Causal Inference (6.S059/15.C08/17.C08)

Recitation, Week 8.

Topic: Unsupervised Learning

Benjamín Muñoz

April 12, 2024

MIT

Massachusetts Institute of Technology

Table of contents

1/ Unsupervised Learning

Machine Learning Language

1. **Features** = Inputs, Regressors, Covariates, Independent Variables (**X**).
2. **Targets** = Outputs, Responses, Outcomes, Dependent Variables (Y).
3. **Labeled Set** of input-output pairs ($D = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$).
   - D is called the training set, N is the number of training examples, and p the number of features (variables).
4. **Function** (f): transforming the features or relating the features to the target.

Unsupervised Learning Algorithms

- Algorithms that experience only “features” but not a supervision signal. We are only given inputs, $D = \{x_i\}_{i=1}^{N}$ (**Unlabeled Data**). Nevertheless, we can learn relationships and structure from such data.
- The goal is to find **“interesting patterns”** in the data (not predicting the target).
    - To explore, learn, and summarize large amounts of data in an efficient way.
- **Knowledge Discovery**: much less well-defined problem and more subjective (not told what kinds of patterns to look for, and there is no obvious error metric to use).

Unsupervised Learning Algorithms

- **Particularities:**
  1. Unconditional density estimation $p(\boldsymbol{x}|\theta)$.
  2. Multivariate Probability Models: High dimensionality of $\boldsymbol{X}$.
  3. Hard to assess the results (no universally accepted performance measure). No measure of "success".

- "Best" representation of the data:
  1. Clustering.
  2. Dimension Reduction.
  3. Other: Association Rules, Self-Organizing Maps, etc.

Clustering and Dimension Reduction

Dimension Reduction

Clustering

Key Variable	Variable A	Variable B	Variable C	Variable D	Variable E	Variable F	Variable G	Variable H
1	3.1	7.3	1	23	86	Red	4.9	19
2	5.0	8.5	0	44	95	Green	5.0	20
3	5.0	8.5	0	44	78	Red	5.0	14
4	1.0	8.4	1	50	91	Blue	4.1	13

- Traditional Distinction:
  1. Dimension Reduction is related to the number of features (p).
  2. Clustering is related to partitioning the data points (n) into groups (similarity of subpopulations, within-group homogeneity).
- For a statistical connection see Ding and He (2004).
- Dimension Reduction implies a hidden **continuous** factor and Clustering a latent variable with K **finite** (categorical) states.

Dimension Reduction

- Dimension reduction comes in many flavors (see Waggoner (2021)):
    1. Principal Components Analysis (PCA).
    2. Factor Analysis.
    3. Modern Versions: Probabilistic PCA, Sparse PCA, Independent PCA, Bayesian versions.
    4. Manifolds: Locally Linear Embedding, Nonlinear (t-SNE).
    5. Measurement techniques: see Armstrong et al. (2020).

OLS vs PCA

$$Y = X\beta + \epsilon$$

where vector $Y_{n\times 1}$ and matrix $X_{n\times p}$ are observed data, and $\beta_{p\times 1}$ and $\epsilon_{n\times 1}$ are unobserved.
We estimate β by minimizing the sum of squared residuals:

