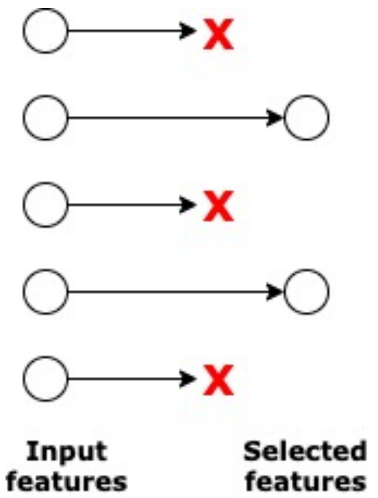
Dimensionality Reduction: Feature Extraction

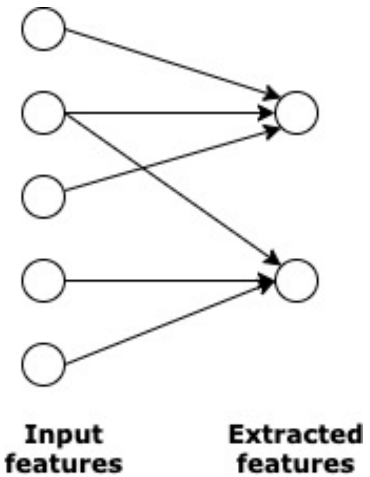
Tozammel Hossain

Data Science & Analytics
University of Missouri

Dimensionality Reduction

- **Removes irrelevant, redundant, and noisy features**
  - Could be used with supervised and unsupervised settings
  - Learns character/structure of data
  - Used as a preprocessing step
- **Dimensionality Reduction**
  - Feature Selection
    - aka subset selection
    - finding a subset of features that give most information
  - Feature Extraction
    - finding a new subset of features that are combinations of original dimensions
    - A type of feature engineering





Feature Extraction

- **Principal Component Analysis**
- **Factor Analysis**
- **Goal**
  - Derive a new set of features that captures the character of data
  - New features might give better downstream analysis
  - Unsupervised Methods
    - Compare with other feature selection techniques
      - ch2, mutual information, wrapper methods

Why study PCA and FA?

- **Remember the discussion from last class**
  - Reduce dimension for simpler model
  - More efficient use of resources
    - e.g., time, memory, communication
  - Visualization
  - Statistical: fewer dimensions => better generalization
  - Noise removal (improving data quality)
- **Study phenomena that can not be directly observed**
  - ego, personality, intelligence in psychology
  - disease state given vitals and lab reports

Why study PCA and FA?

- **Operate with underlying latent factors rather than the observed data**
  - Topics in news articles
  - Interpretable
- **Better representation of data without losing much information**
  - Compression
- **Combinations of observed variables may be more effective bases for insights, even if physical meaning is obscure**
  - We want to discover and exploit hidden relationships
  - E.g., Weight and height have relationship.

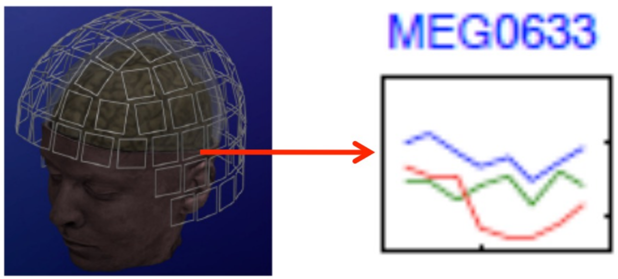
Big & High-Dimensional Data

- **Text Analysis**
  - Features per document = thousands of words/unigrams millions of bigrams, contextual information
- **Review data**
  - Lots of users and reviews over items
  - Netflix data (480189 users x 17770 movies)

	movie 1	movie 2	movie 3	movie 4	movie 5	movie 6
Tom	5	?	?	1	3	?
George	?	?	3	1	2	5
Susan	4	3	1	?	5	1
Beth	4	3	?	2	4	2

Big & High-Dimensional Data

- **MEG Brain Imaging**
  - Features per document = thousands of words/unigrams millions of bigrams, contextual information
  - 120 locations x 500 time points x 20 objects



- **Gene expression data**
  - Each cell, measure gene expression, at various time point

Representation Learning

- **Unsupervised learning techniques for extracting hidden (potentially lower dimensional) structure from high dimensional datasets**
- **Remember this term along with SL, USL, RL**

