Principle Components Analysis (PCA)

Rebecca C. Steorts, Duke University

STA 325, Chapter 10 ISL

Agenda

- Supervised versus unsupervised learning (Chapter 10 ISLR)
- Recap of information retrieval
- Relation to PCA
- Idea behind PCA
- Application to NYTimes data set
- Multidimensional scaling
- Scree plots

Supervised versus Unsupervised Learning

Most statistical learning techniques fall into two categories:

1. Supervised
2. Unsupervised

Supervised versus Unsupervised Learning

Most of the methods in ISLR are **supervised learning techniques**.

This means that for each predictor x_i $i = 1, \ldots, n$ there is an associated response variable y_i,

Our goal is to fit a model relating the response (y) to the predictors (x) such that we can accurately predict future responses (prediction) or such that we can better understand the relationship between the response and the predictor (inference).

Examples of this include regression, logistic regression, boosting, and support vector machines.

Supervised versus Unsupervised Learning

Chapter 10 covers unsupervised learning techniques.

Unsupervised learning covers a more challenging situation where for every observation x_i that we observe, we do not observe a response variable y_i.

Therefore, we cannot fit a linear regression model since we don't have a reponse variable!

In this setting, we lack the supervision of a response y_i to guide the x_i for model fitting, so we develop other methods to guide our analysis.

Typically we are guided by the data!

Supervised versus Unsupervised Learning

One statistical learning tool that we will learn about is clustering.

The goal of clustering is to ascertain, on the basis of $x_1, \ldots, x_n$, where the observations fall into relatively distinct groups.

Another goal may just be to do an exploratory data analysis on $x_1, \ldots, x_n$ and perform some dimension reduction and visualize the features. This is known as principle components analysis.

Information retrieval

- Recall that information retrieval systems often represent documents as what are called **bags of words**.
- Such documents are represented as vectors
- Each component counts how many times each word in the dictionary appears in the text.
- This throws away information about word order.
- Part of the representation of one document might look like:

a	abandoned	abc	ability	able	about	above	abroad	absorbed	absorbing	abstract
43	0	0	0	0	10	0	0	0	0	1

and so on through to `zebra''`,zoology'', "zygote", etc. to the end of the dictionary.

Information retrieval

These bag-of-word vectors have three outstanding properties:

1. Most words do not appear in most documents; the bag-of-words vectors are very **sparse** (most entries are zero).
2. Small number of words appear many times in almost all documents.
   - (Examples: “the”, “is”, “of”, “for”, “at”, “a”, “and”, “here”, “was”, etc.)
3. Many words’ counts are correlated with some but not all other words.

Takeaway

- Don't get much value from keeping around *all* the words.
- Better off projecting down to a smaller number of new variables.
- Project partially since the words mean slightly different things.
- This is *exactly* what principal components analysis does and it's very useful as an exploratory data analysis exercise for dimension reduction.

Principal components analysis (PCA)

- PCA is a tool for exploratory data analaysis and dimension reduction.
- Take large set of correlated variables and replace with smaller number that collectively explain most of the variability.
- The PC directions are directions in the feature space along which the original data are high variable.
- PCA is an unsupervised learning tool – since it involves the features X and no response Y

Why is it useful?

Suppose that we wish to visualize n observations with measurements on a set of p features, $X_1, \ldots, X_p$ as part of an exploratory data analysis.

We could do this by examining two-dimensional scatterplots of the data, each of which contains the n observations' measurements on two of the features.

However, there are $p = p(p-1)/2$ such scatterplots

For example, with $p = 10$ there are 45 plots!

PCA seeks to to find a low-dimensional representation of the data that captures as much of the information as possible.

PCA

PCA seeks a small number of dimensions that are as interesting as possible, where the concept of interesting is measured by the amount that the observations vary along each dimension.

Each of the dimensions found by PCA is a linear combination of the p features.

How many principal components are there?

There are a total number of $\min(n-1, p)$ principal components.

For some reason R prints out the n principal component (and I'm not sure why it does this)!

PCA

The first principal components of a set of features $X_1, \ldots, X_p$ is the normalized linear combination of the features

$$z_{i1} = \phi_{11} X_1 + \phi_{21} X_2 + \cdots + \phi_{p1} X_p = \sum_{j=1}^{p} \phi_{i1} X_i$$

that has the largest variance.

