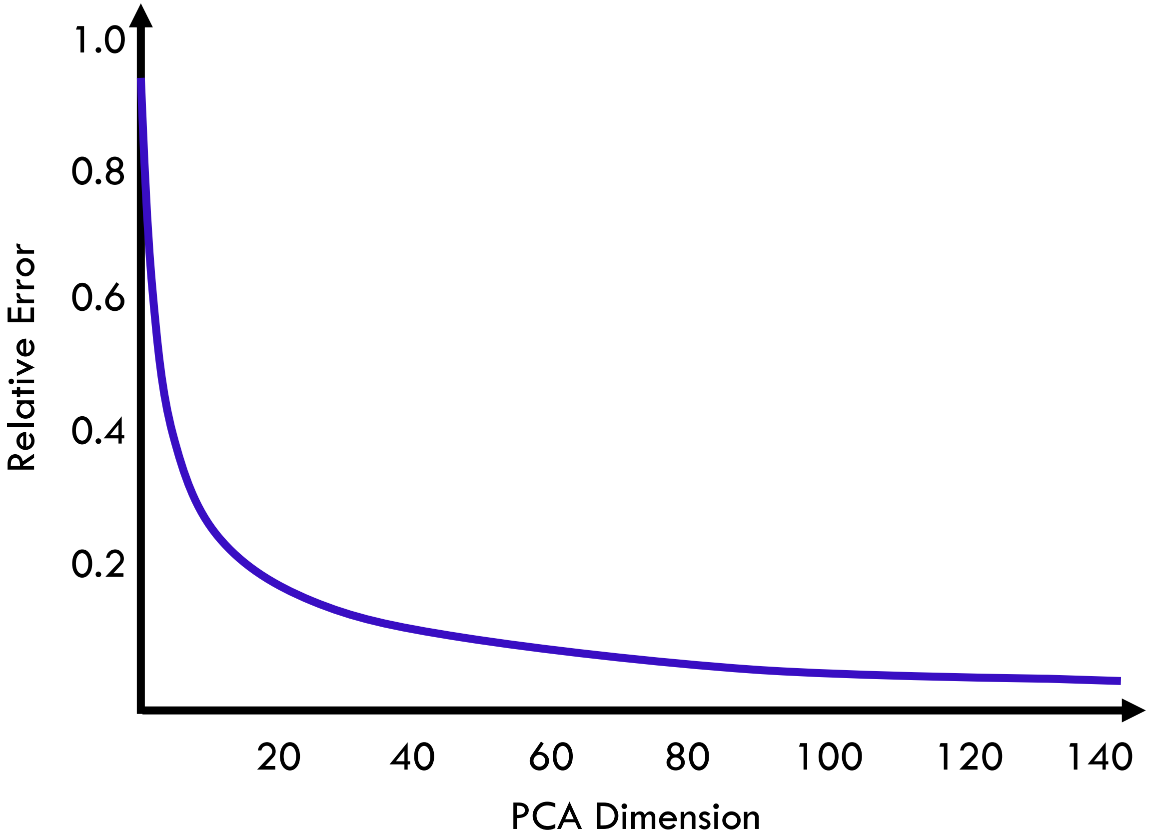
Sixteen Most Important Eigenvectors



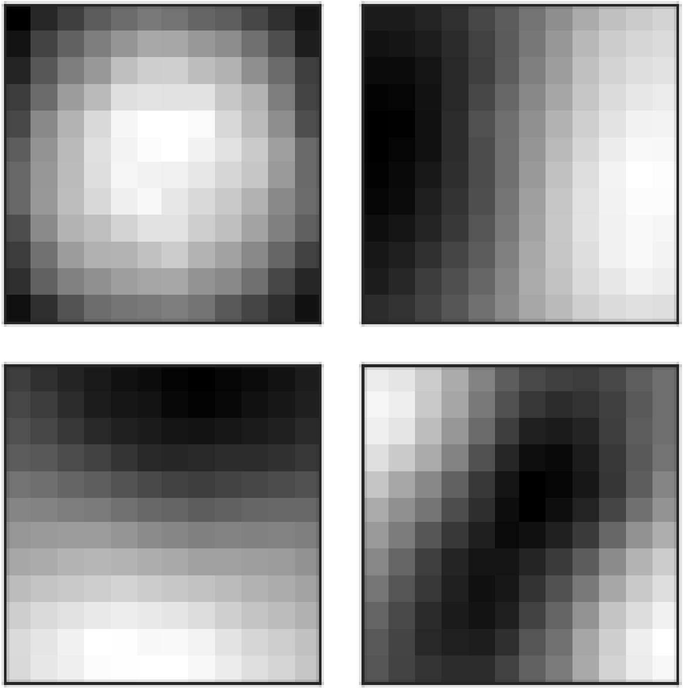
PCA Compression: 144 → 4 Dimensions



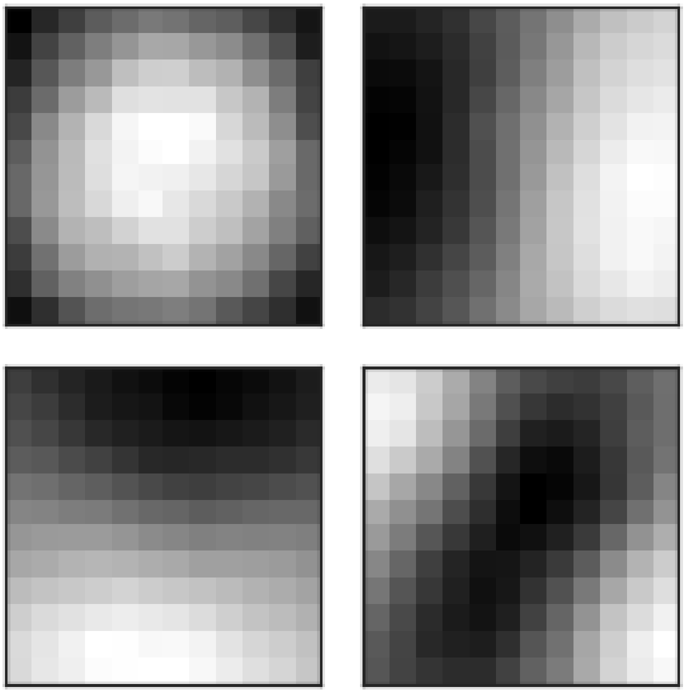