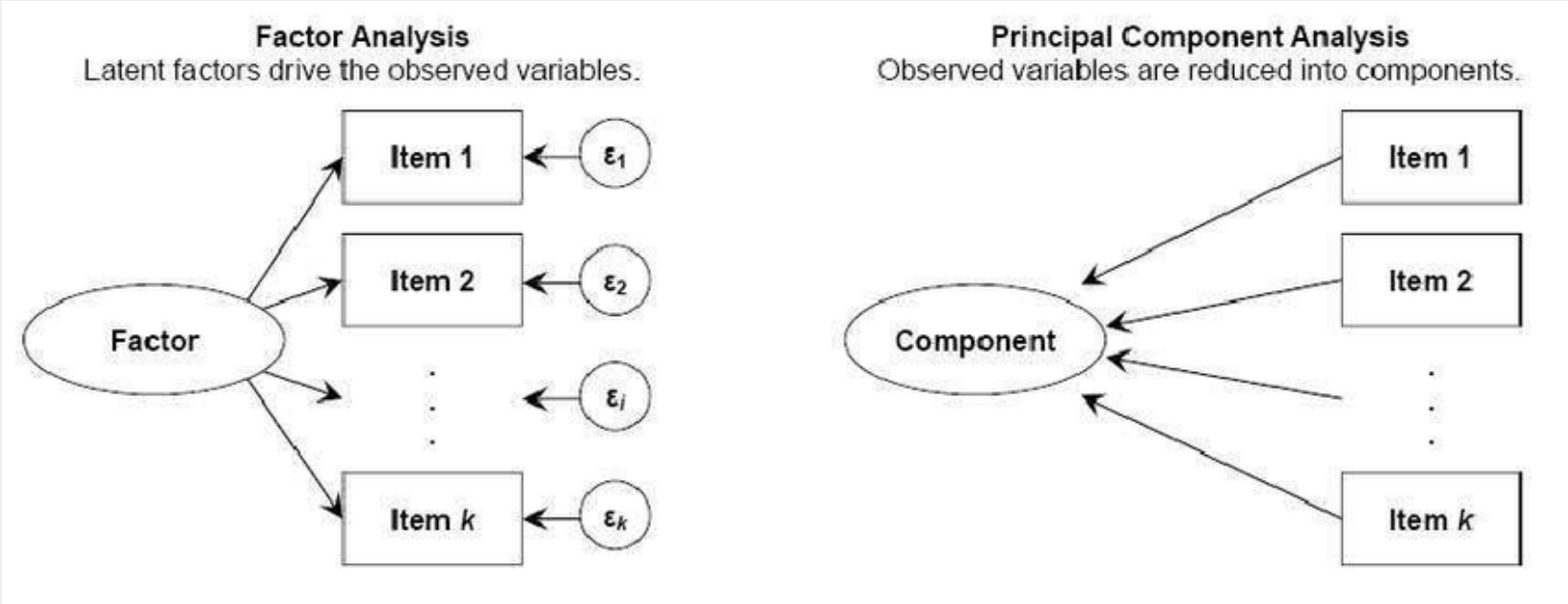
$$\sum_{i=1}^{N}(Y - X\hat{\beta})^2$$

$$Y = X\beta^T + \epsilon$$

where we observe matrix Y of size $n \times d$, but observe no covariates. We estimate a matrix $\hat{X}$ that gives k orthogonal "predictors" and $\hat{\beta}$ that gives $k \times d$ "coefficients" of the k predictors. ϵ is an $n \times d$ matrix. k is the target dimensionality of the data. $\hat{X}$ and $\hat{\beta}$ are chosen to minimize the mean squared error:

$$\sum_{i=1}^{N}(Y - \hat{X}\hat{\beta})^2$$

PCA and Factor Analysis



- Traditional Distinction: PCA is a discriminative model, and FA is a representational/generative model.
- Machine Learning does not impose these restrictions on latent variables. FA= latent factors are Gaussian. See Bishop (2006), section 12.2.