Basic Concept of PCA and FA

- **Areas of variance in data are where items can be best discriminated and key underlying phenomena observed**
  - Areas of greatest "signal" in the data
- **If two features or dimensions are highly correlated or dependent**
  - They are likely to represent highly related phenomena
  - If they tell us about the same underlying variance in the data, combining them to form a single measure is reasonable

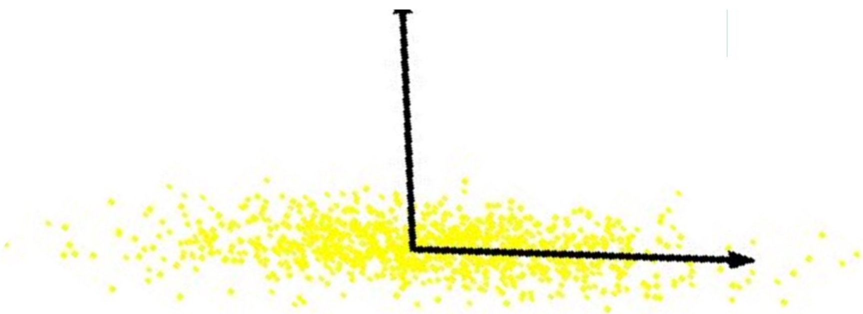
Basic Concept PCA and FA

- **We want to combine related variables, and focus on uncorrelated or independent ones, especially those along which the observations have high variance**
  - Remember multicollinearity from M2
- **We want a smaller set of variables that explain most of the variance in the original data, in more compact and insightful form**

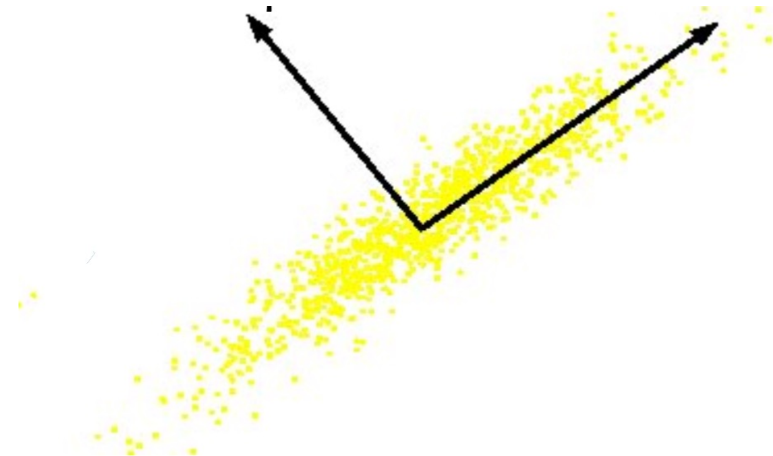
Principal Component Analysis

- **Most common form of feature extraction**
- **Unsupervised technique for extracting variance structure from high dimensional datasets**

