# Information Retrieval

- Word Similarity
- WordNet
- Word Vectors

Development:
Moshe Friedman

Credits:
Yoav Goldberg, Ido Dagan, Reut Tsarfaty , Moshe Koppel, Wei Song, David Bamman, Ed Grefenstette, Chris Manning, Tsvi Kuflik, Hinrich Schütze, Christina Lioma and more

# Information Retrieval - administration

Moshe Friedman

Email: moshefr.teach@gmail.com

Reception time: before/after lesson/zoom with coordination

# Recall (Lecture 2): Binary term-document incidence matrix

| | Antony and Cleopatra | Julius Caesar | The Tempest | Hamlet | Othello | Macbeth |
|---|---|---|---|---|---|---|
| Antony | 1 | 1 | 0 | 0 | 0 | 1 |
| Brutus | 1 | 1 | 0 | 1 | 0 | 0 |
| Caesar | 1 | 1 | 0 | 1 | 1 | 1 |
| Calpurnia | 0 | 1 | 0 | 0 | 0 | 0 |
| Cleopatra | 1 | 0 | 0 | 0 | 0 | 0 |
| mercy | 1 | 0 | 1 | 1 | 1 | 1 |
| worser | 1 | 0 | 1 | 1 | 1 | 0 |

Each document is represented by a binary vector $\in \{0,1\}^{|V|}$

# Term-document count matrices

- Consider the number of occurrences of a term in a document:
  - Each document is a count vector in $\mathbb{N}^{|V|}$: a column below

| | Antony and Cleopatra | Julius Caesar | The Tempest | Hamlet | Othello | Macbeth |
|---|---|---|---|---|---|---|
| Antony | 157 | 73 | 0 | 0 | 0 | 0 |
| Brutus | 4 | 157 | 0 | 1 | 0 | 0 |
| Caesar | 232 | 227 | 0 | 2 | 1 | 1 |
| Calpurnia | 0 | 10 | 0 | 0 | 0 | 0 |
| Cleopatra | 57 | 0 | 0 | 0 | 0 | 0 |
| mercy | 2 | 0 | 3 | 5 | 5 | 1 |
| worser | 2 | 0 | 1 | 1 | 1 | 0 |

# idf example, suppose $N$ = 1 million

| term | $df_t$ | $idf_t$ |
|---|---|---|
| calpurnia | 1 | 6 |
| animal | 100 | 4 |
| sunday | 1,000 | 3 |
| fly | 10,000 | 2 |
| under | 100,000 | 1 |
| the | 1,000,000 | 0 |

$$\text{idf}_t = \log_{10}(N/\text{df}_t)$$

There is one idf value for each term $t$ in a collection.

# Binary → count → weight matrix

| | Antony and Cleopatra | Julius Caesar | The Tempest | Hamlet | Othello | Macbeth |
|---|---|---|---|---|---|---|
| Antony | 5.25 | 3.18 | 0 | 0 | 0 | 0.35 |
| Brutus | 1.21 | 6.1 | 0 | 1 | 0 | 0 |
| Caesar | 8.59 | 2.54 | 0 | 1.51 | 0.25 | 0 |
| Calpurnia | 0 | 1.54 | 0 | 0 | 0 | 0 |
| Cleopatra | 2.85 | 0 | 0 | 0 | 0 | 0 |
| mercy | 1.51 | 0 | 1.9 | 0.12 | 5.25 | 0.88 |
| worser | 1.37 | 0 | 0.11 | 4.15 | 0.25 | 1.95 |

Each document is now represented by a real-valued vector of tf-idf weights $\in \mathbb{R}^{|V|}$

# Summary - Ranking

$$match\_score(doc, query) = \sum_{term \in query \cap doc} score(term)$$

$match\_rank(doc, query) = index\ in\ reverse(sorted([s\ for\ d\ in\ corpus\ for\ s\ in\ match\_score(d, q)]))$

**score term**:

boolean_score$(doc, term) = 1\ if\ term\ in\ doc\ else\ 0$

tf_score$(doc, term) = count(term\ in\ sore)\ if\ term\ in\ doc\ else\ 0$

log_tf_score$(doc, term) = \log\big(1 + count(term\ in\ sore)\big)\ if\ term\ in\ doc\ else\ 0$

tf−idf_score$(doc, term) = tf\ *_{score(doc,term)} \cdot\ idf(doc, term)\ if\ term\ in\ doc\ else\ 0$

idf$(term) = \log\left(\frac{N}{df(term)}\right)$ $\quad df(term) = count(term\ in\ doc\ for\ doc\ in\ corpus)$

* Could be either $tf_{score(doc,term)}\ or$ log_tf_score$(doc, term)$

# But raw frequency is a bad representation

- The co-occurrence matrices we have seen represent each cell by word frequencies.
- Frequency is clearly useful; if *sugar* appears a lot near *apricot*, that's useful information.
- But overly frequent words like *the*, *it,* or *they* are not very informative about the context
- It's a paradox! How can we balance these two conflicting constraints?

# Problems with Discrete Symbols

All vectors are **orthogonal**

No notion of similarity: words **hotel**, **motel** have nothing in common

Search Engines try to address the issue using WordNet synonyms

# Problems with Discrete Symbols

All vectors are **orthogonal**

No notion of similarity: words **hotel**, **motel** have nothing in common

Search Engines try to address the issue using WordNet synonyms

**Still doesn't solve some problems**

- E.g., Base form needed, small subset (Hebrew only around 5,500 synsets)

Alternative solution:

**Encode similarity in the vector themselves**

# Idea for word meaning: Words can be vectors too!!!

| | As You Like It | Twelfth Night | Julius Caesar | Henry V |
|---|---|---|---|---|
| **battle** | 1 | 0 | 7 | 13 |
| **good** | 114 | 80 | 62 | 89 |
| **fool** | 36 | 58 | 1 | 4 |
| **wit** | 20 | 15 | 2 | 3 |

*battle* is "the kind of word that occurs in Julius Caesar and Henry V"

*fool* is "the kind of word that occurs  in comedies, especially Twelfth Night"

# What is the meaning of a word?

- Most words have many different senses
  - dog = animal or sausage?
  - lie = to be in a horizontal position or a false statement made with deliberate intent
- What are the relations of different words in terms of meaning?
  - Specific relations between senses
    - Animal is more general than dog
  - Semantic fields
    - Money is related to bank

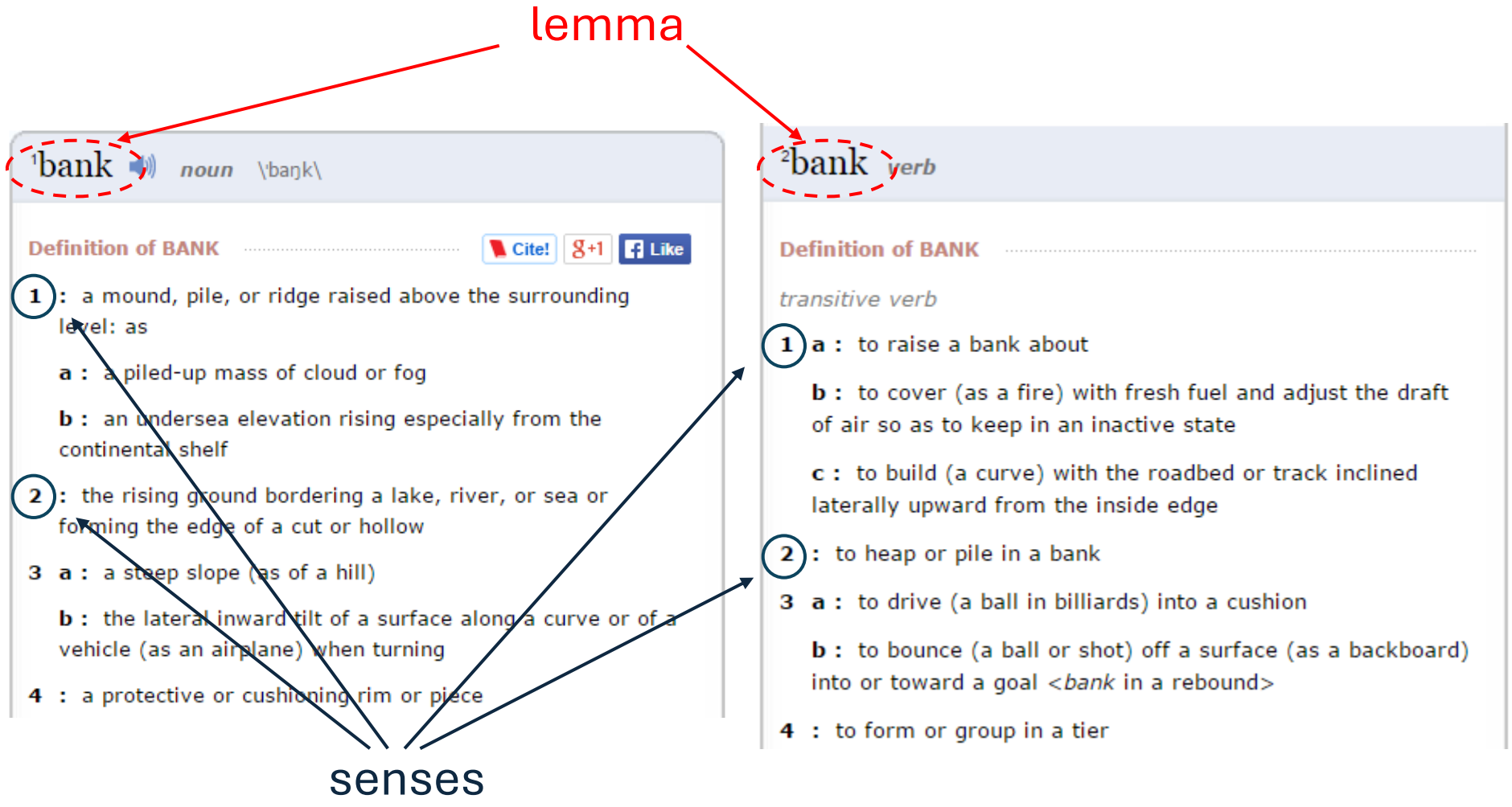


*"a set of words grouped, referring to a specific subject … not necessarily synonymous, but are all used to talk about the same general phenomenon"* - wiki

# Word senses

- What does ‘bank’ mean?
  - A financial institution
    - E.g., “US bank has raised interest rates.”
  - A particular branch of a financial institution
    - E.g., “The bank on Main Street closes at 5pm.”
  - The sloping side of any hollow in the ground, especially when bordering a river
    - E.g., “In 1927, the bank of the Mississippi flooded.”
  - A ‘repository’
    - E.g., “I donate blood to a blood bank.”