Principal Component Analysis

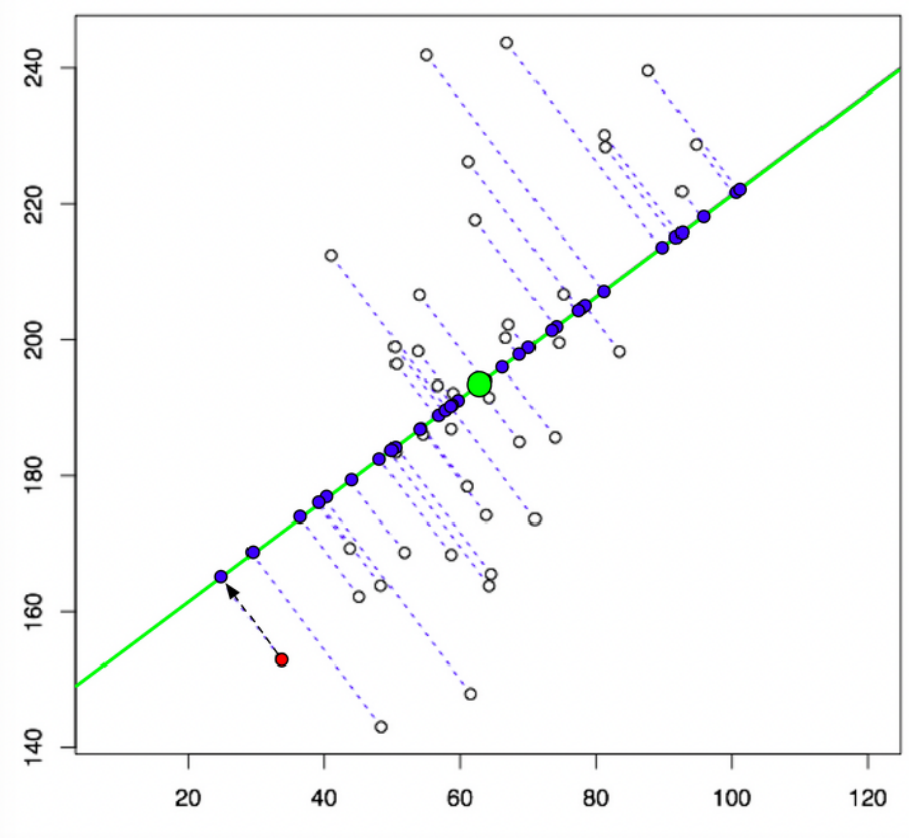
- Starting Point: high dimensionality of $\boldsymbol{X}$.
  - Difficult to "see" and interpret.
  - Loss of degrees of freedom in modeling.
  - **Dimensionality Curse:** as the dimensionality of the space increases, true similarities and differences across the features become less clear.

- **Goal:** reduce the dimensionality by projecting it to a lower dimensional subspace which captures the "essence" of the data (structure of the original data in a clearer, more digestible and generally simpler way).

  - Variance is the best way to conceptualize structure in a data space.
  - Assume that each observed high-dimensional input $\boldsymbol{x}_n \in \mathbb{R}^D$ was generated by a set of hidden or unobserved low-dimensional $\boldsymbol{z}_n \in \mathbb{R}^K$ ($K < D$).
  - Lower-dimensional version of the data space that maximizes the total variance across the full set of inputs, $\boldsymbol{X}$.

Principal Component Analysis

$$\underbrace{\mathbf{X}}_{N\times 2} = \begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ \vdots & \vdots \\ x_{N1} & x_{N2} \end{pmatrix} \qquad \underbrace{\mathbf{Z}}_{N\times 1} = \begin{pmatrix} z_{11} \\ z_{21} \\ \vdots \\ z_{N1} \end{pmatrix}$$



Principal Component Analysis

- PCA computes a new set of components based on a **linear** combination of weighted features values, which are often considered as "prototypes" that capture unique variance in higher dimensions (orthogonal to all other components).

    1. PCA is initialized to look for the direction in the data along which the full data vary most.
    2. Once that direction is found, PCA computes and places a summary line, called a principal component (C_1), that summarizes the direction of maximal variance.
    3. Thus, once C_1 is found, PCA proceeds to search for the next direction in the data along which the second-largest amount of variation exists...

- Sequence of projections of the data, mutually uncorrelated and ordered in variance.