Principal Component Analysis



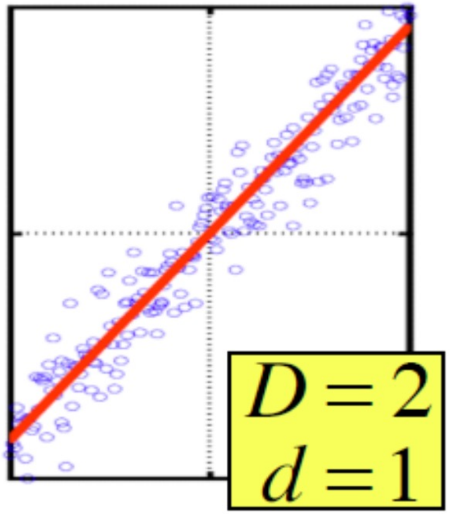
one relevant feature

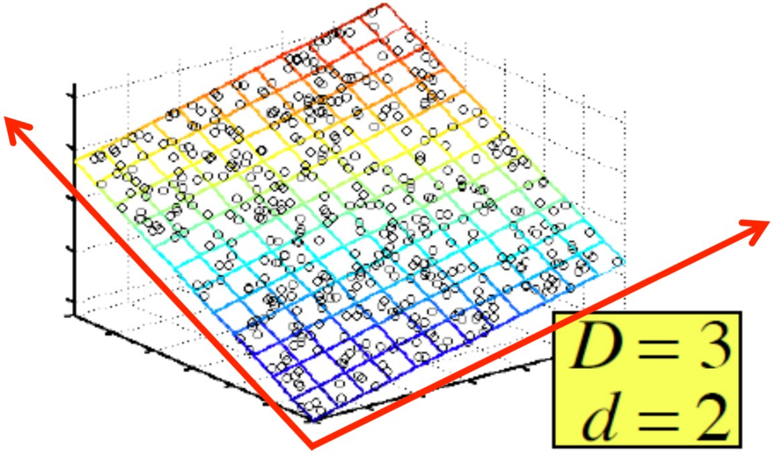


two relevant features

Q. Can we transform the features so that we only need to preserve one latent feature?

Principal Component Analysis





- data lies on or near a low d-dimensional linear subspace
  - axes of this subspace are an effective representation of the data
  - Identifying the axes is known as Principal Components Analysis
    - Done via matrix computation tool (SVD)

Principal Component Analysis

- **The new variables/dimensions**
  - Are linear combinations of the original ones
  - Are uncorrelated with one another
    - Orthogonal in original dimension space
  - Capture as much of the original variance in the data as possible
  - Are called Principal Components

What are the new axes?

- **Orthogonal directions of greatest variance in data**
  - Ranked in terms of variance
- **Projections along PC1 discriminate the data most along any axis**
  - the direction of greatest variability (covariance) in the data
  - PC2 is the next orthogonal (uncorrelated) direction of greatest variability

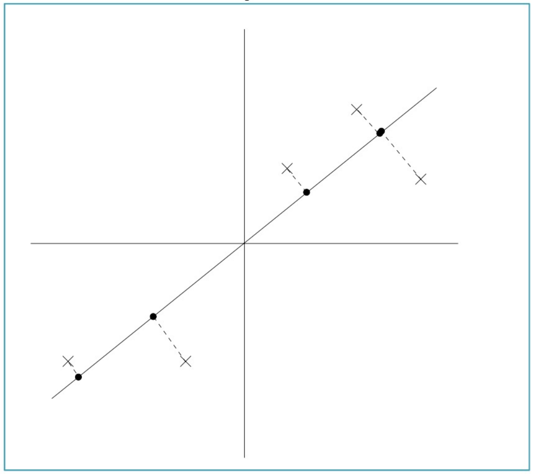
Algorithm

- **Given the centered data {x1, …, x*m*}, compute the principal vectors**

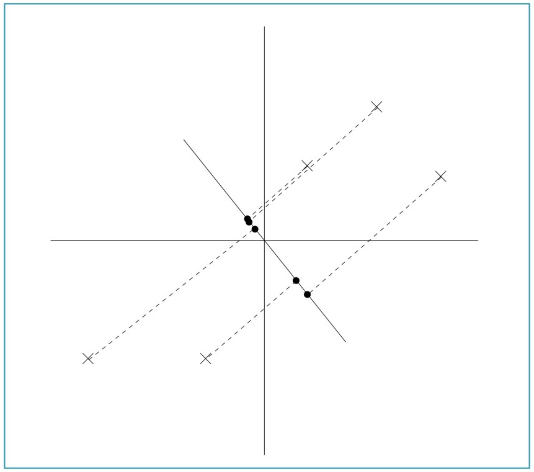
$$\mathbf{w}_1 = \arg\max_{\|\mathbf{w}\|=1} \frac{1}{m}\sum_{i=1}^{m}\{(\mathbf{w}^T\mathbf{x}_i)^2\}$$

