Introduction to Natural Language Processing

BUS 243 F: Spring 2023

Yeabin Moon

Lecture 4

Review

- What have we done?
  - Text representation
    - One-hot encoding
    - Bag-of-words representation
    - TF-IDF
  - Vector reasoning
    - Dot product
    - Cosine similarity
  - Text classification
    - Naïve Bayes approach

Problem of vocabulary

- Frequency based representation has a large vocab
  - Zipf's law: a few words are dominating, and most words (tokens) used once
  - Overfitting is inevitable
- What are we assuming?
- If the frequency of the tokens matters, why not the combinations of words?
  - If so, can we attach some meanings to them?

Goal of today

- Study the latent semantic analysis (LSA)
  - Data Dimensionality reduction technique (PCA)
- Understand the resulting topic vectors
- Textbook is rather ambitious
  - Represent meaning?
- Don’t be scared about the math

Simple Idea, eventually

- We are going to reduce the size of the matrix
  - Entire vocabulary → a set of columns
  - Benefit?
  - Cost?
- The terms “*topic*,” “*semantic*,” and “*meaning*” have used interchangeably in this chapter
  - I am not a fan ☹

Need to tone down

- Frequency based approach fails to catch similar texts with different spellings
- LSA technique tries to solve this problem by constructing topics vectors
- However, “topic” has hardly intrinsic meaning
  - Need to discuss later

This is it!

… can you imagine how you might squash a TF-IDF vector with one million dimensions (terms) down to a vector with 200 or so dimensions (topics)? This is like identifying the right mix of primary colors to try to reproduce the paint color in your apartment so you can cover over those nail holes in your wall.

Thought experiment

- Textbook shows the thought experiment in 4.1.3
  - This is a unicorn version of topic modeling
  - Keep in mind that this is what we want to achieve
  - See whether the PCA actually does this job
  - Let’s see the code

What are we doing?

- We have 6X1 TF-IDF vector
- Forget about the weights now
- We need to decide
  - The number of topics
  - Assign the weights corresponding to the topics

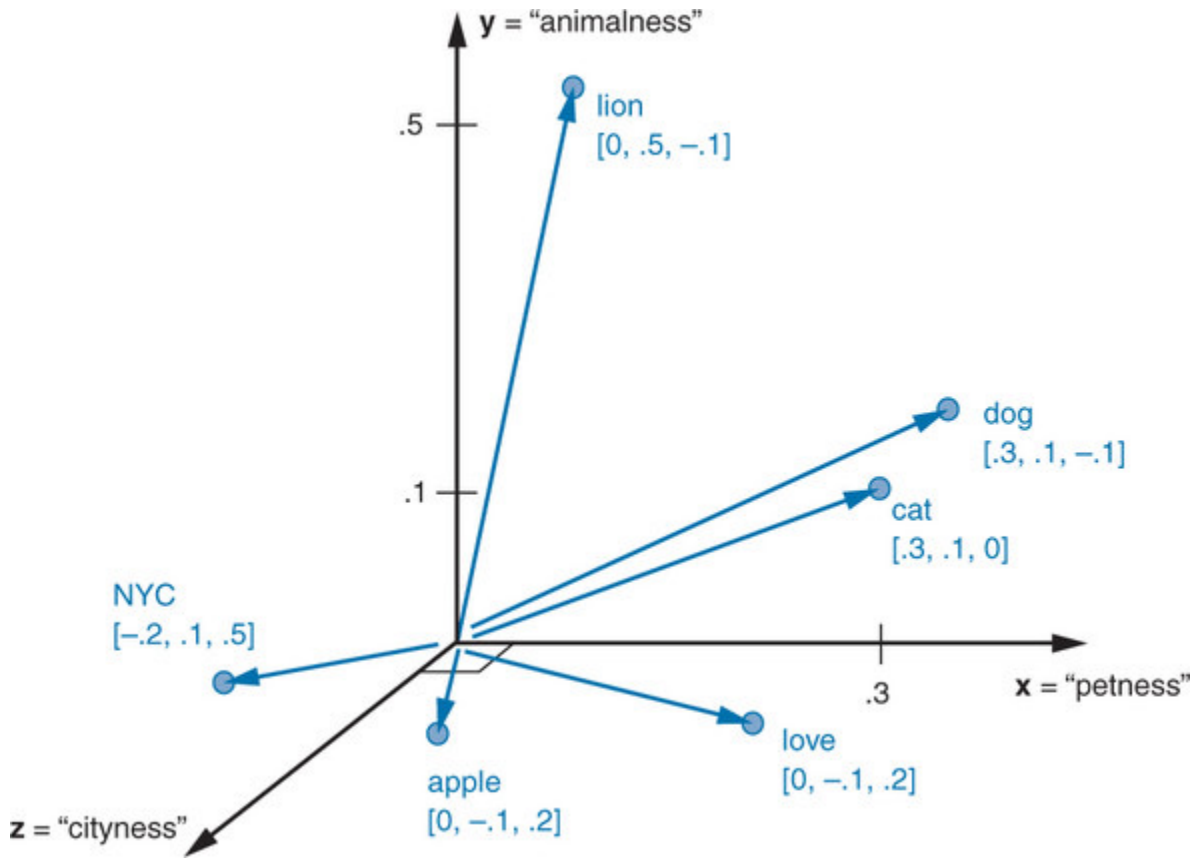
Human approach

- Consider you are doing the two tasks manually
- What’s you rule?
  - The number of topics
  - How about weights?
    - Probably you perform a semantic approach
- How about a machine?

The relationships can be flipped

- 3X1 topic vector → 6X1 TF-IDF
  - Need to have 6X3 weights
  - Human approach: no problem
  - See the code and make sense of it
- If we have the weights, possible to represent each words in terms of topics

Represent each word in terms of topics



Wonderful topic vectors

- Using three topics animal, pet, and NYC, we can represent all the words in our corpus, then.
  - In the thought experiment, you compressed six dimensions (TF-IDF normalized frequencies) into three dimensions (topics)
- This subjective, labor-intensive approach to semantic analysis relies on ***human intuition and common sense*** to break down documents into topics
  - Topics mean really something here

Machine Approach

- How could we deal with common sense and intuition?
- That is, what's the algorithm to generate topic vectors?
- Textbook borrows the idea:
  - *You shall know a word by the company it keeps*
  - Well, not necessary true here
- Let's start with Linear discriminant analysis
  - Discuss whether we could find the ***topics***

Linear discriminant analysis (LDA)

- It is not a topic modeling
- Remember what we need to do for the topic modeling
  - The number of topics
  - The corresponding weights
- An LDA classifier is a supervised algorithm, so you do need labels for your document classes
  - The document classes should be given!
    - Supervised learning
    - Let’s see the code

What are we doing?