Principal Component Analysis

- Some Tips:
  - Not full statistical independence (linearity).
  - Standardization of inputs (mean zero and variance one).
    1. Mean-transformation: change of location (linearity) $Z = X - \mathbb{E}[X], \mathbb{E}[Z] = 0$.
    2. Variance transformation: change of spread $\mathbb{V}[aX] = a^2\mathbb{V}[X]$. If $a = \frac{1}{\sigma}$ (inverse of SD), $\mathbb{V}[aX] = \frac{1}{\sigma^2}\mathbb{V}[X] = \frac{\mathbb{V}[X]}{\mathbb{V}[X]} = 1$.
  - Different algorithms:
    1. Eigen/Spectral Decomposition (function `princomp`).
    2. Singular Value Decomposition (function `prcomp`).
  - Whether these patterns accurately reflect "real life" or preconceptions of substantive phenomena is often left to the researcher to decide.
  - The researcher does not define the number of components calculated by the PCA algorithm. She only decides how to interpret them and whether she will take one (or several) components in subsequent statistical tests.

2/ Practical Exercise

China's Ideological Spectrum (Pan and Xu, 2018).

- Ideological structure in authoritarian contexts.
- Survey: 10,000 respondents and 50 ideological questions.
- In each question, a respondent is asked to what extent she agrees with the statement shown to her on a likert scale from 1 to 4, where 1 indicates "Strongly Disagree" and 4 indicates "Strongly Agree."
- See the Online Appendix for more details.

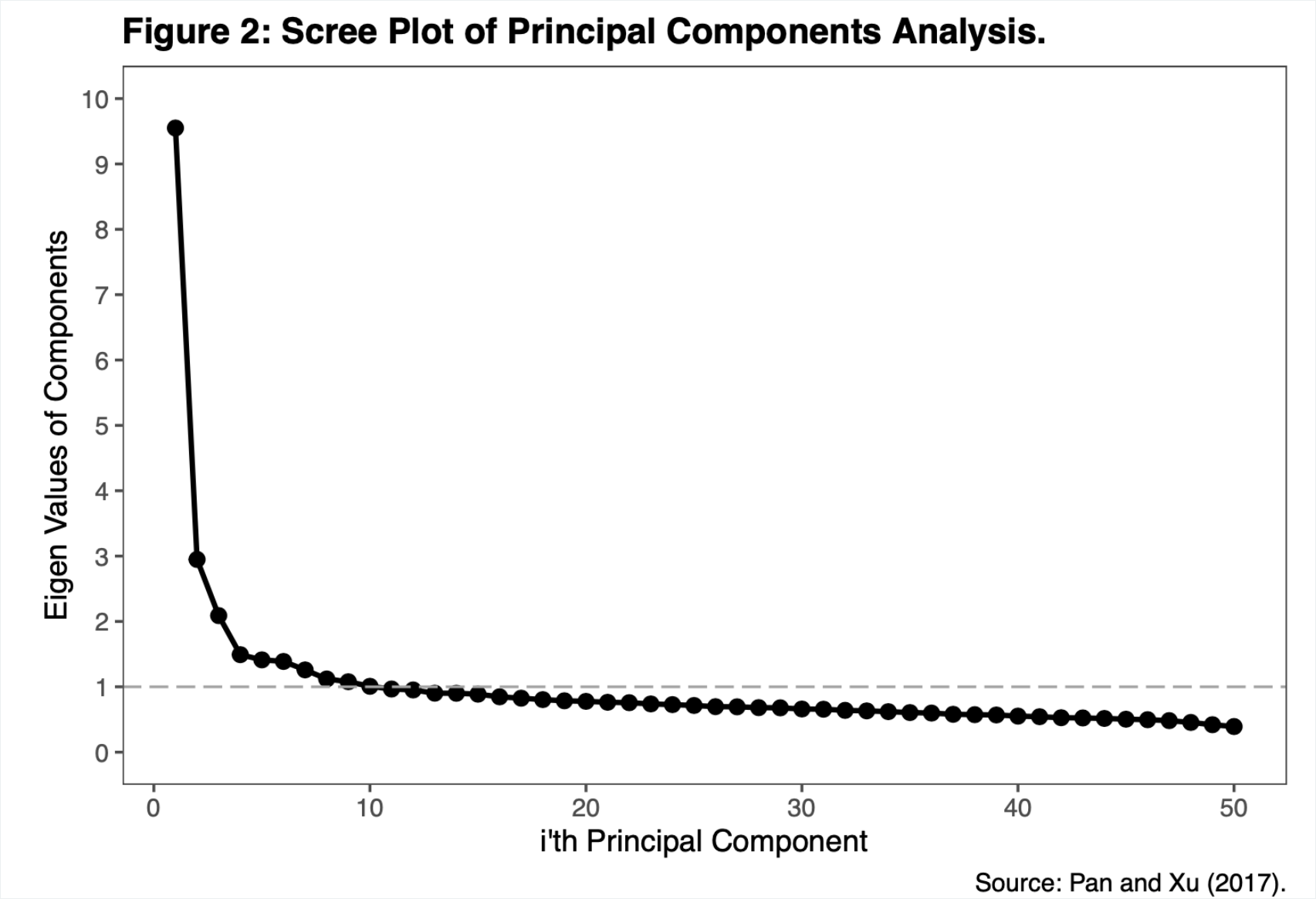
R Code

```
### Load packages
library(tidyverse)

### Load dataset
pan_xu <- read_csv(file = "Pan2018.csv")

## Formula
f <- ~q1 + q2 + q3 + q4 + q5 + q6 + q7 + q8 + q9 + q10 +
                q11 + q12 + q13 + q14 + q15 + q16 + q17 + q18 + q19 + q20 +
                q21 + q22 + q23 + q24 + q25 + q26 + q27 + q28 + q29 + q30 +
                q31 + q32 + q33 + q34 + q35 + q36 + q37 + q38 + q39 + q40 +
                q41 + q42 + q43 + q44 + q45 + q46 + q47 + q48 + q49 + q50

### PCA (canned function)
pca_01 <- prcomp(formula = f, data = pan_xu , center = TRUE , scale = TRUE , na.action = na.omit )

### Scree Plot
screeplot(pca_01)
```