# Lexicon entries



# Some terminologies

- **Word forms**: runs, ran, running; good, better, best
  - Any, possibly inflected, form of a word
- **Lemma** (citation/dictionary form): run; good
  - A basic word form (e.g. infinitive or singular nominative noun) that is used to represent all forms of the same word
- **Lexeme**: RUN(V), GOOD(A), $BANK^{1}$(N), $BANK^{2}$(N)
  - An abstract representation of a word (and all its forms), with a part-of-speech and a set of related word senses
  - Often just written (or referred to) as the lemma, perhaps in a different FONT
- **Lexicon**
  - A (finite) list of lexemes

# Make sense of word senses

- Polysemy
  - A lexeme is polysemous if it has different related senses



bank = financial institution or a building

# Make sense of word senses

- Homonyms
  - Two lexemes are homonyms if their senses are unrelated, but they happen to have the same spelling and pronunciation


bank = financial institution or river bank

# Relations between senses

- Symmetric relations
  - Synonyms: couch/sofa
    - Two lemmas with the same sense
  - Antonyms: cold/hot, rise/fall, in/out
    - Two lemmas with the opposite sense
- Hierarchical relations:
  - Hypernyms and hyponyms: pet/dog
    - The hyponym (dog) is more specific than the hypernym (pet)
  - Holonyms and meronyms: car/wheel
    - The meronym (wheel) is a part of the holonym (car)

# WordNet

*George Miller, Cognitive Science Laboratory of Princeton University, 1985*

- A very large lexical database of English:
  - 117K nouns, 11K verbs, 22K adjectives, 4.5K adverbs
- Word senses grouped into synonym sets ("synsets") linked into a conceptual-semantic hierarchy
  - 82K noun synsets, 13K verb synsets, 18K adjectives synsets, 3.6K adverb synsets
  - Avg. # of senses: 1.23/noun, 2.16/verb, 1.41/adj, 1.24/adverb
- Conceptual-semantic relations
  - hypernym/hyponym

# A WordNet example

- http://wordnet.princeton.edu/

# Hierarchical synset relations: nouns

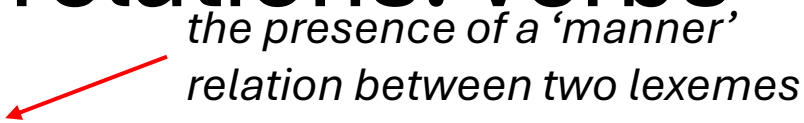
- **Hypernym/hyponym** (between concepts)
  - The more general ‘meal’ is a hypernym of the more specific ‘breakfast’
- **Instance hypernym/hyponym** (between concepts and instances)
  - Austen is an instance hyponym of author
- **Member holonym/meronym** (groups and members)
  - professor is a member meronym of (a university’s) faculty
- **Part holonym/meronym** (wholes and parts)
  - wheel is a part meronym of (is a part of) car.
- **Substance meronym/holonym** (substances and components)
  - flour is a substance meronym of (is made of) bread

*Jane Austen, 1775–1817, English novelist*

# WordNet hypernyms & hyponyms

- S: (n) **bank** (sloping land (especially the slope beside a body of water))
  - *direct hyponym* / *full hyponym*
    - S: (n) riverbank, riverside (the bank of a river)
    - S: (n) waterside (land bordering a body of water)
  - ***direct hypernym*** / *inherited hypernym* / *sister term*
    - S: (n) slope, incline, side (an elevated geological formation)
  - *derivationally related form*
- S: (n) depository financial institution, **bank**, banking concern, banking company (a financial institution that accepts deposits and channels the money into lending activities)
  - *direct hyponym* / *full hyponym*
    - S: (n) credit union (a cooperative depository financial institution whose members can obtain loans from their combined savings)
      - *direct hypernym* / *inherited hypernym* / *sister term*
        - S: (n) depository financial institution, **bank**, banking concern, banking company (a financial institution that accepts deposits and channels the money into lending activities)
    - S: (n) Federal Reserve Bank, reserve bank (one of 12 regional banks that monitor and act as depositories for banks in their region)
    - S: (n) agent bank (a bank that acts as an agent for a foreign bank)
    - S: (n) commercial bank, full service bank (a financial institution that accepts demand deposits and makes loans and provides other services for the public)

# Hierarchical synset relations: verbs




- Hypernym/troponym (between events)
  - travel/fly, walk/stroll
  - Flying is a troponym of traveling: it denotes a specific manner of traveling
- Entailment (between events):
  - snore/sleep
    - Snoring entails (presupposes) sleeping

# WordNet similarity

- Path based similarity measure between words
  - Shortest path between two concepts (Leacock & Chodorow 1998)
    - sim = 1/|shortest path|
  - Path length to the root node from the least common subsumer (LCS) of the two concepts (Wu & Palmer 1994)
    - sim = $2*depth(LCS)/(depth(w_1)+depth(w_2))$
- http://wn-similarity.sourceforge.net/

*the most specific concept which is an ancestor of both A and B.*

# WordNet::Similarity

| Measure | Word 1 | Word 2 | Score | Trace |
|---|---|---|---|---|
| path | apple#n#1 | pizza#n#1 | 0.0909 | HyperTree: *Root*#n#1 entity#n#1 physical_entity#n#1 matter#n#3 solid#n#1 food#n#2 produce#n#1 edible_fruit#n#1 apple#n#1<br>HyperTree: *Root*#n#1 entity#n#1 physical_entity#n#1 object#n#1 whole#n#2 natural_object#n#1 plant_part#n#1 plant_organ#n#1 reproductive_structure#n#1 fruit#n#1 edible_fruit#n#1 apple#n#1<br>HyperTree: *Root*#n#1 entity#n#1 physical_entity#n#1 object#n#1 whole#n#2 natural_object#n#1 plant_part#n#1 plant_organ#n#1 reproductive_structure#n#1 fruit#n#1 pome#n#1 apple#n#1<br>HyperTree: *Root*#n#1 entity#n#1 physical_entity#n#1 matter#n#3 substance#n#7 food#n#1 nutriment#n#1 dish#n#2 pizza#n#1<br>Shortest path: apple#n#1 edible_fruit#n#1 produce#n#1 food#n#2 solid#n#1 matter#n#3 substance#n#7 food#n#1 nutriment#n#1 dish#n#2 pizza#n#1<br>Path length = 11 |
| path | apple#n#2 | pizza#n#1 | 0.0526 | HyperTree: *Root*#n#1 entity#n#1 physical_entity#n#1 object#n#1 whole#n#2 living_thing#n#1 organism#n#1 plant#n#2 vascular_plant#n#1 woody_plant#n#1 tree#n#1 angiospermous_tree#n#1 fruit_tree#n#1 apple_tree#n#1 apple#n#2<br>HyperTree: *Root*#n#1 entity#n#1 physical_entity#n#1 matter#n#3 substance#n#7 food#n#1 nutriment#n#1 dish#n#2 pizza#n#1<br>Shortest path: apple#n#2 apple_tree#n#1 fruit_tree#n#1 angiospermous_tree#n#1 tree#n#1 woody_plant#n#1 vascular_plant#n#1 plant#n#2 organism#n#1 living_thing#n#1 whole#n#2 object#n#1 physical_entity#n#1 matter#n#3 substance#n#7 food#n#1 nutriment#n#1 dish#n#2 pizza#n#1<br>Path length = 19 |

# WordNet::Similarity

| Measure | Word 1 | Word 2 | Score | Trace |
|---|---|---|---|---|
| wup | apple#n#1 | pizza#n#1 | 0.4444 | HyperTree: *Root*#n#1 entity#n#1 physical_entity#n#1 matter#n#3 solid#n#1 food#n#2 produce#n#1 edible_fruit#n#1 apple#n#1<br>HyperTree: *Root*#n#1 entity#n#1 physical_entity#n#1 object#n#1 whole#n#2 natural_object#n#1 plant_part#n#1 plant_organ#n#1 reproductive_structure#n#1 fruit#n#1 edible_fruit#n#1 apple#n#1<br>HyperTree: *Root*#n#1 entity#n#1 physical_entity#n#1 object#n#1 whole#n#2 natural_object#n#1 plant_part#n#1 plant_organ#n#1 reproductive_structure#n#1 fruit#n#1 pome#n#1 apple#n#1<br>HyperTree: *Root*#n#1 entity#n#1 physical_entity#n#1 matter#n#3 substance#n#7 food#n#1 nutriment#n#1 dish#n#2 pizza#n#1<br>Lowest Common Subsumers: matter#n#3 (Depth=4)<br>Depth(apple#n#1) = 9<br>Depth(pizza#n#1) = 9 |
| wup | apple#n#2 | pizza#n#1 | 0.25 | HyperTree: *Root*#n#1 entity#n#1 physical_entity#n#1 object#n#1 whole#n#2 living_thing#n#1 organism#n#1 plant#n#2 vascular_plant#n#1 woody_plant#n#1 tree#n#1 angiospermous_tree#n#1 fruit_tree#n#1 apple_tree#n#1 apple#n#2<br>HyperTree: *Root*#n#1 entity#n#1 physical_entity#n#1 matter#n#3 substance#n#7 food#n#1 nutriment#n#1 dish#n#2 pizza#n#1<br>Lowest Common Subsumers: physical_entity#n#1 (Depth=3)<br>Depth(apple#n#2) = 15<br>Depth(pizza#n#1) = 9 |