1st PCA vector

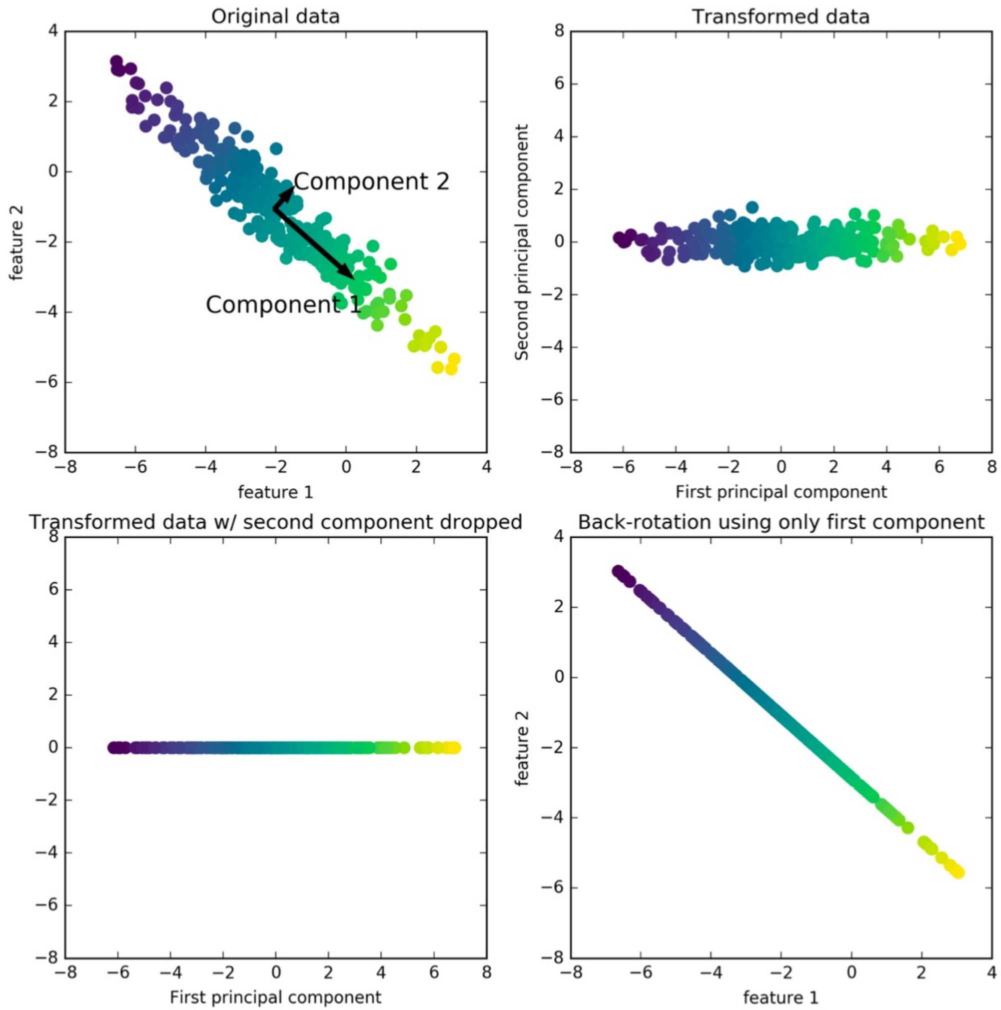
Option A



Option B



Data Transformation & Reconstruction



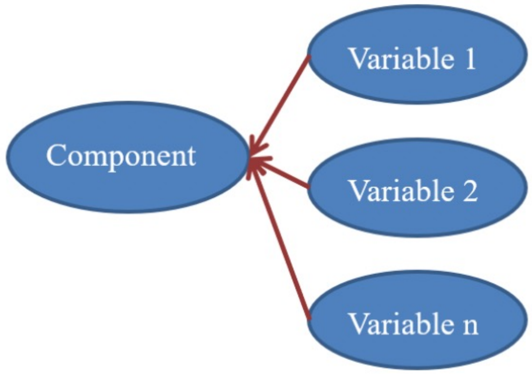
source: Intro to ML with Python by Muller and Guido

PCA: Two Interpretations

- **Maximum Variance Direction:**
  - 1st PC is a vector v such that projection on to this vector capture maximum variance in the data (out of all possible one dimensional projections)
- **Minimum Reconstruction Error:**
  - 1st PC is a vector v such that projection on to this vector yields minimum MSE reconstruction

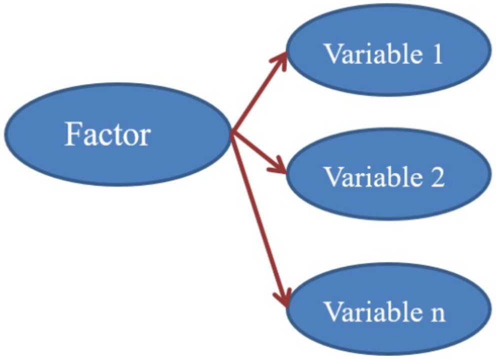
Feature selection with PCA

- **How many components?**
  - n features = n components
- **Feature selection with PCA**
  - After rotation, select components (i.e, new features) that are important in explaining the data