- Still frequency matters
- Calculate the TF-IDF
- Assume each token plays a role for the particular class
- Calculate the centroid of each token by class
  - Assume each word has some certain frequencies for the particular the class
  - Consider: tfidf_docs.dot(spam_centroid - ham_centroid)
  - Explain in words
    - E.g. what's the meaning of 0 for a token?
- Try to compare it with Naïve Bayes (optional: see the code)

Latent semantic analysis

- LDA is not very interesting in terms of a topic modeling
  - We are wondering how machines understand a topic!
    - Remember human intuition and common sense?
- Let’s pause here and explore the concept of PCA
  - PCA is essentially the Latent semantic analysis

Change the order

- Many students feel overwhelmed on this subject
- I will guide you
  - the PCA code first
  - NLP application
  - And then, explanation
  - Let’s see the code

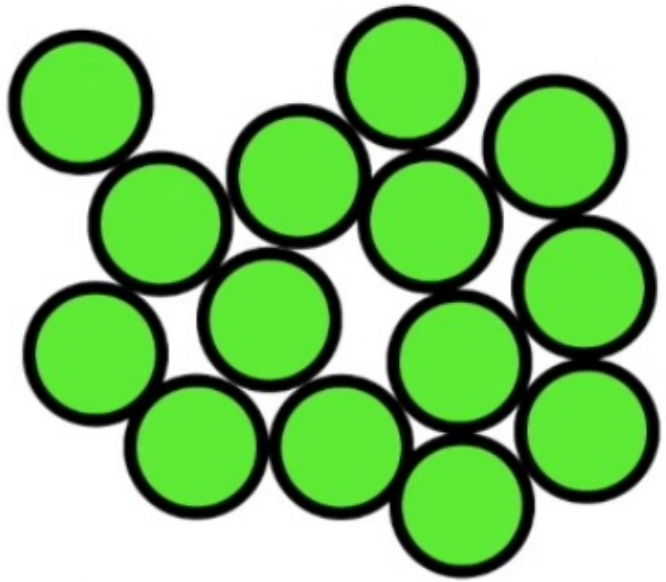
Principal Component Analysis

- Do not look at the name of the columns yet
- We have 4,837 sms spam-labeled messages
- The size of vocab is 9,232
- Apply PCA with 16 PCs
    - Explain about 10 % variance of the data

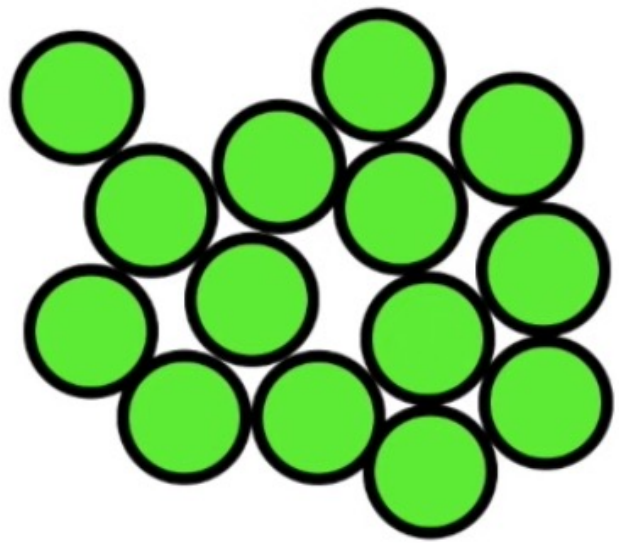
See the values

	topic0	topic1	topic2	topic3	topic4	topic5	topic6
sms0	0.201	0.003	0.037	0.011	-0.019	-0.053	0.039
sms1	0.404	-0.094	-0.078	0.051	0.100	0.047	0.023
sms2!	-0.030	-0.048	0.090	-0.067	0.091	-0.043	-0.000
sms3	0.329	-0.033	-0.035	-0.016	0.052	0.056	-0.165
sms4	0.002	0.031	0.038	0.034	-0.075	-0.092	-0.044
sms5!	-0.016	0.059	0.014	-0.006	0.122	-0.040	0.005

Now let’s find out the meaning of PCA

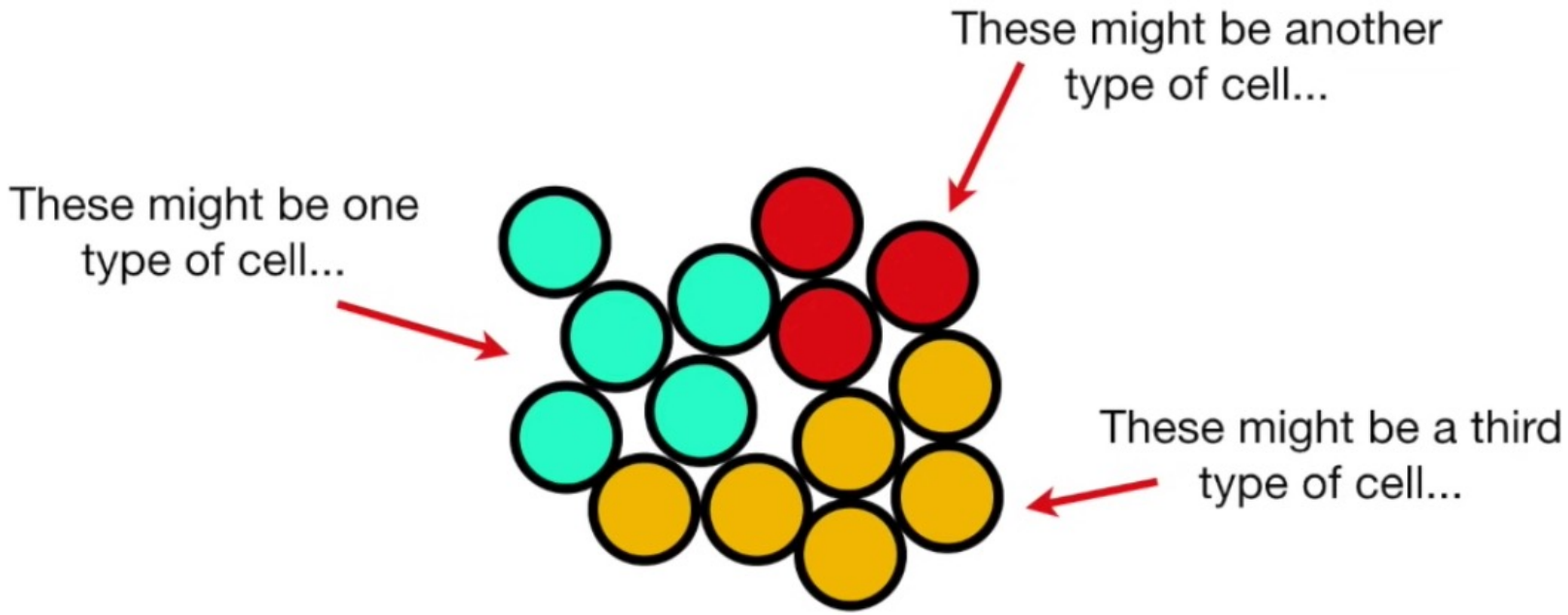


Let's say we had some normal cells...