# Taxonomies/thesaurus level

Thesaurus has a main function to connect different surface word forms with the same meaning into one sense, called synonyms. For instance:

  - musician, instrumentalist, player
  - person, individual, someone
  - life form, organism, being
- The most commonly used general thesaurus is WordNet which exists in many other languages (e.g. EuroWordNet)
  - http://www.illc.uva.nl/EuroWordNet/

# WordNet relations

- Each WordNet entry is connected with other entries in the graph through relations
- Relations in the database of nouns:

| Relation | Definition | Example |
|---|---|---|
| Hypernym | From lower to higher concepts | breakfast -> meal |
| Hyponym | From concepts to subordinates | meal -> lunch |
| Has-Member | From groups to their members | faculty -> professor |
| Member-Of | From members to their groups | copilot -> crew |
| Has-Part | From wholes to parts | table -> leg |
| Part-Of | From parts to wholes | course -> meal |
| Antonym | Opposites | leader -> follower |

# Problems with Discrete Symbols

All vectors are **orthogonal**

No notion of similarity: words **hotel**, **motel** have nothing in common

Search Engines try to address the issue using WordNet synonyms

# Problems with Discrete Symbols

All vectors are **orthogonal**

No notion of similarity: words **hotel**, **motel** have nothing in common

Search Engines try to address the issue using WordNet synonyms

**Still doesn't solve some problems**

- E.g., Base form needed, small subset (Hebrew only around 5,500 synsets)

Alternative solution:

**Encode similarity in the vector themselves**

# Relation: **Similarity**

Words with similar meanings.  Not synonyms, but sharing some element of meaning

`car, bicycle`

`cow, horse`

# Ask humans how similar 2 words are

| word1 | word2 | similarity |
|---|---|---|
| vanish | disappear | 9.8 |
| behave | obey | 7.3 |
| belief | impression | 5.95 |
| muscle | bone | 3.65 |
| modest | flexible | 0.98 |
| hole | agreement | 0.3 |

SimLex-999 dataset (Hill et al., 2015)

# But what about co-occurrences in large documents? A better solution

## Distributional Hypothesis

- **A word's meaning is given by the words that frequently appear close-by**

  *"You shall know a word by the company it keeps"* (J. R. Firth 1957)
  Old idea but not quite successful until the rise of modern statistical NLP

- When a word $w$ appears in a text, its **context** is the set of words that appear nearby (within a fixed-size window), aka **collocations**.
- Use the many contexts of $w$ to build up a representation of $w$

| | | |
|---|---|---|
| ...government debt problems turning into | **banking** | crises as happened in 2009... |
| ...saying that Europe needs unified | **banking** | regulation to replace the hodgepodge... |
| ..India has just given its | **banking** | system a shot in the arm... |

# Idea for word meaning: Words can be vectors too!!!

| | As You Like It | Twelfth Night | Julius Caesar | Henry V |
|---|---|---|---|---|
| **battle** | 1 | 0 | 7 | 13 |
| **good** | 114 | 80 | 62 | 89 |
| **fool** | 36 | 58 | 1 | 4 |
| **wit** | 20 | 15 | 2 | 3 |

*battle* is "the kind of word that occurs in Julius Caesar and Henry V"

*fool* is "the kind of word that occurs  in comedies, especially Twelfth Night"

# Co-occurrence Matrix

*One row per word with counts of co-occurrences with any other word*

| | | Context words | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | … | cook | eat | … | changed | broke | … |
| Words | cake | … | 10 | 20 | … | 0 | 0 | … |
| | steak | … | 12 | 22 | … | 0 | 0 | … |
| | engine | … | 0 | 0 | … | 3 | 10 | … |
| | tire | … | 0 | 0 | … | 10 | 1 | … |
| | … | … | … | … | … | … | … | … |

# Pointwise Mutual Information

- **Pointwise mutual information**:
  Do events x and y co-occur more than if they were independent?

$$\text{PMI}(X,Y) = \log_2 \frac{P(x,y)}{P(x)P(y)}$$

**PMI between two words**: (Church & Hanks 1989)
Do words x and y co-occur more than if they were independent?

$$\text{PMI}(word_1, word_2) = \log_2 \frac{P(word_1, word_2)}{P(word_1)P(word_2)}$$

# Positive Pointwise Mutual Information

- PMI ranges from $-\infty$ to $+\infty$
- But the negative values are problematic
  - Things are co-occurring **less than** we expect by chance
  - Unreliable without enormous corpora
    - Imagine w1 and w2 whose probability is each $10^{-6}$
    - Hard to be sure p(w1,w2) is significantly different than $10^{-12}$
  - Plus it's not clear people are good at "unrelatedness"
- So we just replace negative PMI values by 0
- Positive PMI (**PPMI**) between word1 and word2:

$$\text{PPMI}(word_1, word_2) = \max\left(\log_2 \frac{P(word_1, word_2)}{P(word_1)P(word_2)}, 0\right)$$

# Computing PPMI on a term-context matrix

- Matrix $F$ with $W$ rows (words) and $C$ columns (contexts)
- $f_{ij}$ is # of times $w_i$ occurs in context $c_j$

$$p_{ij} = \frac{f_{ij}}{\sum_{i=1}^{W}\sum_{j=1}^{C} f_{ij}} \qquad p_{i*} = \frac{\sum_{j=1}^{C} f_{ij}}{\sum_{i=1}^{W}\sum_{j=1}^{C} f_{ij}} \qquad p_{*j} = \frac{\sum_{i=1}^{W} f_{ij}}{\sum_{i=1}^{W}\sum_{j=1}^{C} f_{ij}}$$

| | computer | data | result | pie | sugar | count(w) |
|---|---|---|---|---|---|---|
| **cherry** | 2 | 8 | 9 | 442 | 25 | 486 |
| **strawberry** | 0 | 0 | 1 | 60 | 19 | 80 |
| **digital** | 1670 | 1683 | 85 | 5 | 4 | 3447 |
| **information** | 3325 | 3982 | 378 | 5 | 13 | 7703 |
| **count(context)** | 4997 | 5673 | 473 | 512 | 61 | 11716 |

$$pmi_{ij} = \log_2 \frac{p_{ij}}{p_{i*}p_{*j}} \qquad ppmi_{ij} = \begin{cases} pmi_{ij} & \text{if } pmi_{ij} > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$p_{ij} = \frac{f_{ij}}{\sum_{i=1}^{W}\sum_{j=1}^{C} f_{ij}}$$

| | computer | data | result | pie | sugar | count(w) |
|---|---|---|---|---|---|---|
| **cherry** | 2 | 8 | 9 | 442 | 25 | 486 |
| **strawberry** | 0 | 0 | 1 | 60 | 19 | 80 |
| **digital** | 1670 | 1683 | 85 | 5 | 4 | 3447 |
| **information** | 3325 | 3982 | 378 | 5 | 13 | 7703 |
| **count(context)** | 4997 | 5673 | 473 | 512 | 61 | 11716 |

- p(w=information,c=data) = 3982/111716 = .3399
- p(w=information) = 7703/11716 = .6575
- p(c=data) = 5673/11716 = .4842

$$p(w_i) = \frac{\sum_{j=1}^{C} f_{ij}}{N} \qquad p(c_j) = \frac{\sum_{i=1}^{W} f_{ij}}{N}$$

| | p(w,context) | | | | | p(w) |
|---|---|---|---|---|---|---|
| | **computer** | **data** | **result** | **pie** | **sugar** | **p(w)** |
| **cherry** | 0.0002 | 0.0007 | 0.0008 | 0.0377 | 0.0021 | 0.0415 |
| **strawberry** | 0.0000 | 0.0000 | 0.0001 | 0.0051 | 0.0016 | 0.0068 |
| **digital** | 0.1425 | 0.1436 | 0.0073 | 0.0004 | 0.0003 | 0.2942 |
| **information** | 0.2838 | 0.3399 | 0.0323 | 0.0004 | 0.0011 | 0.6575 |
| **p(context)** | 0.4265 | 0.4842 | 0.0404 | 0.0437 | 0.0052 | |

$$pmi_{ij} = \log_2 \frac{p_{ij}}{p_{i*}p_{*j}}$$

| | p(w,context) | | | | | p(w) |
|---|---|---|---|---|---|---|
| | **computer** | **data** | **result** | **pie** | **sugar** | **p(w)** |
| **cherry** | 0.0002 | 0.0007 | 0.0008 | 0.0377 | 0.0021 | 0.0415 |
| **strawberry** | 0.0000 | 0.0000 | 0.0001 | 0.0051 | 0.0016 | 0.0068 |
| **digital** | 0.1425 | 0.1436 | 0.0073 | 0.0004 | 0.0003 | 0.2942 |
| **information** | 0.2838 | 0.3399 | 0.0323 | 0.0004 | 0.0011 | 0.6575 |
| **p(context)** | 0.4265 | 0.4842 | 0.0404 | 0.0437 | 0.0052 | |

