Section III: Principal Component Analysis

450C

Stanford University

Department of Political Science

Toby Nowacki

Overview

1. Overview
2. Big Picture: Classification
3. PCA: Intuition
4. PCA: Mechanics
5. PCA: Implementation

Big Picture: Classification

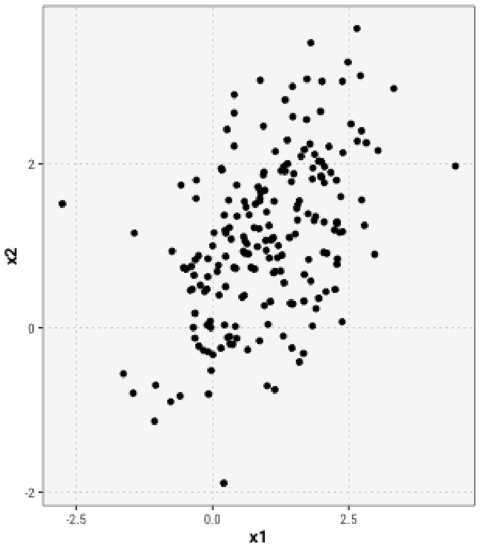
- We're leaving the realm of causal inference
- The goal is to develop tools to accurately label and classify different observations
- Difference between supervised and unsupervised learning
- **unsupervised**: classify, categorise and cluster data
  - principal components analysis;
  - factor analysis;
  - k-means clustering;
  - scaling
- **supervised**: prediction
  - regression;
  - random forests;
  - LASSO;
  - support vector machines;
  - neural networks

PCA: Intuition

- Introduction to **unsupervised** problems
- Useful if our data is stretched across many, many dimensions (covariates)
- Dimensionality reduction technique
- **Examples**
    - How can we order Democratic congressmen from most liberal to most conservative?
    - How can we rank vice-presidential candidates on different dimensions?
    - How can we classify speeches or votes?

PCA: Intuition (cont'd)

- Suppose we have the following two-dimensional data, and want to reduce it to just one dimension:



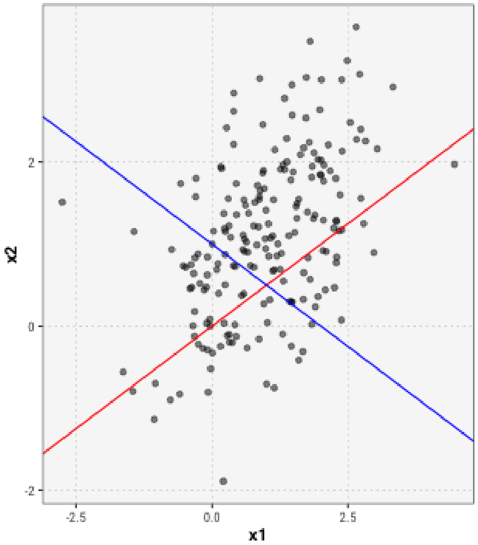
PCA: Intuition (cont'd)

Breakout activity:

Just by intuition, how would you reduce these points down to one dimension?

PCA: Intuition (cont'd)

Which of these lines would be a better fit?



PCA: Intuition (cont'd)

Summary

- The idea behind PCA is to pick the vector through the dimensions along which most the variance in the data is represented
- That way, we retain as much information as possible!
- Conversely, we minimise the reconstruction error -- because we maximise the amount of information that we retain.

PCA: Mechanics

- **Setup**
  - Matrix $\mathbf{X}$ with dimensions $n \times p$.
  - Objective is to reduce matrix to K dimensions.
  - PCA dimensions denoted by $\mathbf{w}_k$
  - Each data point reconstructed by observation-specific weight z_{ik} on dimensions $\mathbf{w}_k$.

$$\hat{\mathbf{x}}_i = \sum_{k=1}^{K} z_{ik} \mathbf{w}_k$$

- **Objective**
  - Pick $\mathbf{w}_k, z_{ik}$ as to minimise avg. reconstruction error:

$$\min_{\mathbf{w}, z_{ik}} \frac{1}{N} \sum_{i=1}^{N} ||\mathbf{x}_i - \sum_{k=1}^{K} z_{ik} \mathbf{w}_k||^2$$

PCA: Mechanics (cont'd)

- Following a lot of algebra, we can show that

$$w_k^T \Sigma \mathbf{w}_k = \lambda_k$$

such that $\mathbf{w}_k^*$ is equal to the k th eigenvector of Σ, and $z_{ik}^* = \mathbf{w}_k^T \mathbf{x}_i$.