By normalized, we mean that $\sum_{i=1}^{p} \phi_{i1}^2 = 1$.

- $\phi_{11}, \ldots, \phi_{p1}$ are the loadings of the first principal component.
- Together the loadings make up the the principle components loading vector $\phi_1 = (\phi_{11}, \ldots, \phi_{p1})^T$

Finding the first PC

Given a data set $X_{n\times p}$, how do we compute the first principle component?

Since we're only interested in the variance, let us assume that each of the variables in $X_{n\times p}$ has been centered to have mean zero.

We then look for the linear combination of the sample feature values of the form

$$z_{i1} = \phi_{11}x_{i1} + \phi_{21}x_{i2} + \ldots + \phi_{p1}x_{ip} = \sum_{j=1}^{p} \phi_{i1}x_{i1}. \qquad (1)$$

that has the largest sample variance subject to the constraint $\sum_{j=1}^{p} \phi_{j1}^2 = 1$.

Finding the first PC

Finding the first principle component loading vector solves the optimatization problem:

$$\max_{\phi_{11},\ldots,\phi_{p1}} \left\{ \frac{1}{n} \sum_{i=1}^{n} (\sum_{j=1}^{p} \phi_{j1} x_{ij})^2 \right\} \quad \text{subject to} \quad \sum_{j=1}^{p} \phi_{j1}^2 = 1 \qquad (2)$$

Using equation 1, we can write the objective in equation 2 as

$$\frac{1}{n} \sum_{i=1}^{n} z_{i1}^2$$

Note: Since $\frac{1}{n} \sum_{i=1}^{n} x_{ij} = 0$, the average of $z_{11}, \ldots, z_{n1}$ is also zero.

Thus, the objective that we are maximizing in equation 2 is just the sample variance of the n values of z_{i1}

where $z_{11}, \ldots, z_{n1}$ are the scores of the first principal component.

PC

The optimization problem in equation 2 is beyond the scope of this course, but it can be solved by what is called an eigen-decomposition (it requires linear algebra).

Geometric Interpreation

There is a very nice geometric interpretation for the first principal component.

The loading vector ϕ_1 with elements $\phi_{11}, \ldots, \phi_{p1}$ defines a direction in the feature space along which the data vary the most.

If we project the n data points onto this direction, then the projected values are the principle component scores $z_{11}, \ldots, z_{n1}$.

New York Times Example

- Have news stories taken from the New York *Times* Annotated Corpus
- Consists of about 1.8 million stories from the *Times*, from 1987 to 2007.
- Stories have been hand-annotated by humans with standardized machine-readable information about their contents.
- From this corpus, have randomly selected 57 stories about art and 45 stories about music.[1]

[1] Turned them into a bag-of-words data frame, one row per story, one column per word; plus an indicator in the first column of whether the story is one about art or one about music.

PCA for NYTimes

```
load("pca-examples.Rdata")
  # The workspace now contains:
  # nyt.frame.raw: a data frame
  #with counts of words (columns) in stories (rows)
  # first column, "class.labels",
  # is a factor indicating "art"
  # or "music"
  # nyt.frame: the same, with word
 # counts suitably normalized and weighted
  # art: vector where each row is itself a
 # vector of words giving the
  # actual stories about art, with
 # punctuation removed, etc.
  # music: ditto
  # Some miscellaneous functions used to
 # create the data sets (see end of
  # this file for gory details)
```

PCA for NYTimes

```
# How big is it?
dim(nyt.frame)
```

```
## [1]  102 4432
```

```
# Remember: rows = stories, columns = words (except the first column, which
  # is the type of story)
# What are some typical words?
colnames(nyt.frame)[sample(ncol(nyt.frame),30)]
```

```
##  [1] "penchant"  "brought"   "structure" "willing"   "yielding"
##  [6] "bare"      "school"    "halls"     "challenge" "step"
## [11] "largest"   "lovers"    "intense"   "borders"   "mall"
## [16] "classic"   "conducted" "mirrors"   "hole"      "location"
## [21] "desperate" "published" "head"      "paints"    "another"
## [26] "starts"    "familiar"  "window"    "thats"     "broker"
```

```
# A little bit of the data
signif(nyt.frame[sample(nrow(nyt.frame),5),sample(ncol(nyt.frame),10)],3)
```