- pmi(information,data) = .3399 / (.6575*.4842) ) = .0944
  $\log_2$ (

Resulting PPMI matrix (negatives replaced by 0)

| | **computer** | **data** | **result** | **pie** | **sugar** |
|---|---|---|---|---|---|
| **cherry** | 0 | 0 | 0 | 4.38 | 3.30 |
| **strawberry** | 0 | 0 | 0 | 4.10 | 5.51 |
| **digital** | 0.18 | 0.01 | 0 | 0 | 0 |
| **information** | 0.02 | 0.09 | 0.28 | 0 | 0 |

# Weighting PMI

- PMI is biased toward infrequent events
  - Very rare words have very high PMI values
- Two solutions:
  - Give rare words slightly higher probabilities
  - Use add-one smoothing (which has a similar effect)

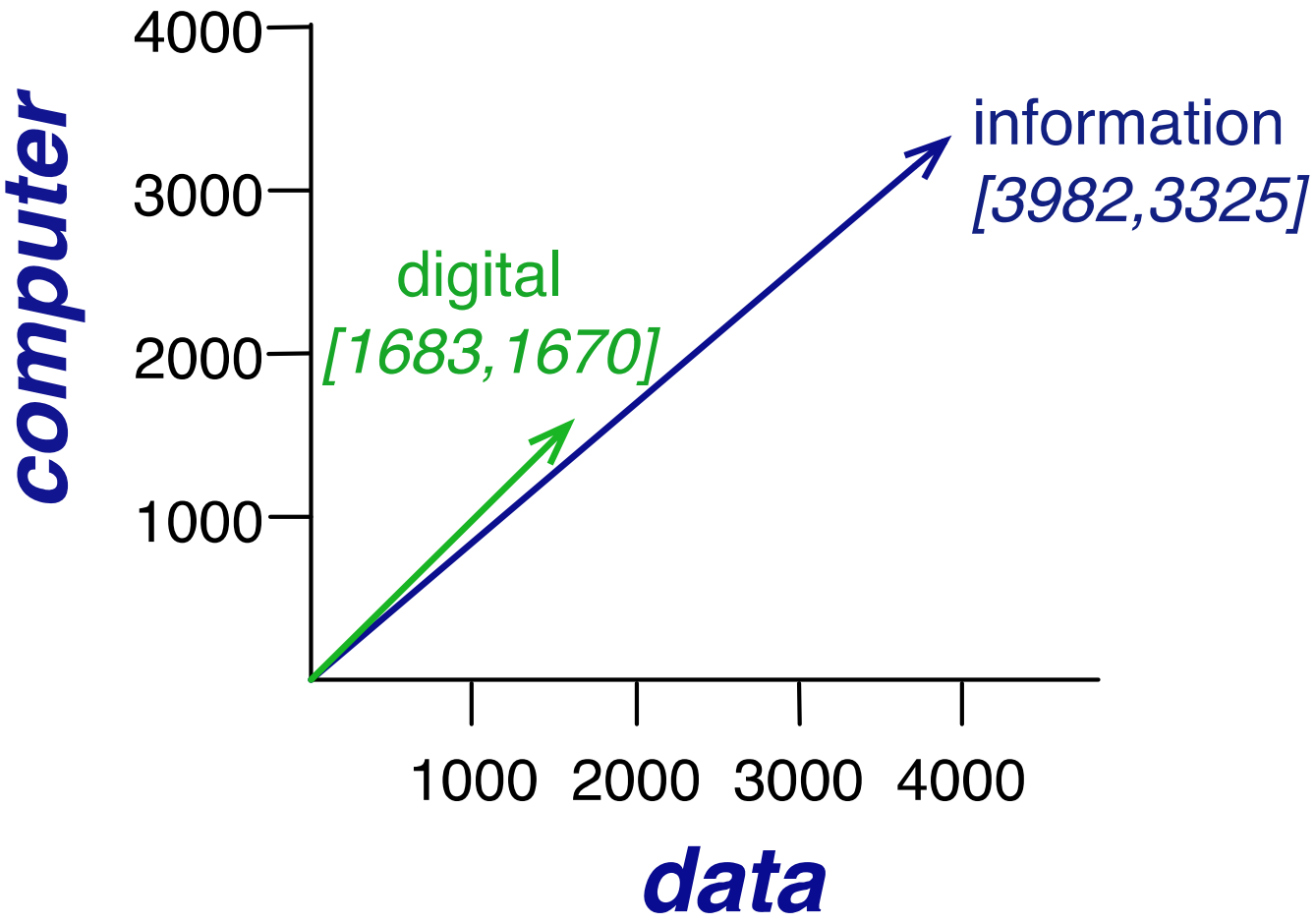
# Weighting PMI: Giving rare context words slightly higher probability

- Raise the context probabilities to $\alpha = 0.75$:

$$\text{PPMI}_\alpha(w,c) = \max(\log_2 \frac{P(w,c)}{P(w)P_\alpha(c)}, 0)$$

$$P_\alpha(c) = \frac{count(c)^\alpha}{\sum_c count(c)^\alpha}$$

- This helps because $P_\alpha(c) > P(c)$ for rare c
- Consider two events, P(a) = .99 and P(b)=.01
- $P_\alpha(a) = \frac{.99^{.75}}{.99^{.75}+.01^{.75}} = .97 \;\; P_\alpha(b) = \frac{.01^{.75}}{.01^{.75}+.01^{.75}} = .03$


4000
3000
2000
1000
computer
digital
[1683,1670]
information
[3982,3325]
1000
2000
3000
4000
data

# Co-occurrence Matrix

*One row per word with counts of co-occurrences with any other word*

| | | Context words | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | … | cook | eat | … | changed | broke | … |
| Words | cake | … | 10 | 20 | … | 0 | 0 | … |
| | steak | … | 12 | 22 | … | 0 | 0 | … |
| | engine | … | 0 | 0 | … | 3 | 10 | … |
| | tire | … | 0 | 0 | … | 10 | 1 | … |
| | … | … | … | … | … | … | … | … |

Rows of *C* capture similarity, yet *C* is still high dimensional and **sparse**.

**Semantic similarity** between terms **not always captured**

# But what about co-occurrences in large documents? Define a document a bit differently

Because that documents can be **anything, they don't have to be original documents.**

For example, we often treat each paragraph as a document

This allows us to calculate co-occurrences only in the smaller “document” (in this case a paragraph)

# Word-Context Matrix

A word-context (or word-word) matrix is a $|V| \times |V|$ matrix $C$ that **counts** the frequencies of co-occurrence of words in a collection of contexts (i.e, text spans of a given length).

*You cook the **cake** twenty minutes in the oven at 220 C.*
*I eat my **steak** rare.*
*I'll throw the **steak** if you cook it too much.*
*The **engine** broke due to stress.*
*I broke a **tire** hitting a curb, I changed the **tire**.*

$\text{Context}_{-2,+2}$('cake') = {['cook', 'the', 'twenty', 'minutes']}
$\text{Context}_{-2,+2}$('tire') = {['broke', 'a', 'hitting', 'a'], ['changed', 'the']}

# But raw frequency is a bad representation

- The co-occurrence matrices we have seen represent each cell by word frequencies.
- Frequency is clearly useful; if *sugar* appears a lot near *apricot*, that's useful information.
- But overly frequent words like *the*, *it,* or *they* are not very informative about the context
- It's a paradox! How can we balance these two conflicting constraints?

# Two common solutions for word weighting

- **Term frequency-inverse document frequency (TF-IDF):** Takes into account both the frequency of a word in a document (i.e., its **term frequency**) and the frequency of the word across all documents in a corpus (i.e., its **inverse document frequency**). Words that are **common in a specific document** but **rare in the overall corpus** are given **higher weight**, as they are considered **more important** to that particular document

- **Binary weighting:** Assigns a weight of **1** to each word that **appears** in a document and a weight of **0** to each word that **doesn't appear**. This method is useful for certain types of text classification tasks, such as **spam detection**, where the **presence or absence of specific words can be a strong indicator** of the class of a document.