Even though they look the same, we suspect that there are differences...

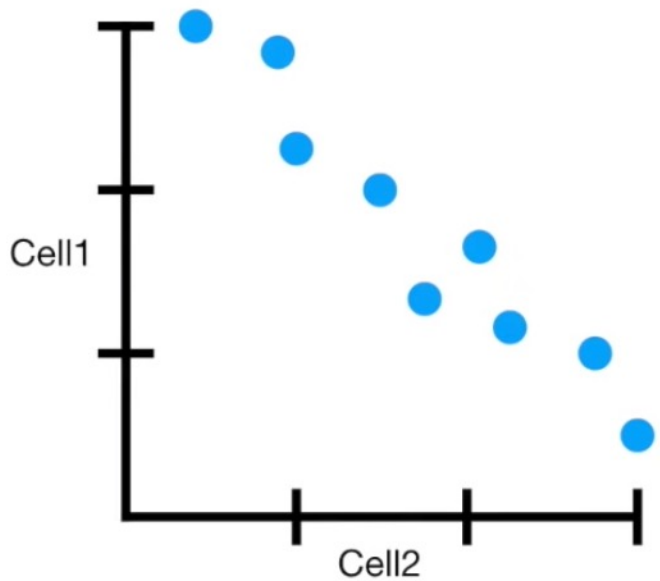
We can’t observe the differences from the outside



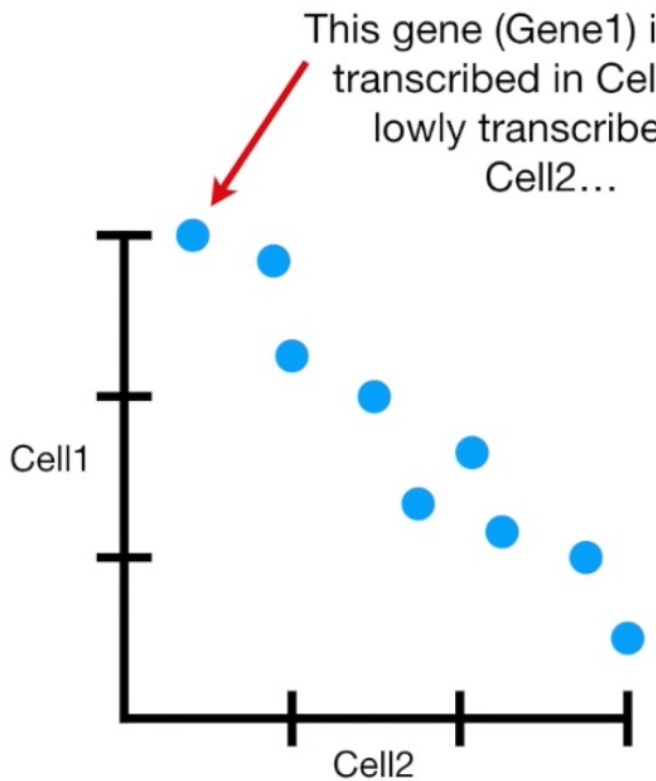
Here's the data…

	Cell1	Cell2	Cell3	Cell4	...
Gene1	3	0.25	2.8	0.1	...
Gene2	2.9	0.8	2.2	1.8	...
Gene3	2.2	1	1.5	3.2	...
Gene4	2	1.4	2	0.3	...
Gene5	1.3	1.6	1.6	0	...
Gene6	1.5	2	2.1	3	...
Gene7	1.1	2.2	1.2	2.8	...
Gene8	1	2.7	0.9	0.3	...
Gene9	0.4	3	0.6	0.1	...

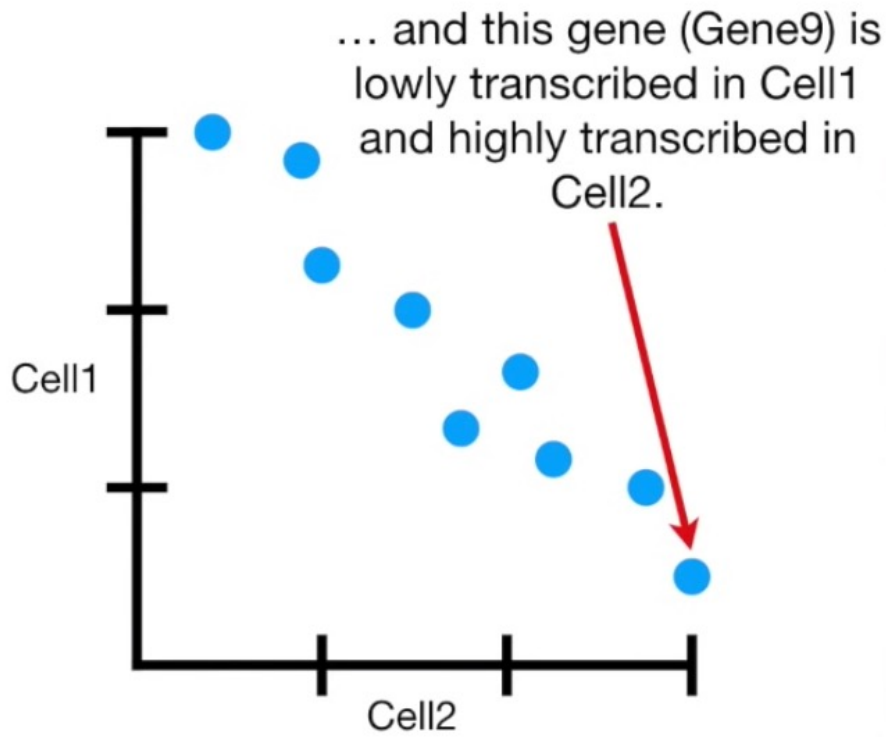
We we just have 2 cells, we can plot the measurements for each gene.