Python Code

Python Code

```
### Import Libraries
import pandas as pd
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt

### Load dataset
pan_xu = pd.read_csv("Pan2018.csv")

### Selecting only the columns for PCA
columns = ['q1', 'q2', 'q3', 'q4', 'q5', 'q6', 'q7', 'q8', 'q9', 'q10',
'q11', 'q12', 'q13', 'q14', 'q15', 'q16', 'q17', 'q18', 'q19', 'q20',
'q21', 'q22', 'q23', 'q24', 'q25', 'q26', 'q27', 'q28', 'q29', 'q30',
'q31', 'q32', 'q33', 'q34', 'q35', 'q36', 'q37', 'q38', 'q39', 'q40',
'q41', 'q42', 'q43', 'q44', 'q45', 'q46', 'q47', 'q48', 'q49', 'q50']

X = pan_xu[columns].dropna()

### Standardize the data
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

### Implement PCA
pca = PCA()
pca.fit(X_scaled)

### Scree Plot
plt.figure(figsize=(10, 5))
plt.plot(range(1, len(pca.explained_variance_ratio_) + 1), pca.explained_variance_ratio_)
plt.title('Scree Plot')
plt.xlabel('Principal Components')
plt.ylabel('Variance Explained')
plt.show()
```



Function `screeplot`.

- How to choose the number of relevant components?: Rules of Thumbs + Expert Knowledge
  1. **Rule of one:** explore eigenvalues greater than one.
  2. **Elbow rule:** on the scree-plot, look for a point at which the proportion of variance explained by each subsequent principal component drops off.
  3. **Threshold rule:** set a threshold, say 80%, and stop when the first k components account for a percentage of total variation greater than this threshold (Jolliffe 2002).