# Relevance feedback: Basic idea – Recap

- Relevance feedback: user feedback on relevance of docs in initial set of results
  - The user issues a (short, simple) query.
  - The search engine returns a set of documents.
  - User marks some docs as relevant, some as nonrelevant.
  - **Search engine computes a new representation of the information need. Hope: better than the initial query.**
  - Search engine runs new query and returns new results.
  - New results have (hopefully) better recall.
- Idea: it may be difficult to formulate a good query when you don't know the collection well, so iterate


49

# Key concept for relevance feedback: Centroid – Recap

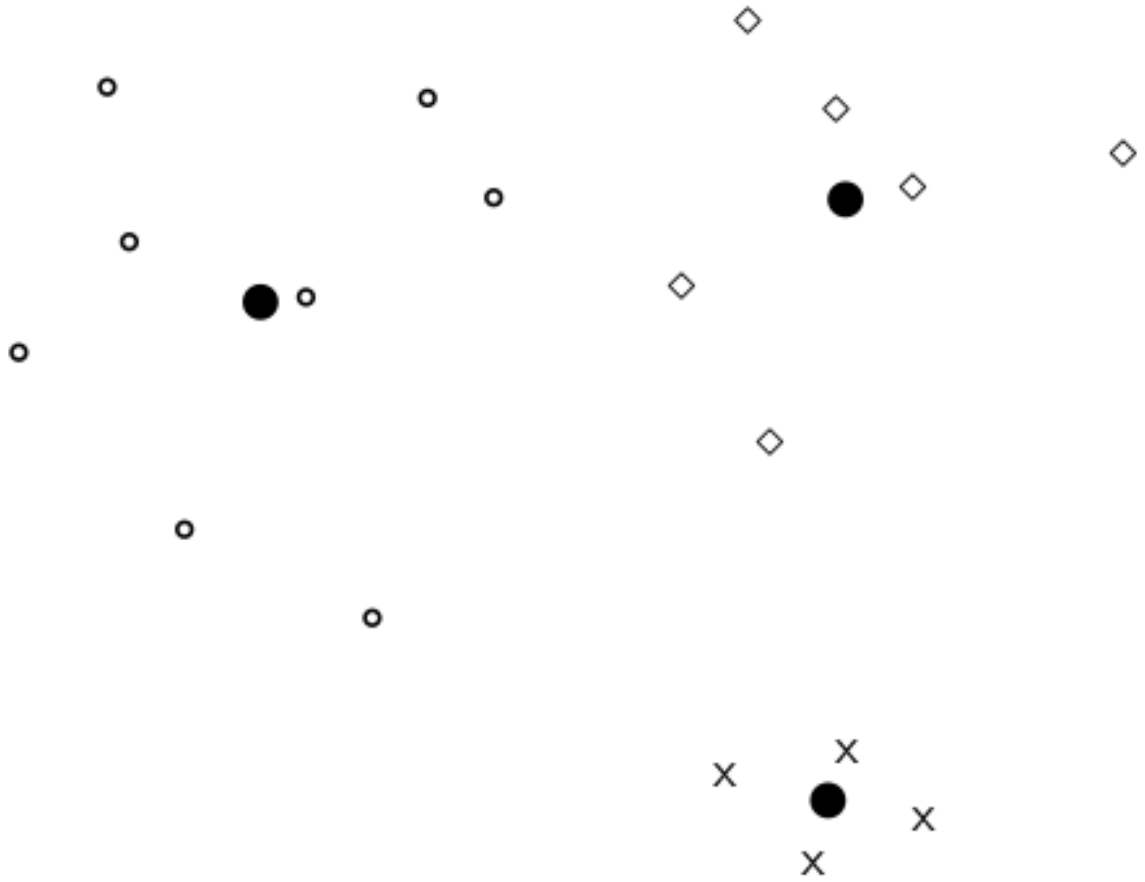
- The centroid is the center of mass of a set of points.
- Recall that we represent documents as points in a high-dimensional space.
- Thus: we can compute centroids of documents.
- Definition: $$\vec{\mu}(D) = \frac{1}{|D|} \sum_{d \in D} \vec{v}(d)$$

where D is a set of documents and $\vec{v}(d) = \vec{d}$ is the vector we use to represent document *d*.



50

# Centroid: Example – Recap





51

# Rocchio' algorithm – Recap

- The Rocchio' algorithm implements relevance feedback in the vector space model.
- Rocchio' chooses the query $\vec{q}_{opt}$ that maximizes

$$\vec{q}_{opt} = \arg\max_{\vec{q}}[\text{sim}(\vec{q}, \mu(D_r)) - \text{sim}(\vec{q}, \mu(D_{nr}))]$$

$D_r$ : set of relevant docs; $D_{nr}$ : set of nonrelevant docs

- Intent: ~$q_{opt}$ is the vector that separates relevant and nonrelevant docs maximally.
- Making some additional assumptions, we can rewrite $\vec{q}_{opt}$ :

$$\vec{q}_{opt} = \mu(D_r) + [\mu(D_r) - \mu(D_{nr})]$$



52

# Rocchio' algorithm – Recap

- The optimal query vector is:

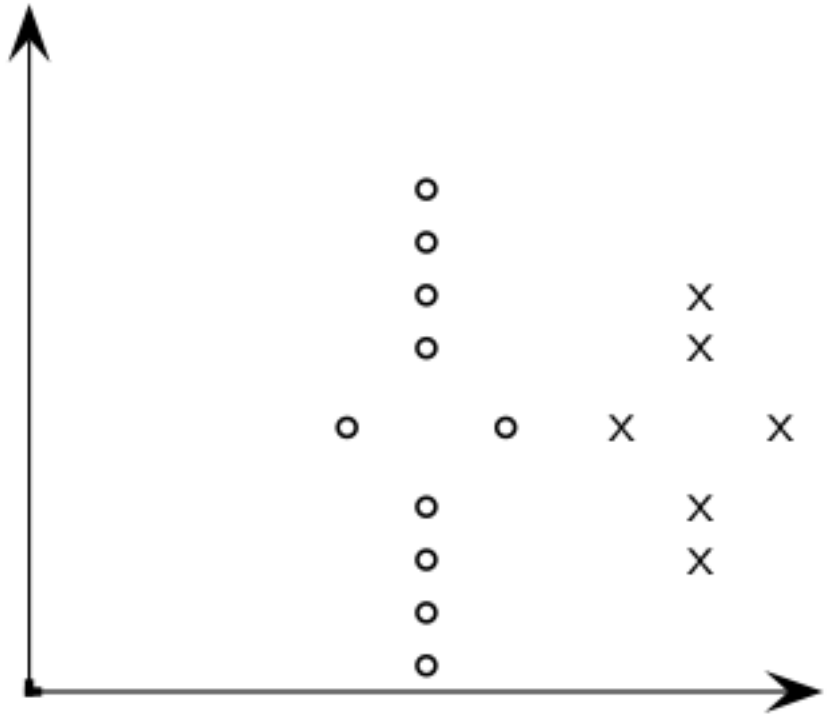
$$\begin{aligned}\vec{q}_{opt} &= \mu(D_r) + [\mu(D_r) - \mu(D_{nr})] \\ &= \frac{1}{|D_r|}\sum_{\vec{d}_j \in D_r} \vec{d}_j + [\frac{1}{|D_r|}\sum_{\vec{d}_j \in D_r} \vec{d}_j - \frac{1}{|D_{nr}|}\sum_{\vec{d}_j \in D_{nr}} \vec{d}_j]\end{aligned}$$

- We move the centroid of the relevant documents by the difference between the two centroids.