- Why does this work?

- Remember that $\mathbf{A}\mathbf{x} = \lambda \mathbf{x}$?

    - The eigenvector $\mathbf{x}$ points out the vector in multidimensional space along which most of the variance-covariance matrix (Σ) can be captured.
    - Geometrically, we're rotating the co-ordinate system as to remove the correlation between the covariates.

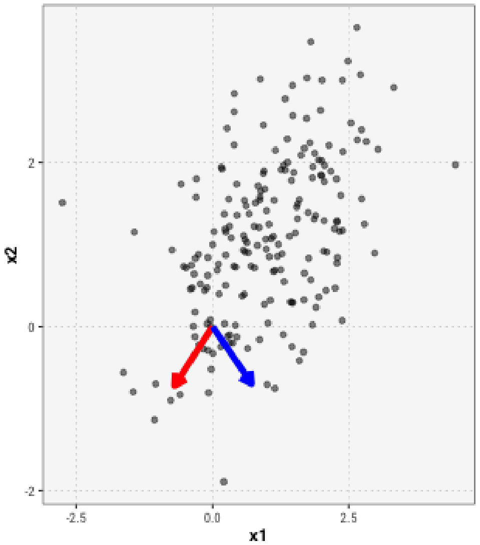
PCA: Mechanics (cont'd)

$$\Sigma = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$

$$\mathbf{x}_1 = \begin{bmatrix} 0.707 \\ 0.707 \end{bmatrix}$$

$$\Sigma \mathbf{x}_1 = \begin{bmatrix} 1.06 & 1.06 \end{bmatrix}$$

PCA: Mechanics (cont'd)



PCA: Mechanics (cont'd)

- Singular Value Decomposition to get eigenvectors and eigenvalues
- In R, `eigen` implements this

PCA: Implementation

- Canned function `prcomp( )` for PCA

```r
pca <- prcomp(obs, scale = FALSE, center = FALSE)
pca
```

```
## Standard deviations (1, .., p=2):
## [1] 1.9327988 0.7419486
##
## Rotation (n x k) = (2 x 2):
##           PC1        PC2
## V1 -0.6960578  0.7179858
## V2 -0.7179858 -0.6960578
```

PCA: Implementation (cont'd)

```
covmat ← cov(as.matrix(obs))
covmat
```

```
##           V1        V2
## V1 1.1037814 0.4993894
## V2 0.4993894 0.9868873
```

```
eigen_mat ← eigen(covmat)
eigen_mat
```

```
## eigen() decomposition
## $values
## [1] 1.5481324 0.5425363
##
## $vectors
##            [,1]       [,2]
## [1,] -0.7470755  0.6647392
## [2,] -0.6647392 -0.7470755
```

PCA: Implementation (cont'd)

- Roll-call example: We know that legislators in parliamentary systems predominantly vote along party lines
- But 2017-2019 UK Parliament was unusual: many, many rebellions with respect to Brexit
- Have a $n \times p$ votes matrix with n MPs and p divisions.
- Code an `Aye` vote as `1`, a `No` vote as `-1`, and an abstention as `0`.
- Use PCA to reduce dimensionality!

PCA: Implementation (cont'd)

Summary

- PCA is a **dimension reduction** technique
- We use a simple trick in linear algebra to summarise a matrix as a vector
- Convenient, but often not ideal:
  - Interpretation of principal components?
  - Information loss
  - No easy way for categorical classification
- Next couple of weeks: more classification methods