	Cell1	Cell2
Gene1	3	0.25
Gene2	2.9	0.8
Gene3	2.2	1
Gene4	2	1.4
Gene5	1.3	1.6
Gene6	1.5	2
Gene7	1.1	2.2
Gene8	1	2.7
Gene9	0.4	3

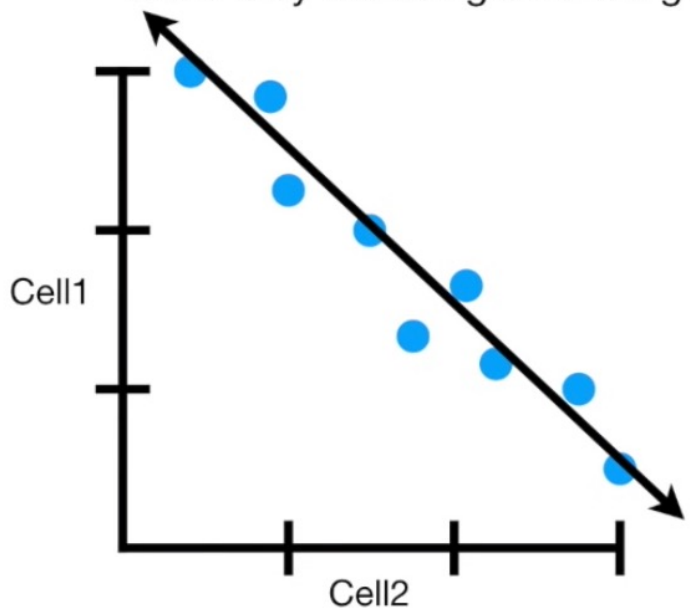


	Cell1	Cell2
Gene1	3	0.25
Gene2	2.9	0.8
Gene3	2.2	1
Gene4	2	1.4
Gene5	1.3	1.6
Gene6	1.5	2
Gene7	1.1	2.2
Gene8	1	2.7
Gene9	0.4	3



	Cell1	Cell2
Gene1	3	0.25
Gene2	2.9	0.8
Gene3	2.2	1
Gene4	2	1.4
Gene5	1.3	1.6
Gene6	1.5	2
Gene7	1.1	2.2
Gene8	1	2.7
Gene9	0.4	3

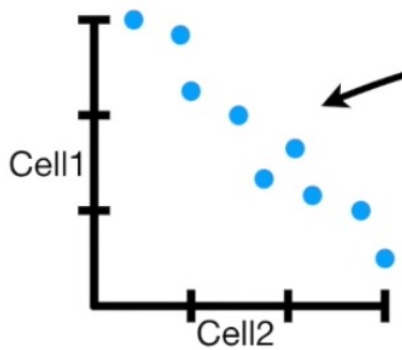
In general, Cell1 and Cell2 have an inverse correlation. This means that they are probably two different types of cells since they are using different genes.



	Cell1	Cell2
Gene1	3	0.25
Gene2	2.9	0.8
Gene3	2.2	1
Gene4	2	1.4
Gene5	1.3	1.6
Gene6	1.5	2
Gene7	1.1	2.2
Gene8	1	2.7
Gene9	0.4	3

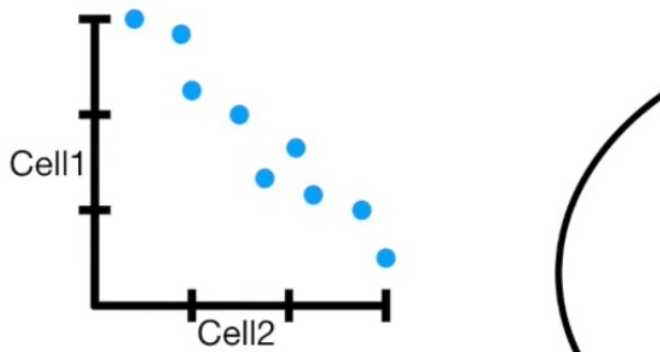
Now let's imagine there are 3 cells.

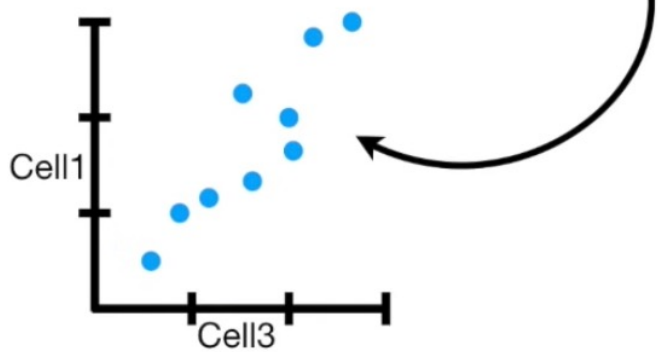
	Cell1	Cell2	Cell3
Gene1	3	0.25	2.8
Gene2	2.9	0.8	2.2
Gene3	2.2	1	1.5
Gene4	2	1.4	2
Gene5	1.3	1.6	1.6
Gene6	1.5	2	2.1
Gene7	1.1	2.2	1.2
Gene8	1	2.7	0.9
Gene9	0.4	3	0.6



We've already seen how we can plot the first 2 cells to see how closely they are related.

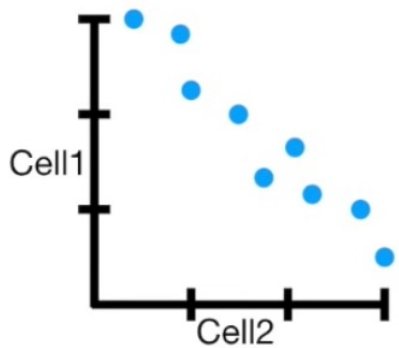
	Cell1	Cell2	Cell3
Gene1	3	0.25	2.8
Gene2	2.9	0.8	2.2
Gene3	2.2	1	1.5
Gene4	2	1.4	2
Gene5	1.3	1.6	1.6
Gene6	1.5	2	2.1
Gene7	1.1	2.2	1.2
Gene8	1	2.7	0.9
Gene9	0.4	3	0.6

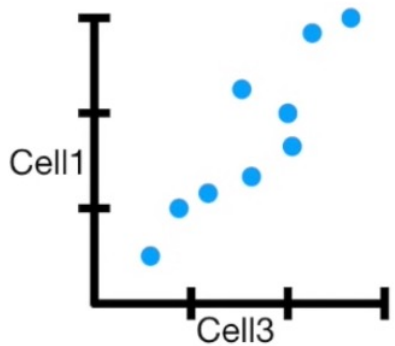




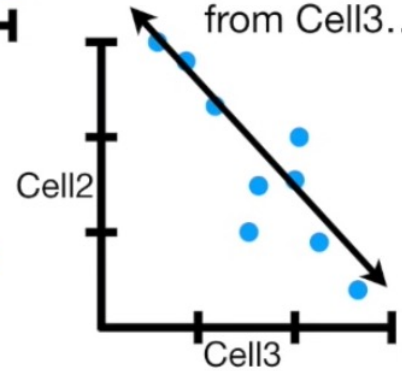
Now we can also compare
Cell1 to Cell3…

	Cell1	Cell2	Cell3
Gene1	3	0.25	2.8
Gene2	2.9	0.8	2.2
Gene3	2.2	1	1.5
Gene4	2	1.4	2
Gene5	1.3	1.6	1.6
Gene6	1.5	2	2.1
Gene7	1.1	2.2	1.2
Gene8	1	2.7	0.9
Gene9	0.4	3	0.6