53

# Exercise: Compute Rocchio' vector – Recap

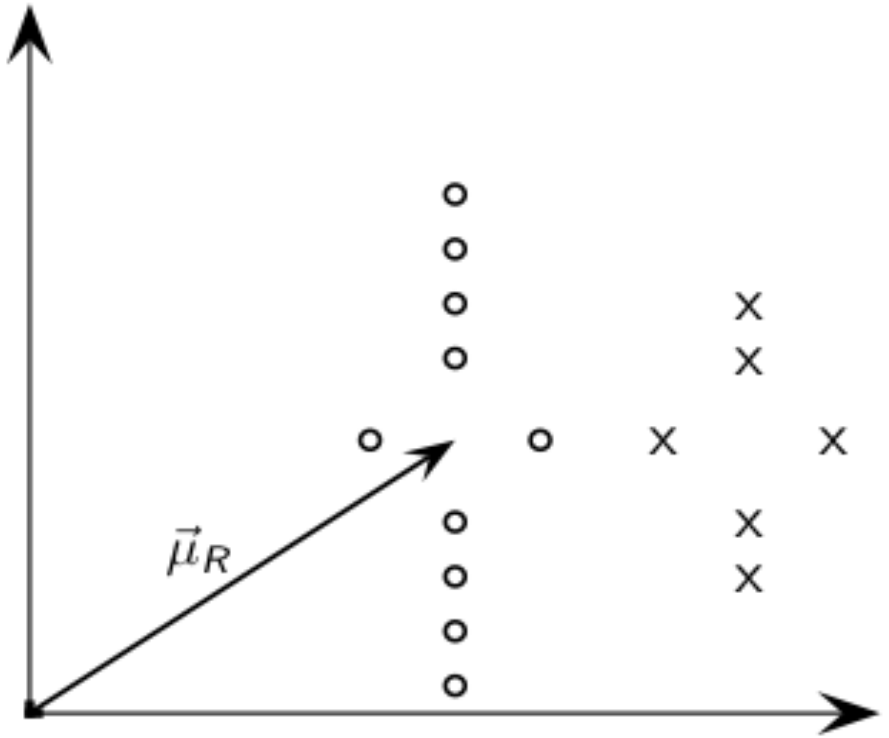


circles: relevant documents, Xs: nonrelevant documents



54

# Rocchio' illustrated – Recap

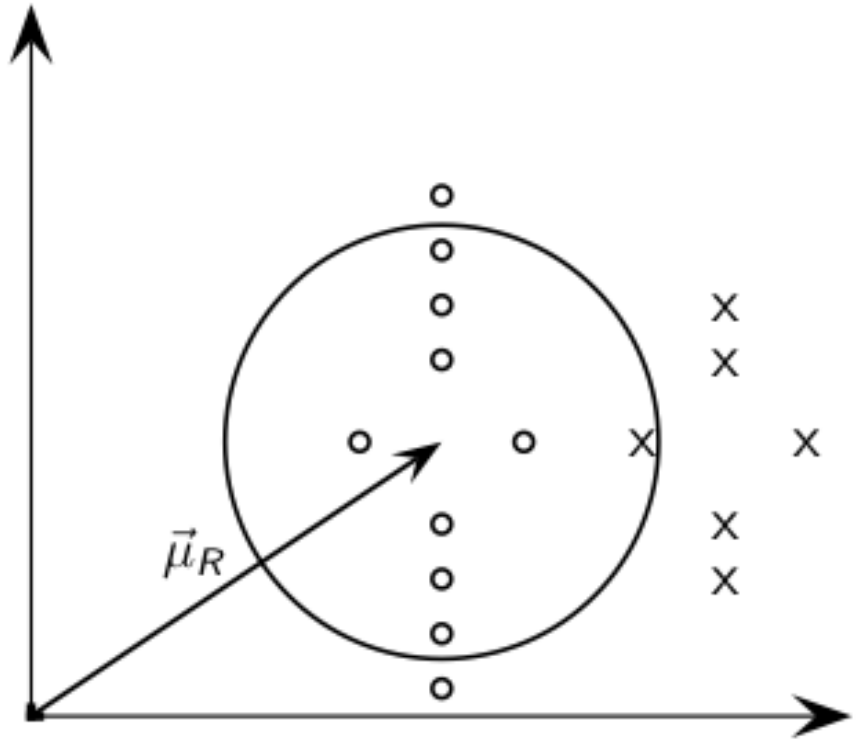



$\vec{\mu}_R$ : centroid of relevant documents



55

# Rocchio' illustrated – Recap

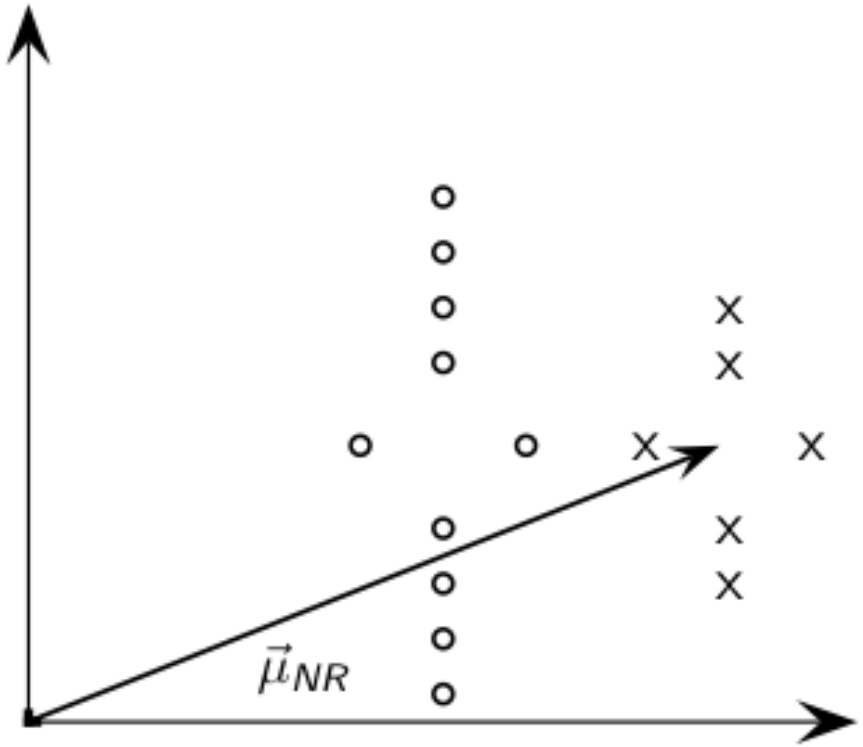



$\vec{\mu}_R$ does not separate relevant / nonrelevant.



56

# Rocchio' illustrated – Recap

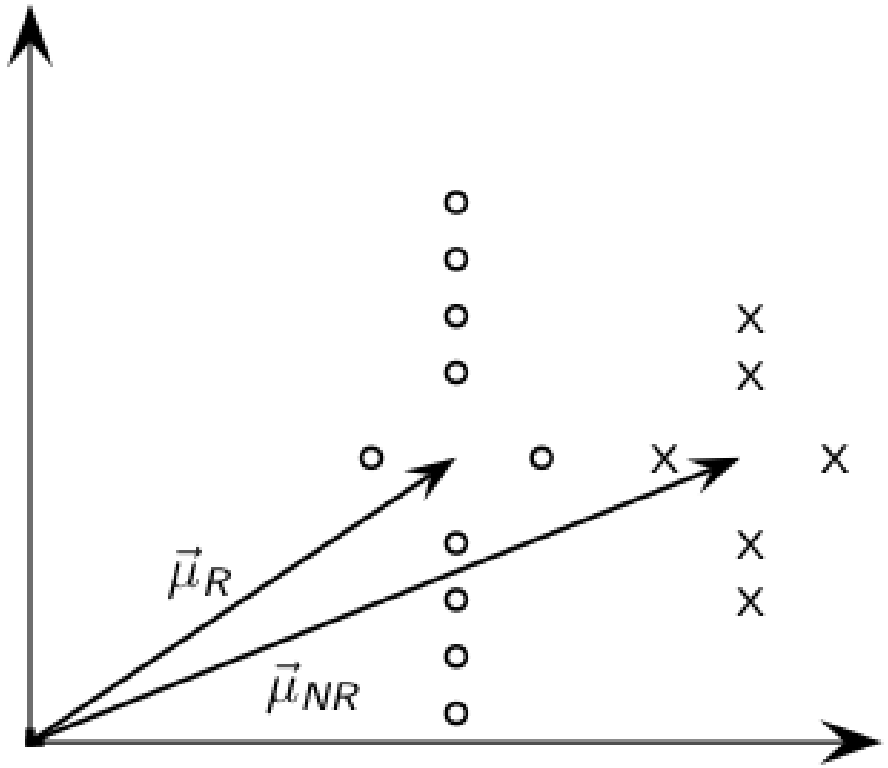



$\vec{\mu}_{NR}$: centroid of nonrelevant documents.