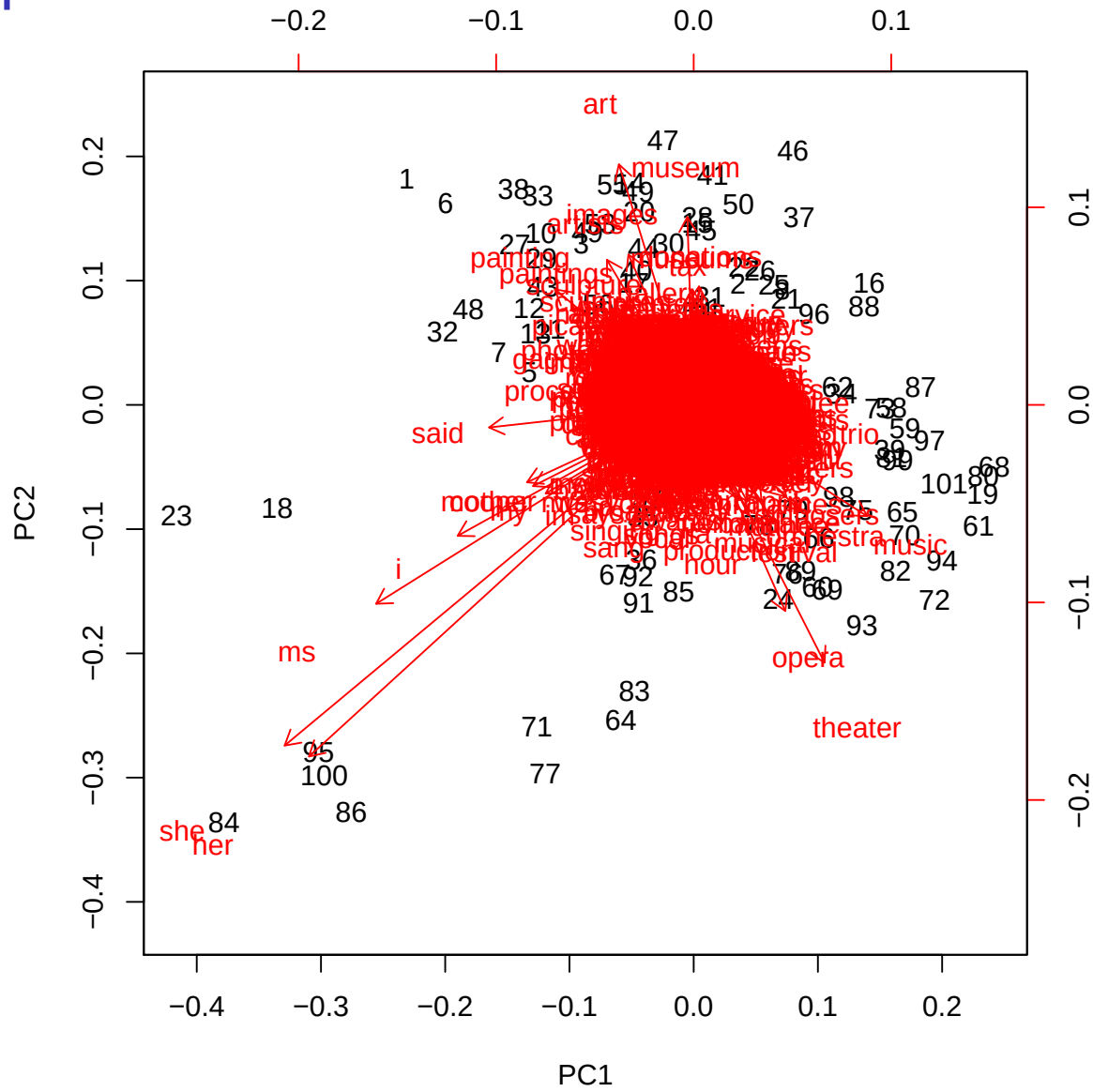
```
##    jacket patch tapes   want   ford failed condemn intentional confined
## 24      0     0     0 0.0000 0.0000 0.0000       0           0        0
## 2       0     0     0 0.0275 0.0704 0.0000       0           0        0
## 85      0     0     0 0.0482 0.0000 0.0000       0           0        0
## 59      0     0     0 0.0000 0.0000 0.0000       0           0        0
## 76      0     0     0 0.0000 0.0000 0.0215       0           0        0
##    destroyed
## 24         0
## 2          0
## 85         0
## 59         0
## 76         0
```