Table 2: Explanatory Power of the Components

Component	Eigen Value	Prop. of Variance	Cumulative Sum	% of Increase
1	9.550	19.101	19.101	NA
2	2.949	5.898	24.999	30.877
3	2.090	4.179	29.178	16.719
4	1.492	2.985	32.163	10.230
5	1.413	2.825	34.988	8.785
6	1.388	2.777	37.765	7.937
7	1.259	2.518	40.283	6.668
8	1.120	2.240	42.523	5.561
9	1.078	2.156	44.679	5.070
10	1.008	2.016	46.695	4.512

More advanced funcions

- How to interpret the components?: Expert knowledge and some tricks:
  1. Explore which features are most related to each component (`check_itemscale` does an automatic classification based on the values of factor loadings).
  2. Examine the biplot (many similar functions, `fviz_pca_biplot` is really cool).

R Code

```
### Exploration of Components and Features
pan_xu %>%  dplyr::select(q1:q50) %>%
parameters::principal_components() %>%
performance::check_itemscale()

### Biplot (too messy!)
pca_01 %>%
factoextra::fviz_pca_biplot()

### A Variation of Biplot: feature loadings/contributions
pca_01 %>%
fviz_pca_var(col.var = "contrib")
```

3/ Appendix (Not required)

Notation

Dimension reduction: $p >> M$

$$\underbrace{\mathbf{X}}_{N \times p} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \cdots & x_{Np} \end{pmatrix} \qquad \underbrace{\mathbf{Z}}_{N \times M} = \begin{pmatrix} z_{11} & z_{12} & \cdots & z_{1M} \\ z_{21} & z_{22} & \cdots & z_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ z_{N1} & z_{N2} & \cdots & z_{NM} \end{pmatrix}$$

Notation

- $\mathbf{x}_i \in \mathbb{R}^p$: i'th high-dimensional observation (let's assume that it is demeaned (centered) such that its mean is zero)
- $\mathbf{z}_i = [z_{i1}, \ldots, z_{iM}] \in \mathbb{R}^M$: i'th low-dimensional representation
- $Z_m = [z_{1m}, \ldots, z_{Nm}] \in \mathbb{R}^N$: m'th component of all the low-dimensional vectors

Factorization of a Matrix

- Eigen (spectral) decomposition: for a symmetric real $n \times n$ matrix $\mathbf{B}$

$$\mathbf{B} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1} \quad \text{where} \quad \mathbf{\Lambda} = diag(\lambda_1, \ldots, \lambda_n)$$

  - eigen values: $\lambda_1, \cdots, \lambda_n$
    - $\lambda_1 \times \cdots \times \lambda_n = det(\mathbf{B})$
    - $\lambda_1 + \cdots + \lambda_n = trace(\mathbf{B})$
  - eigen vectors: $\mathbf{V} = [v_1, \cdots v_n]$ (i.e., $\mathbf{B}v_i = \lambda_i v_i$)
  - orthonormality: $\mathbf{V}^T\mathbf{V} = \mathbf{I}_n$, $\mathbf{V}^{-1} = \mathbf{V}^T$

- Singular value decomposition: for any real $m \times n$ matrix $\mathbf{A}$ (i.e., generalization of eigen-decomposition)

$$\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^T \quad \text{where} \quad \mathbf{D} = diag(d_1, \ldots, d_n)$$

  - singular values of $\mathbf{A}$: $d_1 \geq \cdots \geq d_n \geq 0$
    - The number of non-zero singular values = rank of $\mathbf{A}$
  - orthogonal matrix $\mathbf{U}$ whose columns span the column space of $\mathbf{A}$
  - orthogonal matrix $\mathbf{V}$ whose columns span the row space of $\mathbf{A}$
  - $\mathbf{U}^T\mathbf{U} = \mathbf{U}\mathbf{U}^T = \mathbf{I}_m$ and $\mathbf{U}^T = \mathbf{U}^{-1}$
  - $\mathbf{V}^T\mathbf{V} = \mathbf{V}\mathbf{V}^T = \mathbf{I}_n$ and $\mathbf{V}^T = \mathbf{V}^{-1}$