57

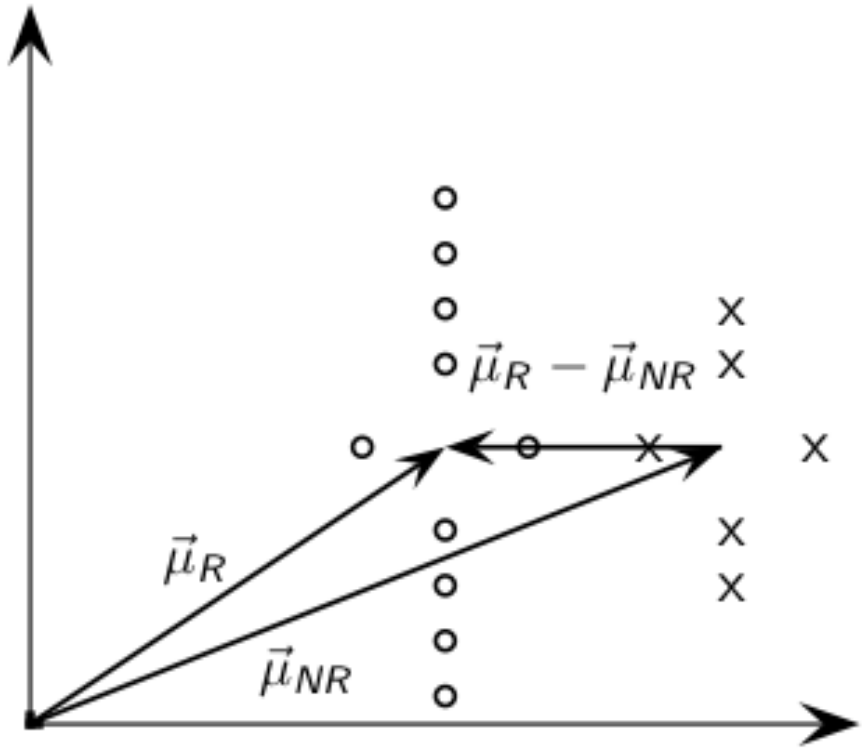
# Rocchio' illustrated – Recap






58

# Rocchio' illustrated – Recap

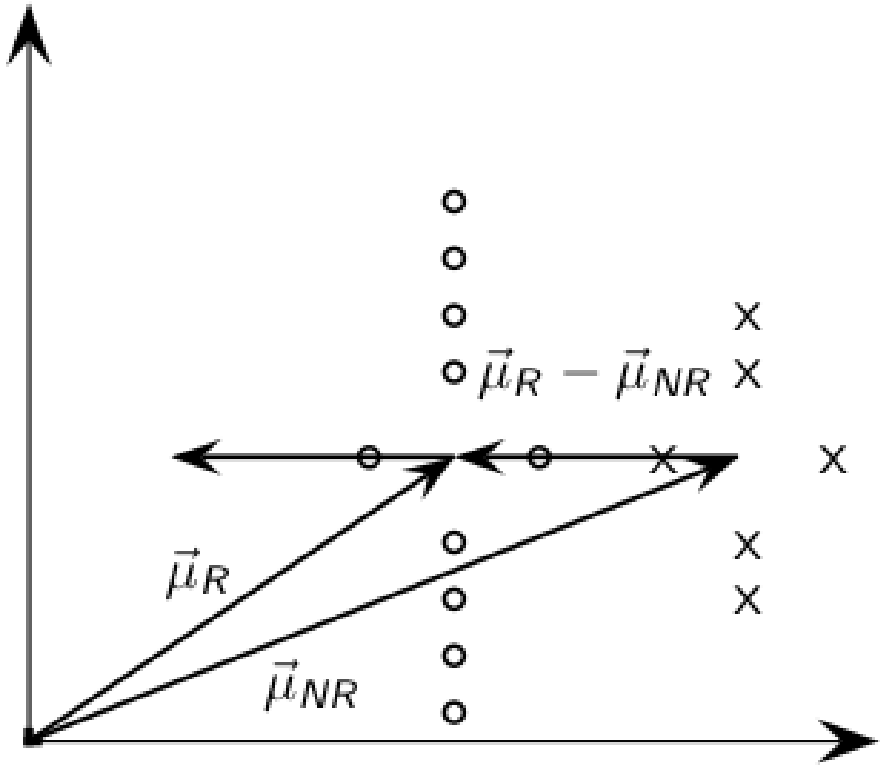



$\vec{\mu}_R$ - $\vec{\mu}_{NR}$: difference vector



59

# Rocchio' illustrated – Recap

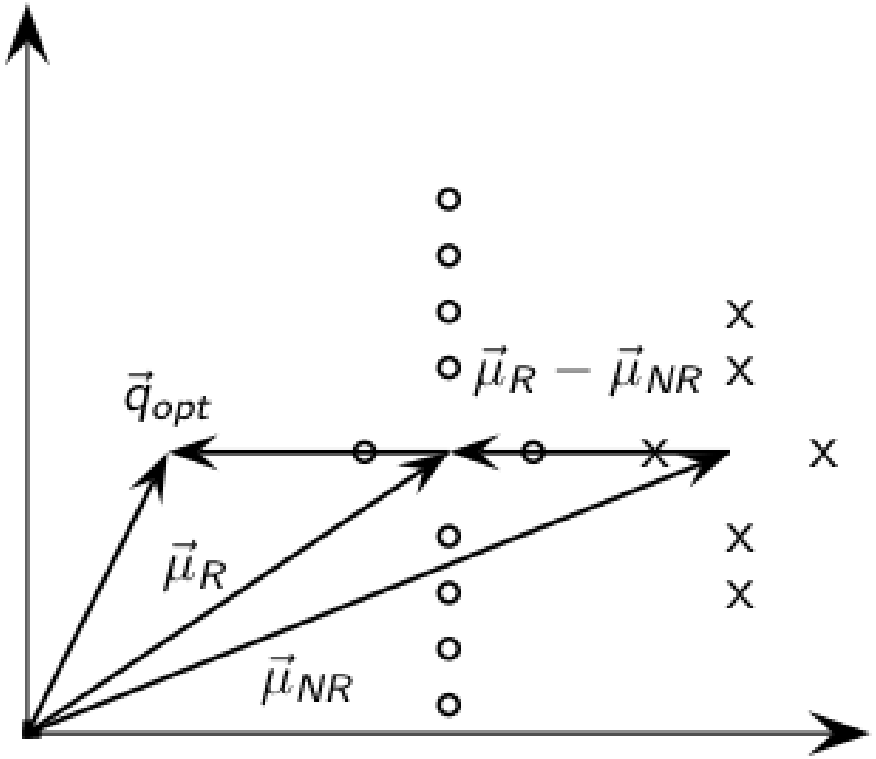



Add difference vector to … $\vec{\mu}_R$



60

# Rocchio' illustrated – Recap

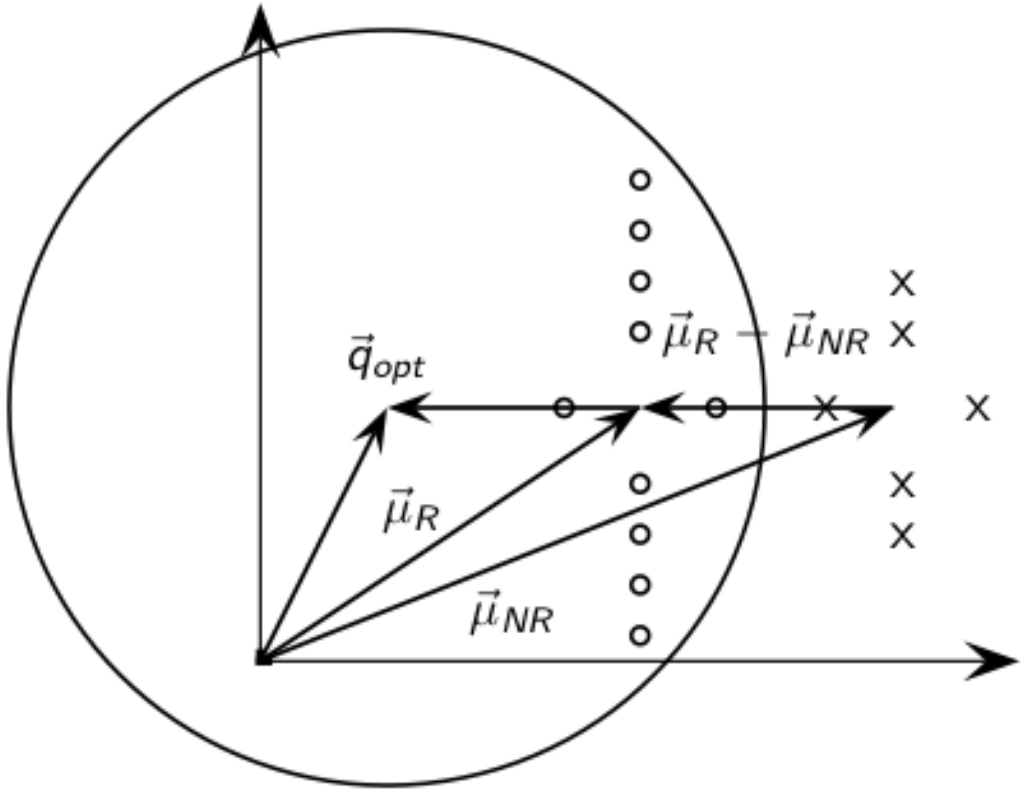



... to get $\vec{q}_{opt}$



61

# Rocchio' illustrated – Recap

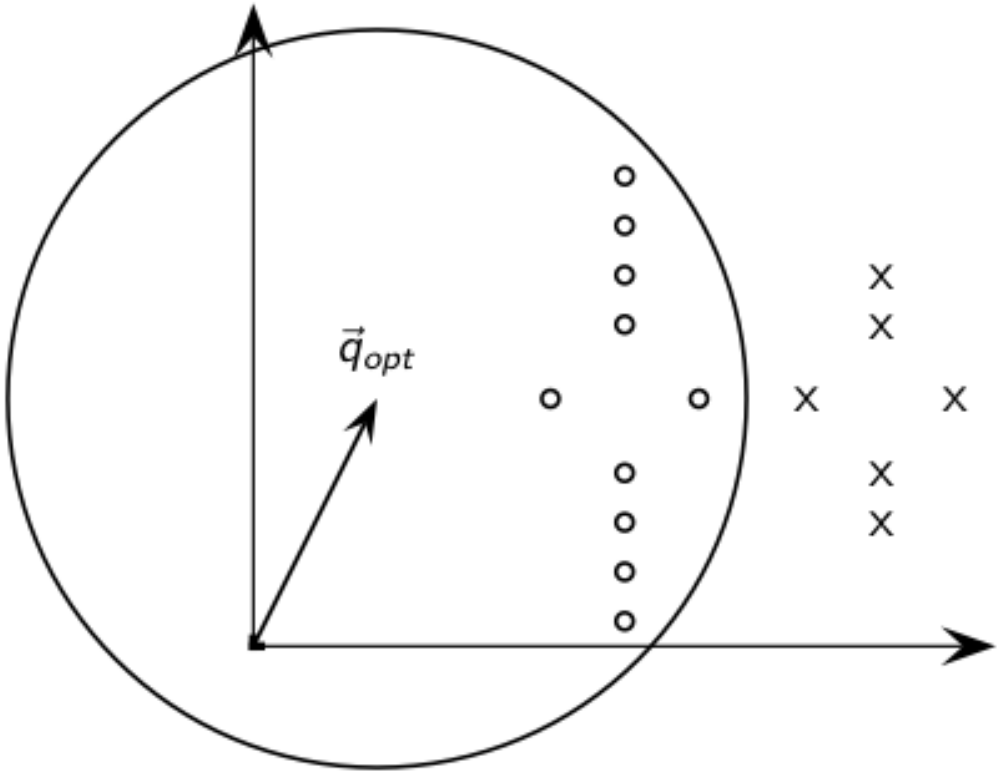



$\vec{q}_{opt}$ separates relevant / nonrelevant perfectly.



62

# Rocchio' illustrated – Recap




$\vec{q}_{opt}$ separates relevant / nonrelevant perfectly.



63

# Pseudo-relevance feedback – Recap

- Pseudo-relevance feedback automates the "manual" part of true relevance feedback.
- Pseudo-relevance algorithm:
  - Retrieve a ranked list of hits for the user's query
  - Assume that the top *k* documents are relevant.
  - Do relevance feedback (e.g., Rocchio)
- Works very well on average
- But can go horribly wrong for some queries.
- Several iterations can cause *query drift*.