PCA for NYTimes

- Need to omit the first column in the first command because it contains categorical variables, and PCA doesn't apply to them.
- The second command just picks out the matrix of projections of the variables on to the components.
- Called rotation because it can be thought of as rotating the coordinate axes in feature-vector space.

```
# Do PCA
nyt.pca = prcomp(nyt.frame[,-1])
  # Omit the first column of class labels
# Extract the actual component directions/weights for ease of reference
nyt.latent.sem = nyt.pca$rotation
```

Bi-plot



- This shows the two first PC for the NYtimes data.
- Blue scores represent the scores for the first two PC's.
- The red arrows represent the first two PC loading vectors.
- Here, a bi-plot is not useful at all since the data is too high dimensional. (See Figure 10.1 for where a bi-plot is useful.)

Look at the leading components

```
# What are the components?
# Show the 30 words with the biggest positive loading on PC1
signif(sort(nyt.latent.sem[,1],decreasing=TRUE)[1:30],2)
```

```
##       music        trio     theater   orchestra   composers       opera
##       0.110       0.084       0.083       0.067       0.059       0.058
##    theaters           m    festival        east     program           y
##       0.055       0.054       0.051       0.049       0.048       0.048
##      jersey     players   committee      sunday        june     concert
##       0.047       0.047       0.046       0.045       0.045       0.045
##    symphony       organ     matinee   misstated instruments           p
##       0.044       0.044       0.043       0.042       0.041       0.041
##         X.d       april      samuel        jazz     pianist     society
##       0.041       0.040       0.040       0.039       0.038       0.038
```

```
  # biggest negative loading on PC1, the other end of that scale
signif(sort(nyt.latent.sem[,1],decreasing=FALSE)[1:30],2)
```

```
##       she       her        ms         i      said    mother    cooper
##    -0.260    -0.240    -0.200    -0.150    -0.130    -0.110    -0.100
##        my  painting   process paintings        im        he       mrs
##    -0.094    -0.088    -0.071    -0.070    -0.068    -0.065    -0.065
##        me  gagosian       was   picasso     image sculpture      baby
##    -0.063    -0.062    -0.058    -0.057    -0.056    -0.056    -0.055
##   artists      work    photos       you    nature    studio       out
##    -0.055    -0.054    -0.051    -0.051    -0.050    -0.050    -0.050
##      says      like
##    -0.050    -0.049
```

```
  # Ditto for PC 2
signif(sort(nyt.latent.sem[,2],decreasing=TRUE)[1:30],2)
```