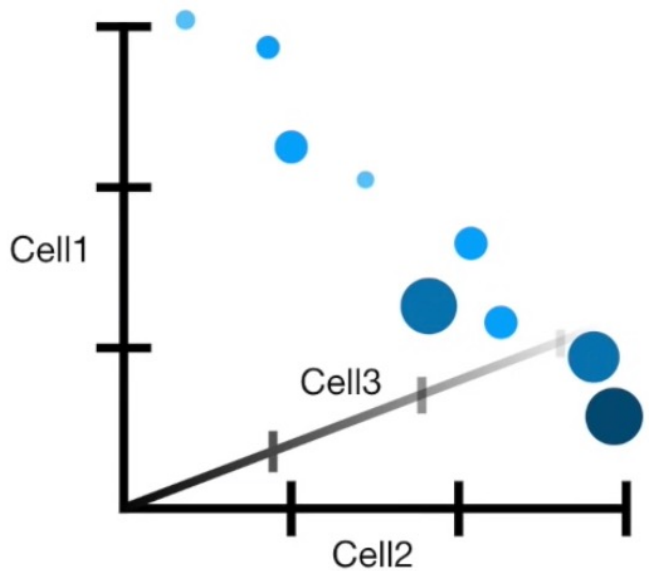


The negative correlation suggests that Cell2 is doing something different from Cell3…



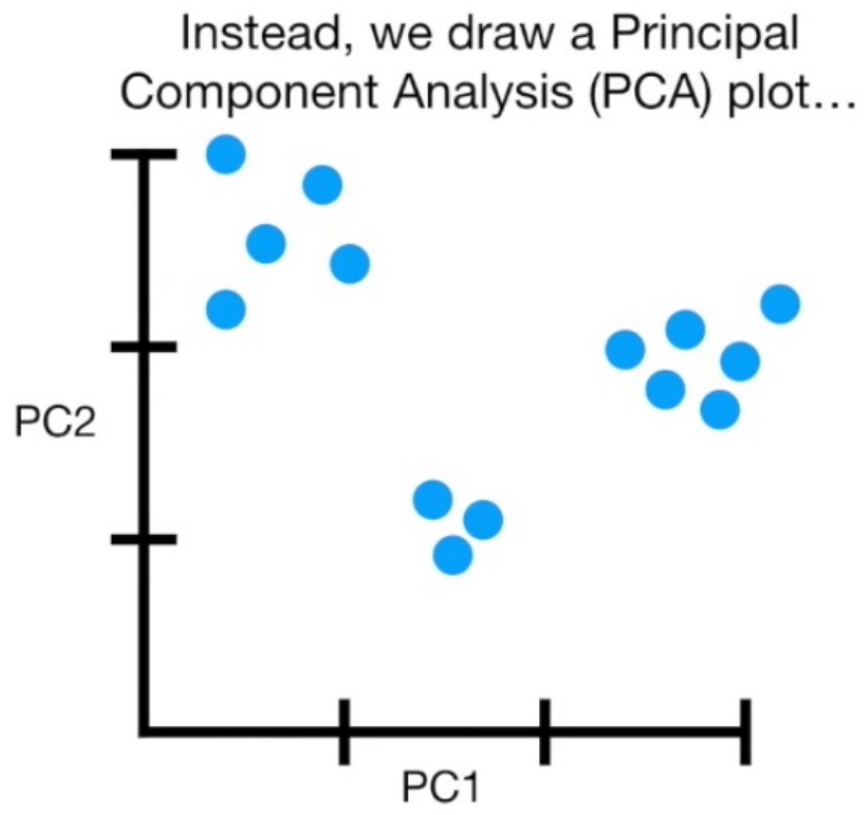
	Cell1	Cell2	Cell3
Gene1	3	0.25	2.8
Gene2	2.9	0.8	2.2
Gene3	2.2	1	1.5
Gene4	2	1.4	2
Gene5	1.3	1.6	1.6
Gene6	1.5	2	2.1
Gene7	1.1	2.2	1.2
Gene8	1	2.7	0.9
Gene9	0.4	3	0.6

Alternatively, we could try to plot all three cells at once on a 3-dimensional graph.



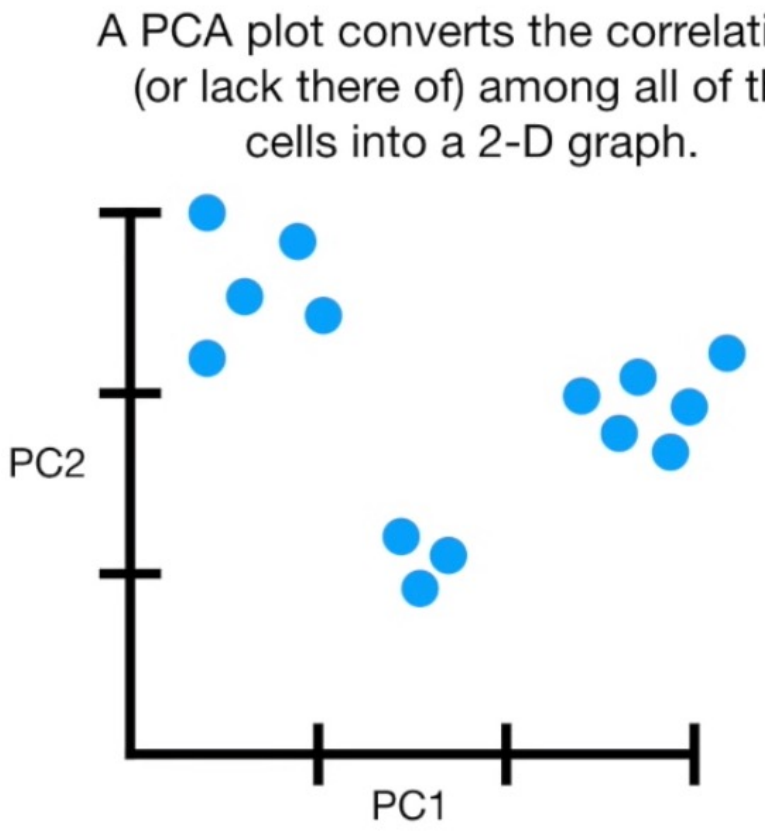
	Cell1	Cell2	Cell3
Gene1	3	0.25	2.8
Gene2	2.9	0.8	2.2
Gene3	2.2	1	1.5
Gene4	2	1.4	2
Gene5	1.3	1.6	1.6
Gene6	1.5	2	2.1
Gene7	1.1	2.2	1.2
Gene8	1	2.7	0.9
Gene9	0.4	3	0.6

But what do we do when we have 4 or more cells?

