Math 152: Topics in Data Science

Thang Huynh, UC San Diego

Contact Information

Instructor: Thang Huynh

Email: tlh007@ucsd.edu

Office Hours: 12:00pm-1:00pm MWF at AP&M 6341
(or by appointment)

Course webpage: Syllabus (must read), Exam Schedule, Homework, TA Information, etc.

`www.thanghuynh.io/teaching/math152_spring19/home/`

What is this course about?

- **Prerequisite:** I highly recommend that students are familiar with linear algebra (**MATH 102**), probability theory (**MATH 180A**), and combinatorics. The class will attempt to be self contained (but this is not always possible). Moreover, the class is theoretical, and is devoted to ideas, algorithms, and proofs.

What is this course about?

- **Prerequisite:** I highly recommend that students are familiar with linear algebra (**MATH 102**), probability theory (**MATH 180A**), and combinatorics. The class will attempt to be self contained (but this is not always possible). Moreover, the class is theoretical, and is devoted to ideas, algorithms, and proofs. *Students who are interested in explicit data science applications should not register.*

What is this course about?

- **Prerequisite:** I highly recommend that students are familiar with linear algebra (**MATH 102**), probability theory (**MATH 180A**), and combinatorics. The class will attempt to be self contained (but this is not always possible). Moreover, the class is theoretical, and is devoted to ideas, algorithms, and proofs. *Students who are interested in explicit data science applications should not register.*
- **We will cover among other topics (tentative)**: sampling, finding frequent items, counting distinct elements, general frequency moment estimation, dimensionality reduction, and matrix approximation.

Course Information

There is no course textbook.

- *Foundations of Data Science* by Avrim Blum, John Hopcroft, and Ravindran Kannan.
- *Mining of Massive Datasets* by Jure Leskovec, Anand Rajaraman, Jeff Ullman.
- *Data Stream Algorithms* by Amit Chakrabarti.

Course Information

- Grading Scheme: Best of
  - 25% Midterm 1, 25% Midterm 2, 50% Final
  - 40% Best Midterm, 60% Final Exam

Course Information

- Grading Scheme: Best of
  - 25% Midterm 1, 25% Midterm 2, 50% Final
  - 40% Best Midterm, 60% Final Exam
- Piazza (see course webpage)

Course Information

- Grading Scheme: Best of
  - 25% Midterm 1, 25% Midterm 2, 50% Final
  - 40% Best Midterm, 60% Final Exam
- Piazza (see course webpage)
- Homework: Homework will be assigned about every two weeks (not collected). You should solve these homework problems and discuss them with your TAs during discussion sessions.

Course Information

- Grading Scheme: Best of
  - 25% Midterm 1, 25% Midterm 2, 50% Final
  - 40% Best Midterm, 60% Final Exam
- Piazza (see course webpage)
- Homework: Homework will be assigned about every two weeks (not collected). You should solve these homework problems and discuss them with your TAs during discussion sessions.
- Midterms: There will be two midterms. Most of the problems on these midterms will be similar to the homework assignments. Midterm 1: April 26; Midterm 2: May 24

Course Information

- Grading Scheme: Best of
  - 25% Midterm 1, 25% Midterm 2, 50% Final
  - 40% Best Midterm, 60% Final Exam
- Piazza (see course webpage)
- Homework: Homework will be assigned about every two weeks (not collected). You should solve these homework problems and discuss them with your TAs during discussion sessions.
- Midterms: There will be two midterms. Most of the problems on these midterms will be similar to the homework assignments. Midterm 1: April 26; Midterm 2: May 24
- Exams: There will be two midterm exams and a final exam. You may use one 8.5 x 11 inch page of handwritten notes.
- There will be no makeup exams.

Plan

Week	Contents
1	Linear Algera Review
2	Probability Review
3	Chernoff's Bound
4	Data Stream
5	Data Stream
6	Singular Value Decomposition (SVD)
7	Singular Value Decomposition (SVD)
8	Matrix sampling
9	Matrix Sampling
10	Matrix Sampling
11	Final Exam

Why dimensionality reduction?

- Many sources of data that can be viewed as large vectors or large matrices ($\Longrightarrow$ big data).

Why dimensionality reduction?

- Many sources of data that can be viewed as large vectors or large matrices ($\Longrightarrow$ big data).
- For these vectors, one may project them onto a much smaller dimensional vector space and still preserve their information (approximately).