L2 Error and PCA Dimension

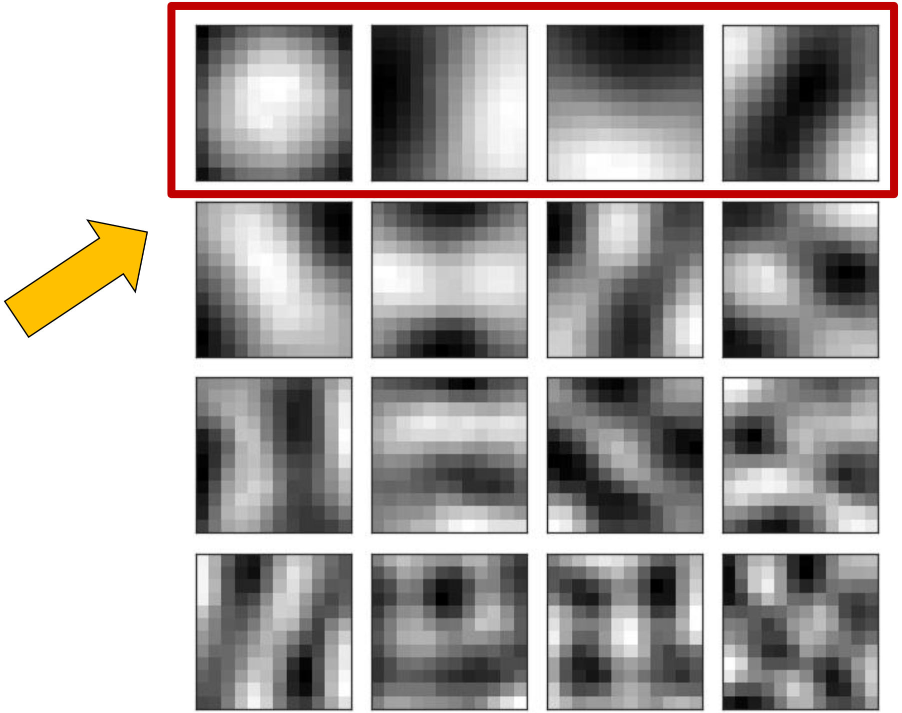


Four Most Important Eigenvectors



Four Most Important Eigenvectors





PCA Compression: 144 → 1 Dimension