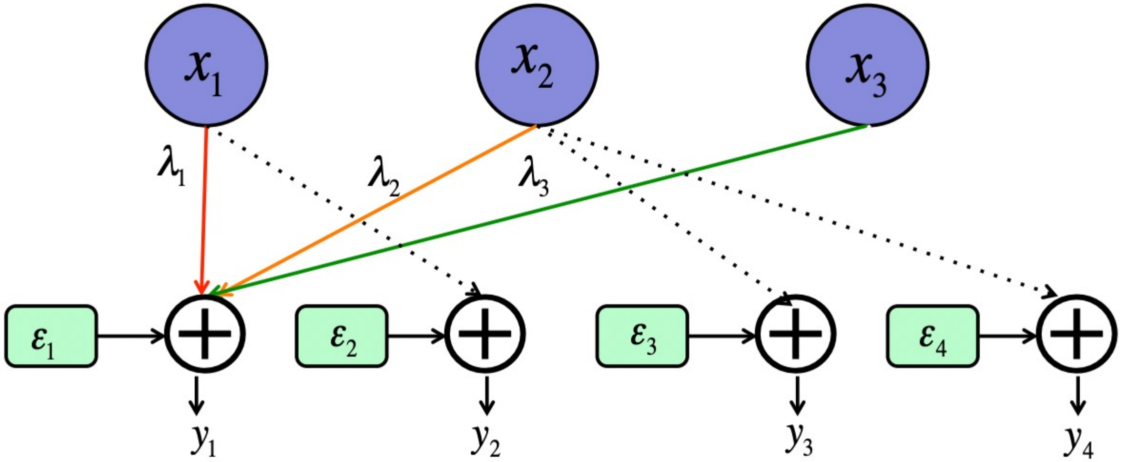
Factor Analysis

- **Hidden/Latent Variable Model**



- **Assumption**
  - Factors generate the features
    - A generative model
      - Can you remember a generative model from M1/M2?
  - A Feature is a linear combination of factors

Factor Analysis



- **Analysis of the covariance in observed variables**
  - In terms of few (latent) common factors + a specific error
- **How to learn such method?**
  - EM algorithm (Expectation Maximization)

FA terminologies

$$\mathbf{X} = W\mathbf{H} + \mathbf{M} + \mathbf{E}$$

- X: features
- W: factor loading
- H: factor scores
- E: error terms
- M: mean of the features

Other Feature Extraction Methods

- **ICA**
  - Independent Component Analysis
- **NMF**
  - Non-negative Matrix Factorization
- **LDA**
  - Latent Dirichlet Distribution
- **Auto-encoder**

Search representation learning

sklearn Decomposition

sklearn.decomposition: Matrix Decomposition

The sklearn.decomposition module includes matrix decomposition algorithms, including among others PCA, NMF or ICA. Most of the algorithms of this module can be regarded as dimensionality reduction techniques.

User guide: See the Decomposing signals in components (matrix factorization problems) section for further details.

decomposition.DictionaryLearning([...])	Dictionary learning
decomposition.FactorAnalysis([n_components, ...])	Factor Analysis (FA).
decomposition.FastICA([n_components, ...])	FastICA: a fast algorithm for Independent Component Analysis.
decomposition.IncrementalPCA([n_components, ...])	Incremental principal components analysis (IPCA).
decomposition.KernelPCA([n_components, ...])	Kernel Principal component analysis (KPCA).
decomposition.LatentDirichletAllocation([...])	Latent Dirichlet Allocation with online variational Bayes algorithm
decomposition.MiniBatchDictionaryLearning([...])	Mini-batch dictionary learning
decomposition.MiniBatchSparsePCA([...])	Mini-batch Sparse Principal Components Analysis
decomposition.NMF([n_components, init, ...])	Non-Negative Matrix Factorization (NMF).
decomposition.PCA([n_components, copy, ...])	Principal component analysis (PCA).
decomposition.SparsePCA([n_components, ...])	Sparse Principal Components Analysis (SparsePCA).
decomposition.SparseCoder(dictionary, *[, ...])	Sparse coding
decomposition.TruncatedSVD([n_components, ...])	Dimensionality reduction using truncated SVD (aka LSA).

decomposition.dict_learning(X, n_components, ...)	Solves a dictionary learning matrix factorization problem.
decomposition.dict_learning_online(X[, ...])	Solves a dictionary learning matrix factorization problem online.
decomposition.fastica(X[, n_components, ...])	Perform Fast Independent Component Analysis.
decomposition.non_negative_factorization(X)	Compute Non-negative Matrix Factorization (NMF).
decomposition.sparse_encode(X, dictionary, *)	Sparse coding

Some points to remember

- **May not give better interpretability**
  - In some case it does
  - Explain in terms of new variables
    - Original features are kind of lost
- **Check the lab “when PCA attacks”**
  - Works independent of the learning method
  - May increase overlap between classes