Why dimensionality reduction?

- Many sources of data that can be viewed as large vectors or large matrices ($\Longrightarrow$ big data).
- For these vectors, one may project them onto a much smaller dimensional vector space and still preserve their information (approximately).
- For the large matrices, one may find narrower matrices, which in some sense are close to the original but can be used more efficiently.

Why dimensionality reduction?

- Many sources of data that can be viewed as large vectors or large matrices ($\Longrightarrow$ big data).
- For these vectors, one may project them onto a much smaller dimensional vector space and still preserve their information (approximately).
- For the large matrices, one may find narrower matrices, which in some sense are close to the original but can be used more efficiently.
- The above two processes are dimensionality reduction.

Dimensionality reduction: A cool example

Let's say you select 20 top political questions in the United States and ask millions of people to answer these questions using a yes or a no.

See `http://www.learnopencv.com/principal-component-analysis/`

Dimensionality reduction: A cool example

Let's say you select 20 top political questions in the United States and ask millions of people to answer these questions using a yes or a no. For example,

1. Do you support gun control?
2. Do you support a woman's right to abortion?

So on and so forth.

See `http://www.learnopencv.com/principal-component-analysis/`

Dimensionality reduction: A cool example

Let's say you select 20 top political questions in the United States and ask millions of people to answer these questions using a yes or a no. For example,

1. Do you support gun control?
2. Do you support a woman's right to abortion?

So on and so forth.
How many different answer sets?

See `http://www.learnopencv.com/principal-component-analysis/`

Dimensionality reduction: A cool example

Let's say you select 20 top political questions in the United States and ask millions of people to answer these questions using a yes or a no. For example,

1. Do you support gun control?
2. Do you support a woman's right to abortion?

So on and so forth.
How many different answer sets? 2^{20} sets.

See http://www.learnopencv.com/principal-component-analysis/

Dimensionality reduction: A cool example

Let's say you select 20 top political questions in the United States and ask millions of people to answer these questions using a yes or a no. For example,

1. Do you support gun control?
2. Do you support a woman's right to abortion?

So on and so forth.
How many different answer sets? 2^{20} sets.
But in practice, you will notice the answer set is much smaller.

See http://www.learnopencv.com/principal-component-analysis/

Dimensionality reduction: A cool example

Let's say you select 20 top political questions in the United States and ask millions of people to answer these questions using a yes or a no. For example,

1. Do you support gun control?
2. Do you support a woman's right to abortion?

So on and so forth.

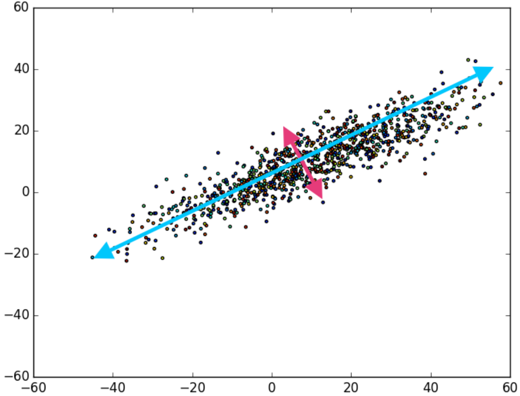
How many different answer sets? 2^{20} sets.

But in practice, you will notice the answer set is much smaller.

> We can ask a single question:
> "Are you a democrat or a republican?"

See `http://www.learnopencv.com/principal-component-analysis/`

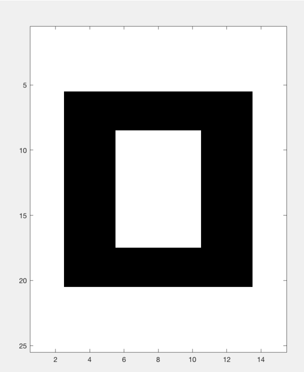
Principal Component Analysis (PCA)



Thanks to Austin G. Walters

Data compression using singular value decompositions

We want transmit the following image, which consists of an array of 15×25 black or white pixels.