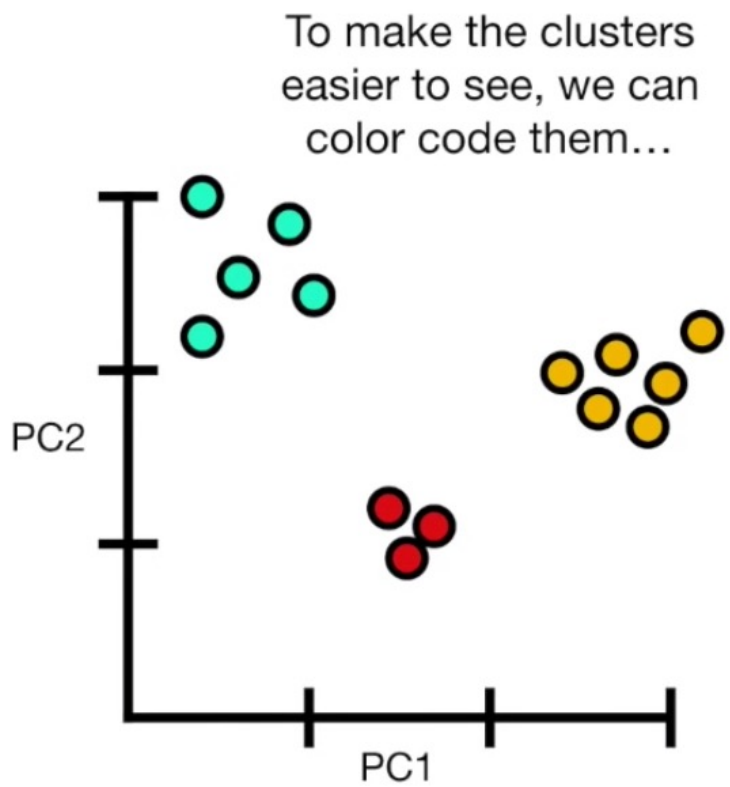


	Cell1	Cell2	Cell3	Cell4	...
Gene1	3	0.25	2.8	0.1	...
Gene2	2.9	0.8	2.2	1.8	...
Gene3	2.2	1	1.5	3.2	...
Gene4	2	1.4	2	0.3	...
Gene5	1.3	1.6	1.6	0	...
Gene6	1.5	2	2.1	3	...
Gene7	1.1	2.2	1.2	2.8	...
Gene8	1	2.7	0.9	0.3	...
Gene9	0.4	3	0.6	0.1	...

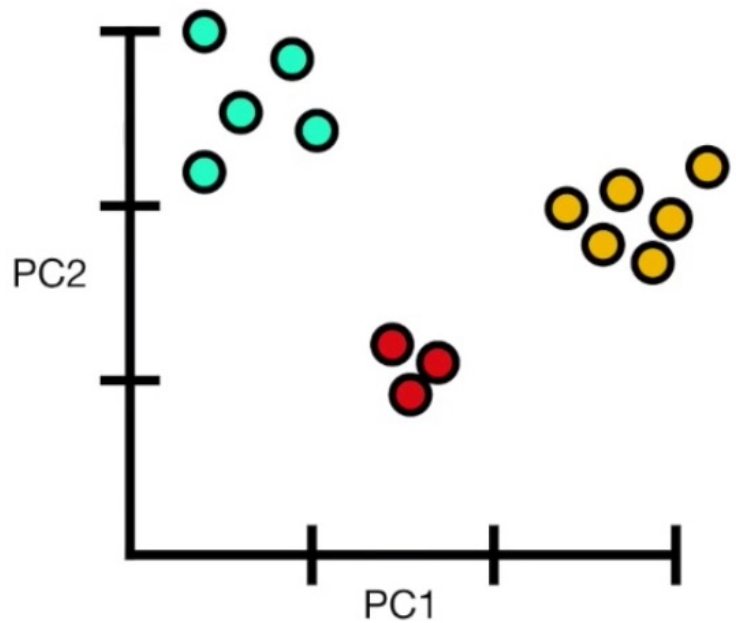
Cells that are highly correlated cluster together



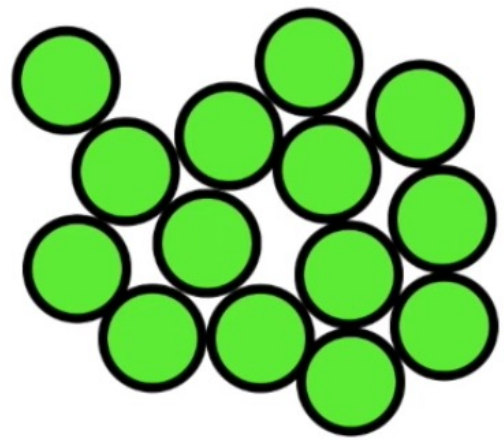
	Cell1	Cell2	Cell3	Cell4	...
Gene1	3	0.25	2.8	0.1	...
Gene2	2.9	0.8	2.2	1.8	...
Gene3	2.2	1	1.5	3.2	...
Gene4	2	1.4	2	0.3	...
Gene5	1.3	1.6	1.6	0	...
Gene6	1.5	2	2.1	3	...
Gene7	1.1	2.2	1.2	2.8	...
Gene8	1	2.7	0.9	0.3	...
Gene9	0.4	3	0.6	0.1	...

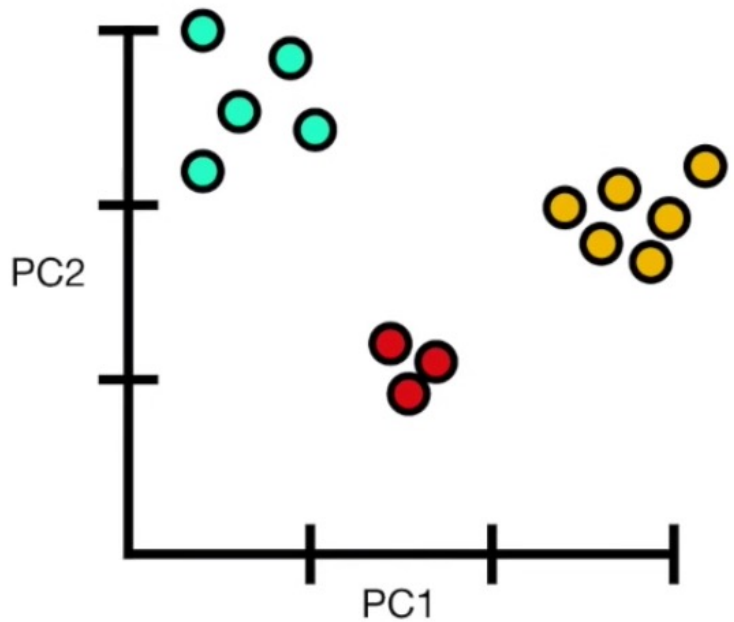


	Cell1	Cell2	Cell3	Cell4	...
Gene1	3	0.25	2.8	0.1	...
Gene2	2.9	0.8	2.2	1.8	...
Gene3	2.2	1	1.5	3.2	...
Gene4	2	1.4	2	0.3	...
Gene5	1.3	1.6	1.6	0	...
Gene6	1.5	2	2.1	3	...
Gene7	1.1	2.2	1.2	2.8	...
Gene8	1	2.7	0.9	0.3	...
Gene9	0.4	3	0.6	0.1	...