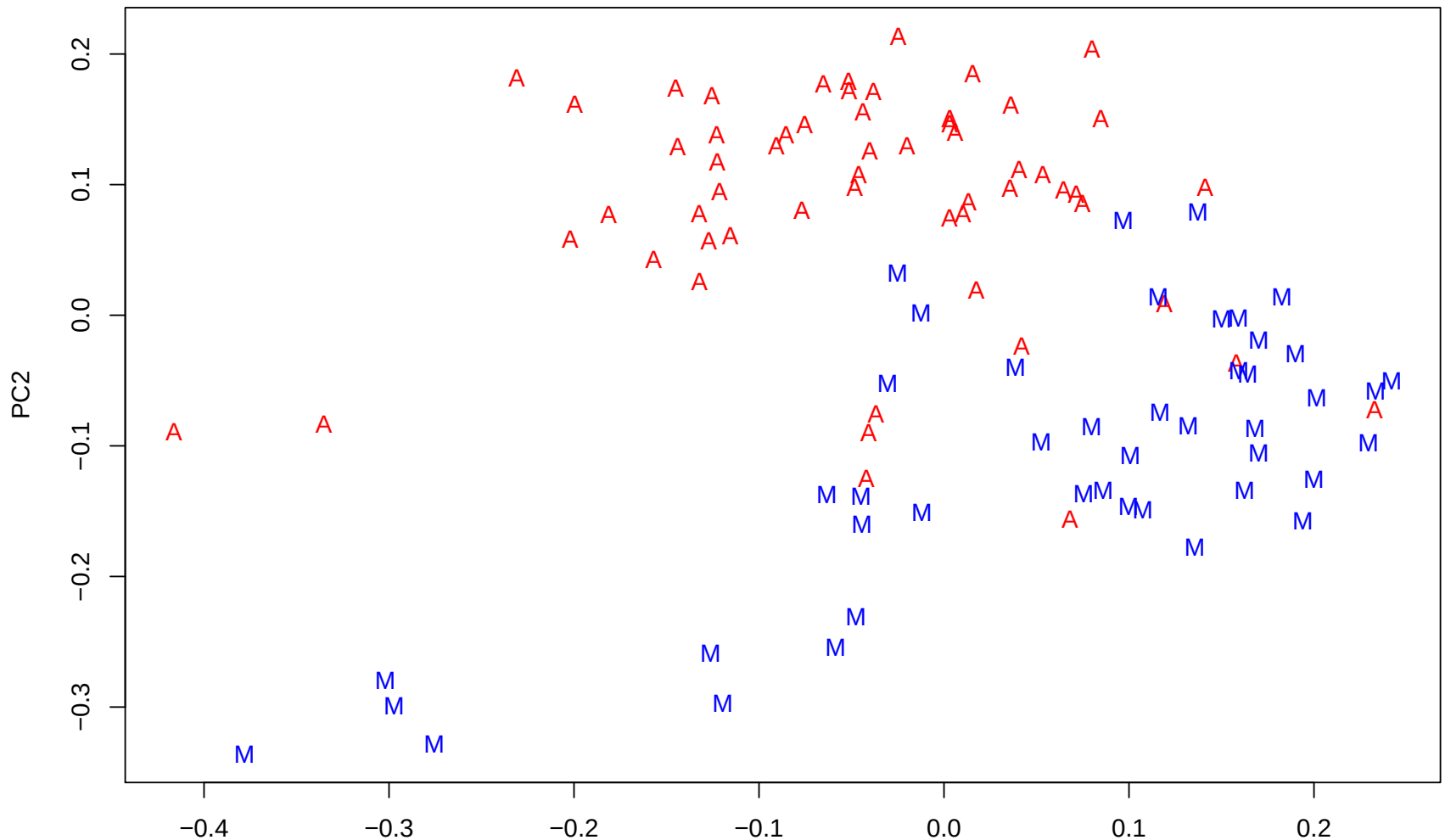
Look at the leading components

- These are the thirty words with the largest positive and negative projections on to the first component[2].
- Words with positive projections mostly associated with music
- Those with negative components with visual arts.
- Why do we see women and mothers here?

[2]Which direction is positive and which is negative depend on internal choices in the PCA algorithm

PCA for NYTimes

```
# Plot the projection of the stories on to the first 2 components
  # Establish the plot window
plot(nyt.pca$x[,1:2],type="n")
  # Arts stories with red As
points(nyt.pca$x[nyt.frame[,"class.labels"]=="art",1:2],pch="A",col="red")
  # Music stories with blue Ms
points(nyt.pca$x[nyt.frame[,"class.labels"]=="music",1:2],pch="M",col="blue")
```



How well is PCA doing?

Even though we have gone from 4431 dimensions to 2, and thrown away a lot of information, we could draw a line across this plot and have most of the art stories on one side of it and all the music stories on the other.

If we let ourselves use the first four or five principal components, we'd still have a thousand-fold savings in dimensions, but we'd be able to get almost-perfect separation between the two classes.

This is a sign that PCA is really doing a good job at summarizing the information in the word-count vectors, and in turn that the bags of words give us a lot of information about the meaning of the stories.