See http://www.ams.org/publicoutreach/feature-column/fcarc-svd

Data compression using singular value decompositions

We will represent the image as a 15×25 matrix in which each entry is either a 0, representing a black pixel, or 1, representing white (375 entries).

```
A =

     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1
     1     1     0     0     0     0     0     0     0     0     0     0     0     1     1
     1     1     0     0     0     0     0     0     0     0     0     0     0     1     1
     1     1     0     0     0     0     0     0     0     0     0     0     0     1     1
     1     1     0     0     0     1     1     1     1     1     0     0     0     1     1
     1     1     0     0     0     1     1     1     1     1     0     0     0     1     1
     1     1     0     0     0     1     1     1     1     1     0     0     0     1     1
     1     1     0     0     0     1     1     1     1     1     0     0     0     1     1
     1     1     0     0     0     1     1     1     1     1     0     0     0     1     1
     1     1     0     0     0     1     1     1     1     1     0     0     0     1     1
     1     1     0     0     0     1     1     1     1     1     0     0     0     1     1
     1     1     0     0     0     1     1     1     1     1     0     0     0     1     1
     1     1     0     0     0     1     1     1     1     1     0     0     0     1     1
     1     1     0     0     0     0     0     0     0     0     0     0     0     1     1
     1     1     0     0     0     0     0     0     0     0     0     0     0     1     1
     1     1     0     0     0     0     0     0     0     0     0     0     0     1     1
     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1
     1     1     1     1     1     1     1     1     1     1     1     1     1     1     1
```

Data compression using singular value decompositions

Using SVD, we can represent A as

$$A = \boldsymbol{u}_1 \sigma_1 \boldsymbol{v}_1^T + \boldsymbol{u}_2 \sigma_2 \boldsymbol{v}_2^T + \boldsymbol{u}_3 \sigma_3 \boldsymbol{v}_3^T$$

Data compression using singular value decompositions

Using SVD, we can represent A as

$$A = \boldsymbol{u}_1 \sigma_1 \boldsymbol{v}_1^T + \boldsymbol{u}_2 \sigma_2 \boldsymbol{v}_2^T + \boldsymbol{u}_3 \sigma_3 \boldsymbol{v}_3^T$$

- Each $\boldsymbol{v}_i$ has 15 entries,
- each $\boldsymbol{u}_i$ has 25 entries, and
- there are three singular values σ_i.

Data compression using singular value decompositions

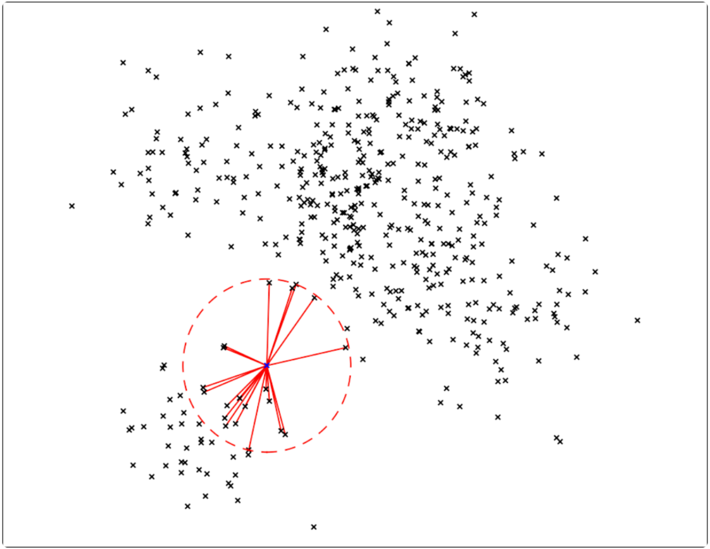
Using SVD, we can represent A as

$$A = \boldsymbol{u}_1 \sigma_1 \boldsymbol{v}_1^T + \boldsymbol{u}_2 \sigma_2 \boldsymbol{v}_2^T + \boldsymbol{u}_3 \sigma_3 \boldsymbol{v}_3^T$$

- Each $\boldsymbol{v}_i$ has 15 entries,
- each $\boldsymbol{u}_i$ has 25 entries, and
- there are three singular values σ_i.

This implies that we may represent the matrix using only 123 numbers rather than the 375 that appear in the matrix, and still preserve all the information of the matrix A.

Nearest neighbor search



Counting distinct elements

How many distinct IP addresses has the router seen? (An IP may have passed once, or many many times.)

Web searchers

Millions of queries / day

- What are the top queries right now?
- Which terms are gaining popularity now?
- What ads should we show for this query and user?