Principal Components

- An $n \times p$ matrix of predictors: $\mathbf{X}$
- Centered (i.e., mean zero) predictors: $\widetilde{\mathbf{X}}$ where $\widetilde{X_k} = X_k - mean(X_k)$
- Sample covariance matrix: $\frac{1}{n}\widetilde{\mathbf{X}}^T\widetilde{\mathbf{X}}$ (i.e., symmetric and square matrix)
- Singular value decomposition of $\widetilde{\mathbf{X}} = \mathbf{UDV}^T$ implies:

$$\widetilde{\mathbf{X}}^T\widetilde{\mathbf{X}} = \mathbf{VD}^2\mathbf{V}^T$$

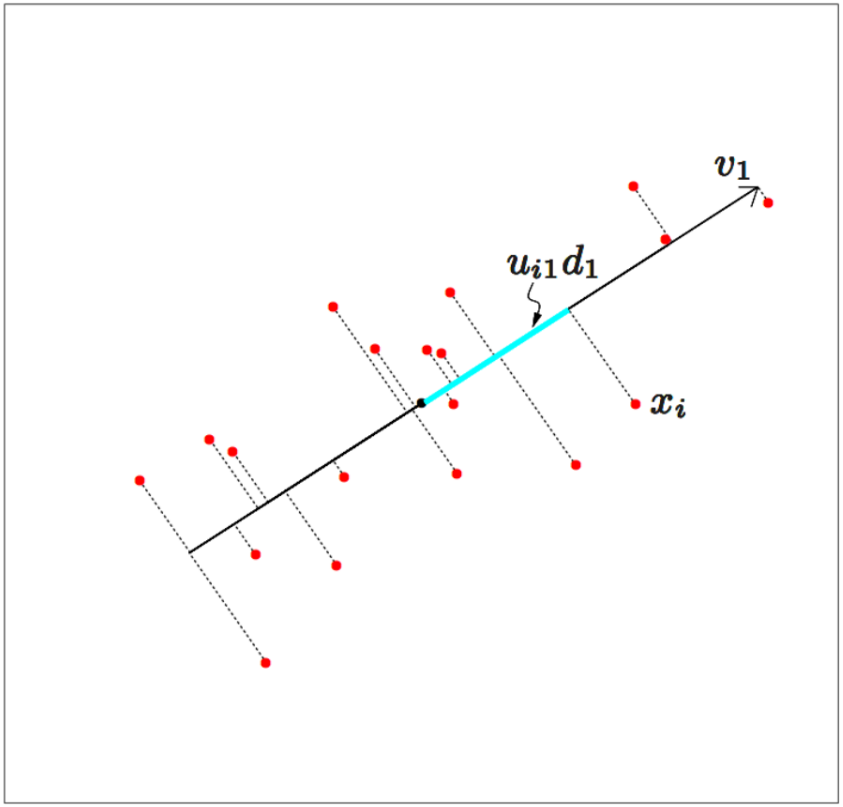
  - This is the eigen decomposition of $\widetilde{\mathbf{X}}^T\widetilde{\mathbf{X}}$
- Columns of $\mathbf{V}$ (i.e., eigen vectors of the sample covariance matrix) are called principal directions (or principal axes) of $\widetilde{\mathbf{X}}$ (see Slide 8)
- $Z_k = \tilde{\mathbf{X}}v_k = u_k d_k$ is called *k*th principal component (Note: $\tilde{\mathbf{X}}\mathbf{V} = \mathbf{UDV}^\mathsf{T}\mathbf{V} = \mathbf{UD}$)
- Variances of principal components:

$$\mathbb{V}(Z_1) = \frac{d_1^2}{n} \geq \mathbb{V}(Z_2) = \frac{d_2^2}{n} \geq \cdots \geq \mathbb{V}(Z_p) = \frac{d_p^2}{n}$$