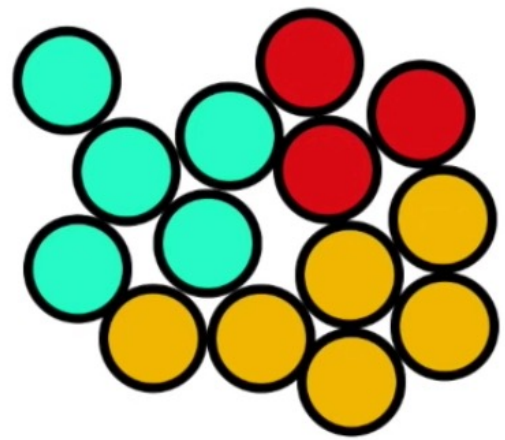


Once we've identified the clusters in the PCA plot, we can go back to the original cells…





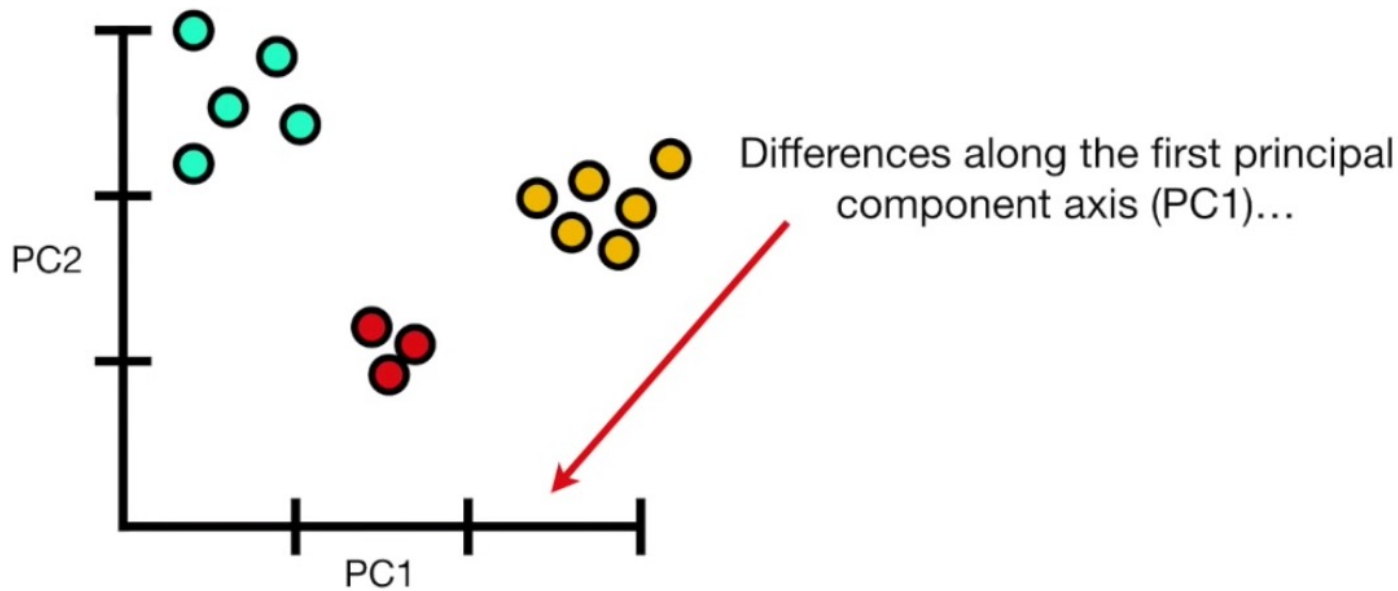
Once we've identified the clusters in the PCA plot, we can go back to the original cells…



…and see that they represent 3 different types of cells doing 3 different things with their genes!!!!

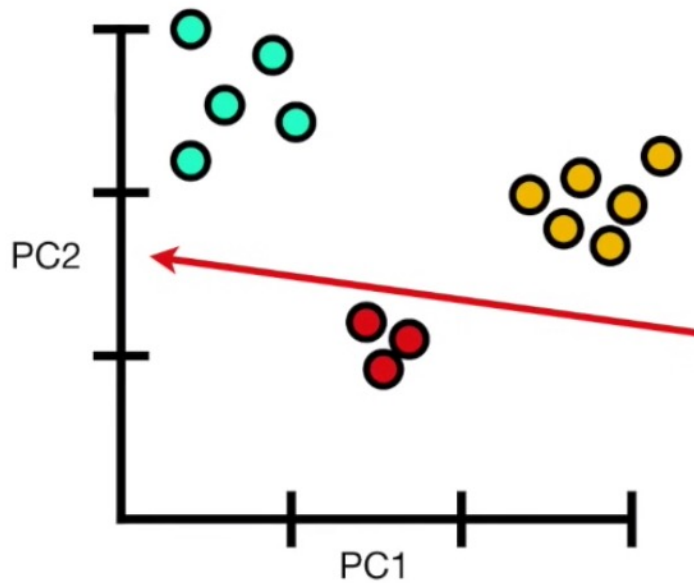
Here's one last main idea about how to interpret PCA plots:

The axes are ranked in order of importance.



Here's one last main idea about how to interpret PCA plots:

The axes are ranked in order of importance.



Differences along the first principal component axis (PC1)…

…are more important than differences along the second principal component axis (PC2).