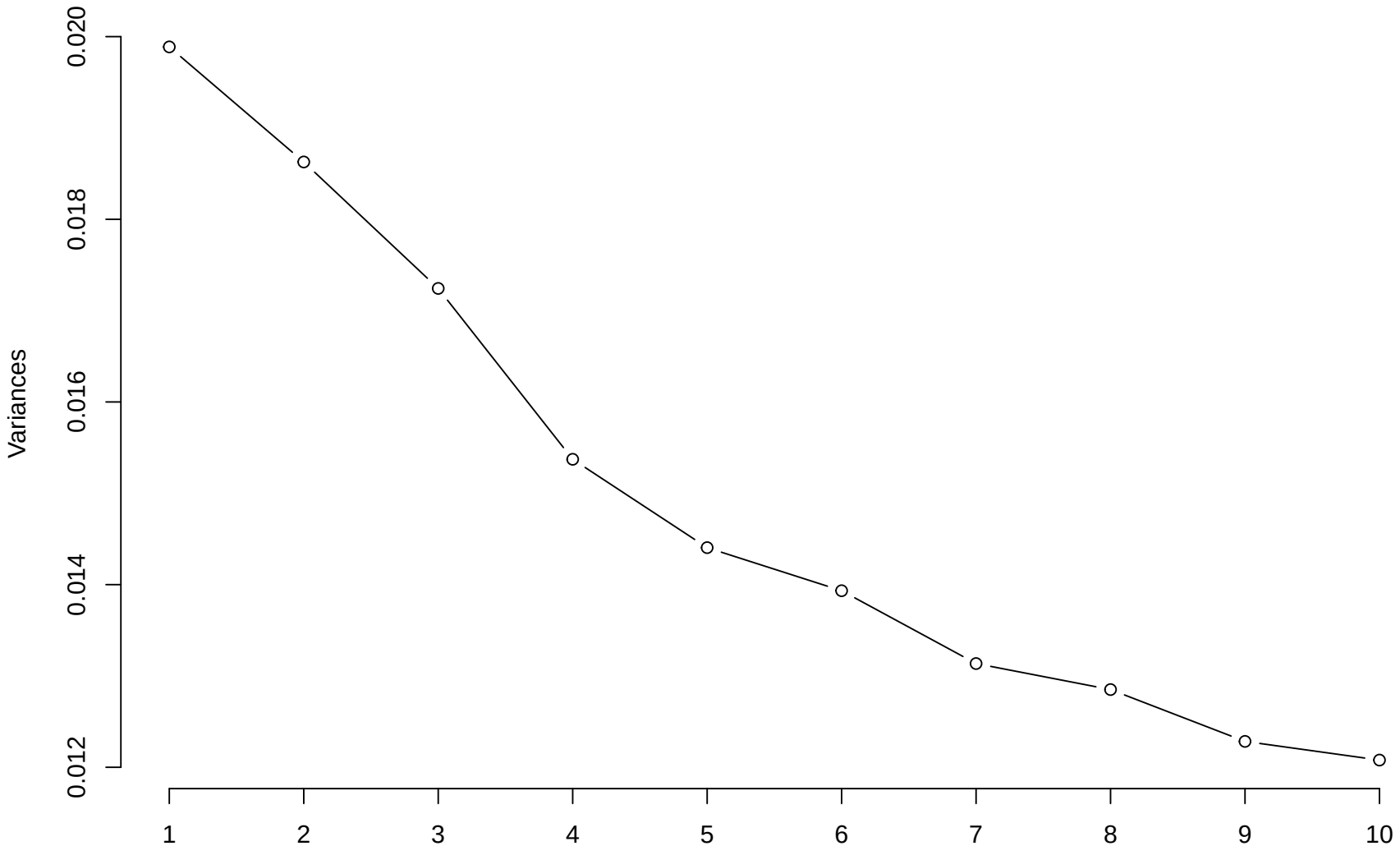
The figure also illustrates the idea of multidimensional scaling — finding low-dimensional points to represent high-dimensional data by preserving the distances between the points.

Scree Plot

- We can figure out the number of principal components by fitting what's called a scree plot.
- Choose the smallest number of principal components that are required such that an adequate amount of variability is explained.
- We look for the point at which the proportion of variance explained by each subsequent principal drops off.
- This is called the elbow of the scree plot.
- These plots are application specific and ad-hoc.

Scree Plot

```
plot(nyt.pca,type="l", main="")
```



Summary

- Visual analysis of PCA is adhoc (such as the scree plot).
- In fact, the question of how many PC to choose is very ill-defined in practice and depends on a specific application.
- In practice, we tend to look at the first few PC in to find interesting patterns in the data.
- If not interesting patterns are found, then further PC are unlikely to be of interest.
- We could use PC in the context of regression (see Section 6.3.1 for further details and this is more principled).