- Recall that Ridge regression shrinks these last directions (corresponding to the directions that explain the variance in data the least) the most

Geometry: Singular Value Decomposition

$$\mathbf{X} = \underbrace{\mathbf{UD}}_{\text{principal component}} \overbrace{\mathbf{V}^T}^{\text{PC directions}} \quad \text{where} \quad \mathbf{D} = diag(d_1, \ldots, d_n)$$



$u_{i1}d_1$ measure distance along the line from the origin $(= z_{i1})$

PCA: Maximizing the variance of the projected data

- Denote a unit vector $\mathbf{v}_1 \in \mathbb{R}^p$ so that $\mathbf{v}_1^\top \mathbf{v}_1 = 1$
- A projection of each data point $\mathbf{x}_i \in \mathbb{R}^p$ onto this vector will be $\mathbf{v}_1^\top \mathbf{x}_i$
  - Why? $\text{proj}_{\mathbf{v}_1}(\mathbf{x}_i) = \frac{\mathbf{v}_1^\top \mathbf{x}_i}{\mathbf{v}_1^\top \mathbf{v}_1}{}_{=1} \mathbf{v}_1 = \overbrace{\underbrace{(\mathbf{v}_1^\top \mathbf{x}_i)}_{\text{scalar value} \equiv z_{i1}} \mathbf{v}_1}^{\text{projection vector}}$
- The variance of these projected data points is given by

$$\mathbb{V}(Z_1) = \frac{1}{N}\sum_{i=1}^{N} z_{i1}^2 = \frac{1}{N}\sum_{i=1}^{N} \mathbf{v}_1^T \mathbf{x}_i \mathbf{x}_i^T \mathbf{v}_1 = \mathbf{v}_1^T \Sigma \mathbf{v}_1$$

  where $\Sigma = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i \mathbf{x}_i^T$ is the covariance matrix
- The maximization problem becomes

$$\mathcal{L}(\mathbf{v}_1) = \mathbf{v}_1^T \Sigma \mathbf{v}_1 - \lambda_1 \left(\mathbf{v}_1^T \mathbf{v}_1 - 1\right) \Rightarrow \boldsymbol{\Sigma \mathbf{v}_1 = \lambda_1 \mathbf{v}_1}$$

  - $\mathbf{v}_1$ is the eigenvector of the covariance of matrix! (See Slide 6)
  - The variance $\mathbf{v}_1^\top \boldsymbol{\Sigma} \mathbf{v}_1 = \lambda_1 \rightsquigarrow$ PCA sets $\mathbf{v}_1$ to the eigenvector having the largest eigenvalue λ_1 (the variance could be arbitrarily high unless we assume $\mathbf{v}_1^\top \mathbf{v}_1 = 1$)

$\mathbf{v}_1$ (i.e., the first principal component) is defined such that it maximizes the variance of the projected data points on it

Principal Component Analysis (PCA)

- Dimension reduction
- Methods for deriving a low-dimensional representation of features from a large set of variables that contain most of the variation
- $Z_1, Z_2, \ldots, Z_M$ represents $M < p$ linear combinations of original predictors X_j

$$Z_m = \sum_{j=1}^{p} v_{jm} X_j, \quad \text{where} \quad \sum_{j=1}^{p} v_{jm}^2 = 1$$

  - loading vector: $\mathbf{v_m} \in \mathbb{R}^p$ (a unit vector so that $\mathbf{v}_m^T \mathbf{v}_m = 1$)
  - v_{jm}: loadings of mth principal component
  - $\sum_{j=1}^{p} v_{jm}^2 = 1$: normalization constraint to ensure that variance of Z_m doesn't get arbitrarily large (also consistent with $\mathbf{v}_m$ being a orthonormal unit vector)
  - $z_{im} = \mathbf{v_m^T x_i}$: scalar projection (a number!) of data $\mathbf{x_i}$ on to the mth principal component direction
  - $z_{im}\mathbf{v_m}$ will be the projection vector in the direction of mth principal component