PCA and Topic modeling

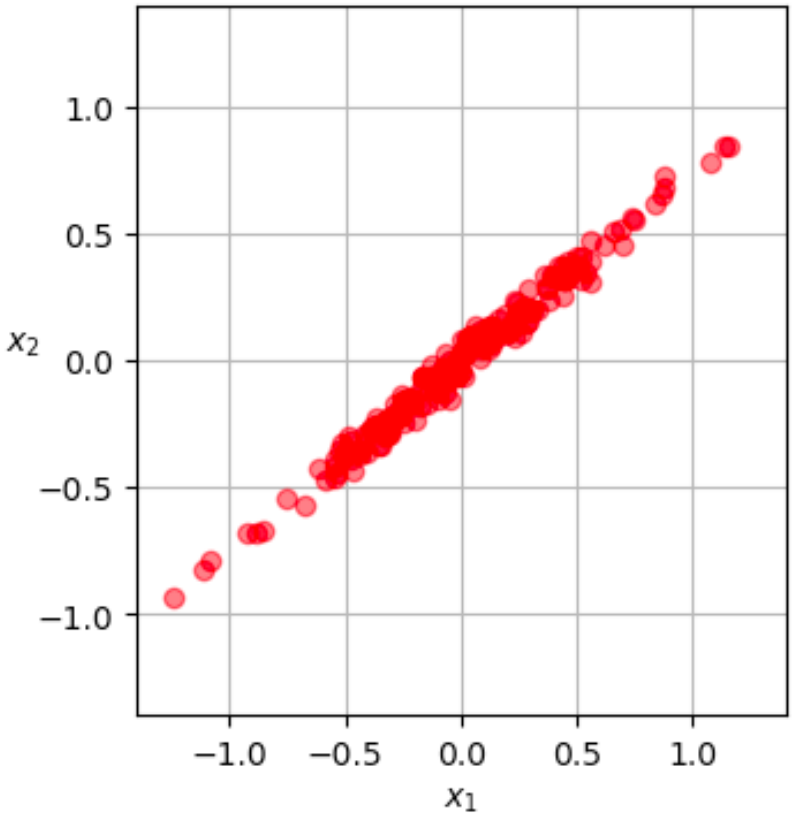
- Selecting principal components is not equivalent to selecting features
  - while selecting principal components may lead to a reduction in the dimensionality of the data, it may not necessarily identify the most relevant subset of features
- Selecting principal components involves identifying the most important patterns or directions of variation in the data
  - These components are linear combinations of the original features
- We will consider whether the linear combinations of the words would imply the ***topics***

PCA in math

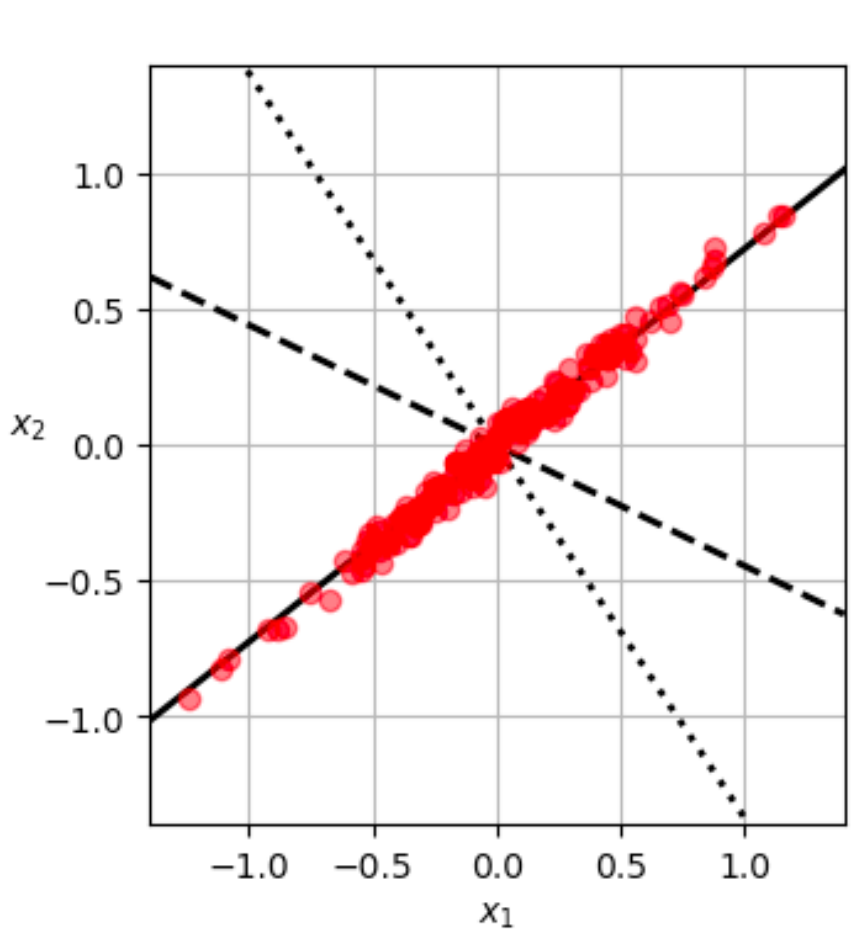
- PCA algorithm first identifies the hyperplane that lies closet to the data and then it projects the data onto it
  - As a result, the corresponding data preserves the maximum variance of the original data

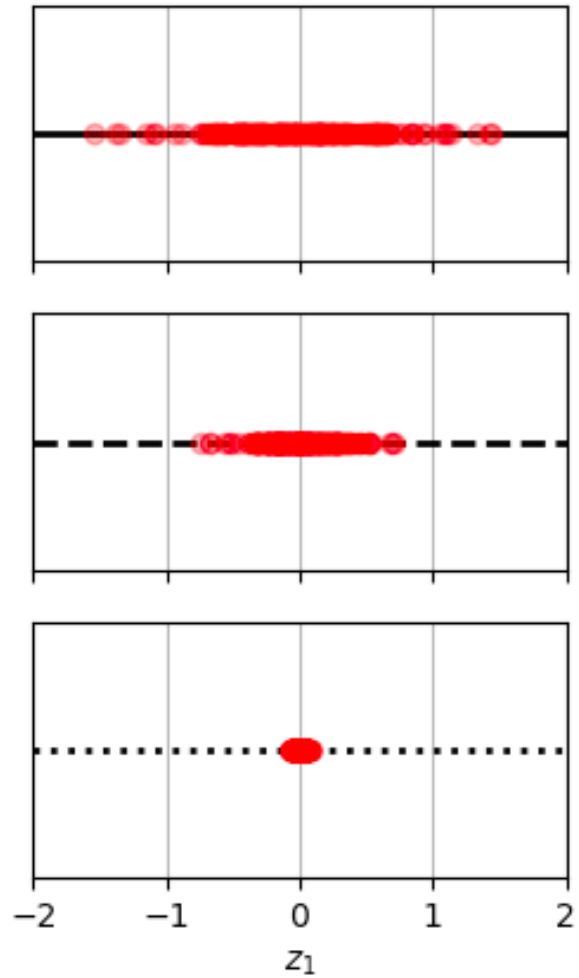
Figure out PCs

- We have the following data
- Need to find the closet hyperplane to data
  - It is just a line
- The projection on the the line



Projection on the 3 hyperplanes





Principal Components

- PCA identifies the solid line earlier
  - Data has two features: x1 and x2
  - PCA identifies the axis accounting for the largest amount of variance in data
  - It also find the dotted line, orthogonal to the first one
    - Accounts for the largest amount of the remaining variance
- It uses a standard matrix factorization technique called singular value decomposition (SVD)

SVD

- $X = U\Sigma V^T$ where V contains the unit vectors that define all the principal components

$$\mathbf{V} = \begin{pmatrix} | & | & & | \\ \mathbf{c}_1 & \mathbf{c}_2 & \cdots & \mathbf{c}_n \\ | & | & & | \end{pmatrix}$$

- After finding the PCs, need to project down to the reduced dimensions
- See the code