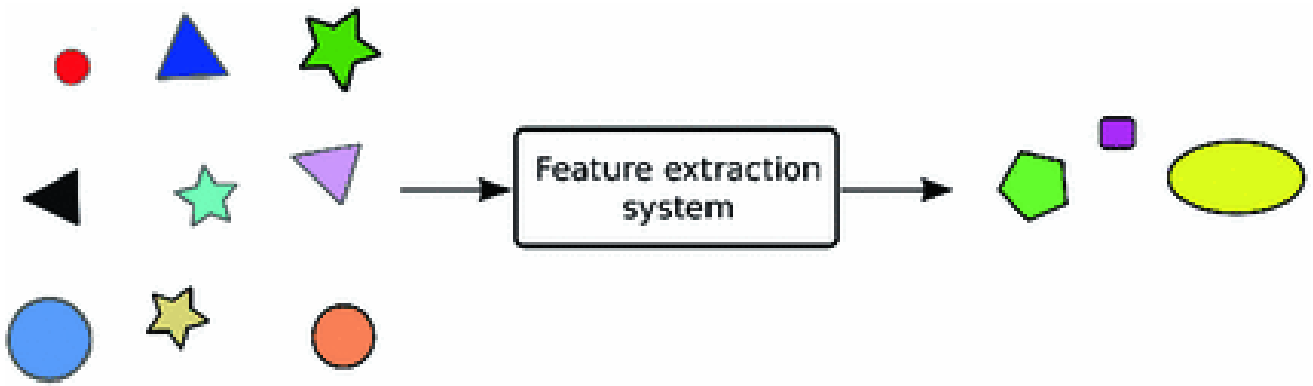
64

# Query expansion – Recap

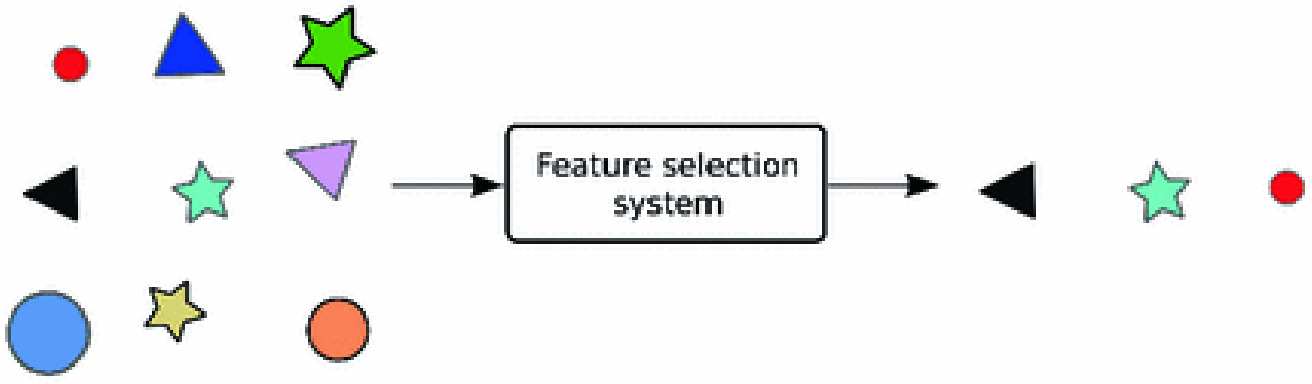
- Query expansion is another method for increasing recall.
- We use “global query expansion” to refer to “global methods for query reformulation”.
- In global query expansion, the query is modified based on some global resource, i.e. a resource that is not query-dependent.
- Main information we use: (near-)synonymy
- A publication or database that collects (near-)synonyms is called a thesaurus.
- We will look at two types of thesauri: manually created and automatically created.


65

# Feature selection




(a) Feature extraction




(b) Feature selection

# Feature selection – techniques - reminder

1. Low Variance
2. Remove highly correlated features
3. Select features with high correlation to target
4. Select features, with the assistance of the validation set

# Dimensionality Reduction

What is the difference between “simple” feature selection and dimensionality reduction?

- The difference is that the set of features made by feature selection must be a subset of the original set of features, and the set made by dimensionality reduction doesn't have to

# Dimensionality Reduction

What is the difference between “simple” feature selection and dimensionality reduction?

- Feature selection: Choosing $k<d$ important features, ignoring the remaining $d - k$
  - Subset selection algorithms
- Dimensionality reduction project the original $x_i$ , $i$ =1,…,$d$ dimensions to new $k<d$ dimensions, $z_j$ , $j$ =1,…,$k$
  - Dimensionality reductio, represent the data in less dimensions, and losses less information
  - Principal Components Analysis (PCA) – explained later

# Dimensionality Reduction – example

**Classification problem example:**

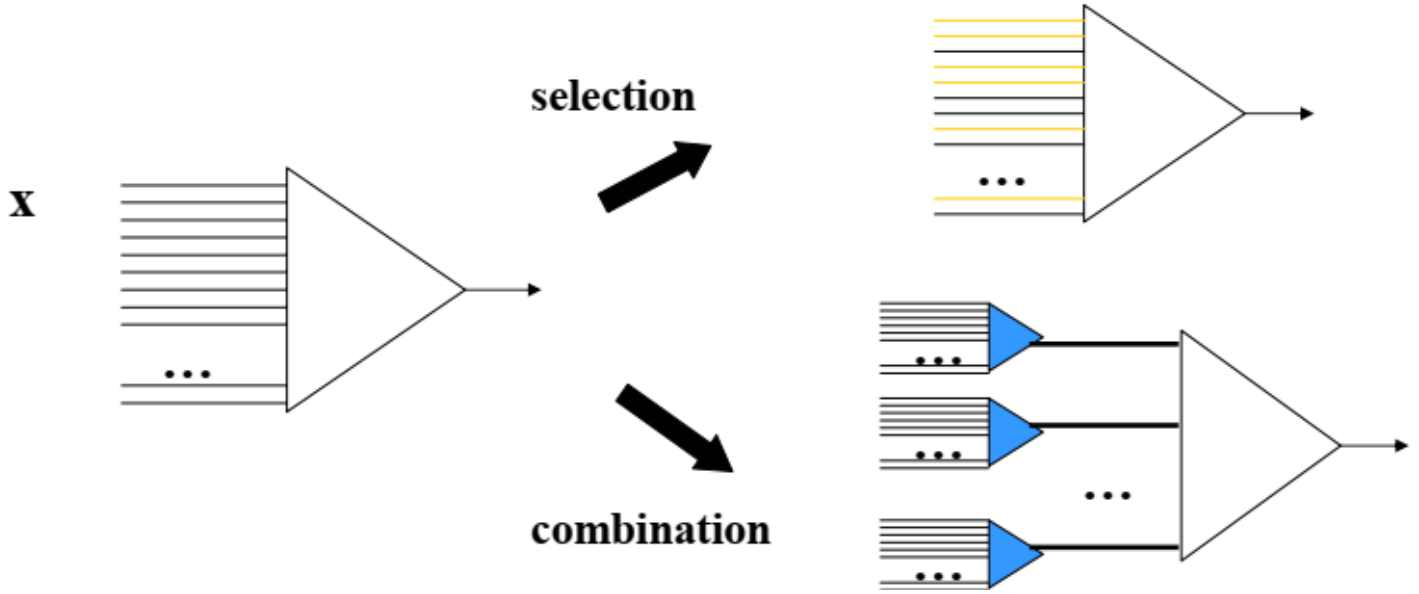
- We have an input data $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$ such that

$$\mathbf{x}_i = (x_i^1, x_i^2, ..., x_i^d)$$

  and a set of corresponding output labels $\{y_1, y_2, ..., y_N\}$
- Assume the dimension $d$ of the data point $\boldsymbol{x}$ is very large
- We want to classify $\boldsymbol{x}$

# Dimensionality Reduction – example

- **Solutions**:
  - **Selection of a smaller subset** of inputs (features) from a large set of inputs; train classifier on the reduced input set
  - **Combination of high dimensional inputs** to a smaller set of features $\phi_k(\mathbf{x})$; train classifier on new features



# הורדת מימדים (dimension reduction) – דוגמה

- **Face recognition problem**
  - Training data input: pairs of Image + Label(name)
  - Classifier input: Image
  - Classifier output: Label(Name)
- Image: Matrix of 256X256=65536 values in range 0..256
- Each pixels bear little information so can't select 100 best ones
- Average of pixels around specific positions may give an indication about an eye color.

# הורדת מימדים (dimension reduction) – דוגמה

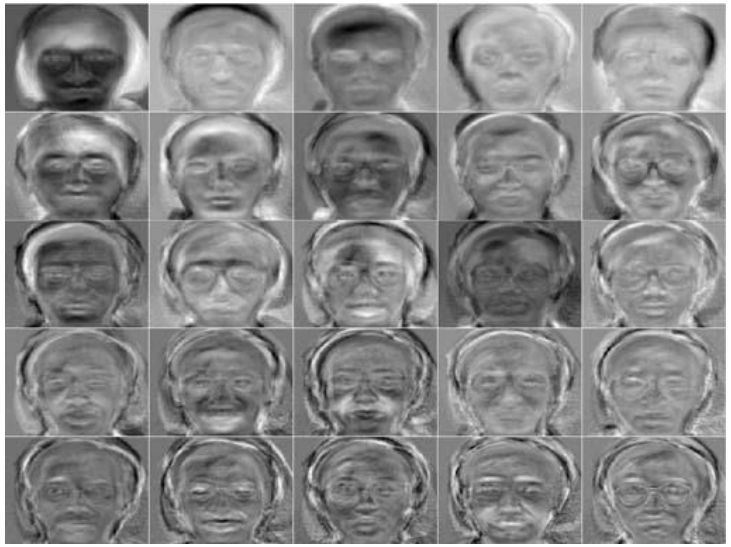
- Want to identify specific person, based on facial image
- Robust to glasses, lighting,...

  ⇒ Using simply the 256 x 256 pixels is a bad idea

# הורדת מימדים (dimension reduction) – דוגמה

Applying PCA: Eigenfaces

- How: Build one PCA database for the whole dataset
  and
  then classify based on the weights



# הורדת מימדים (dimension reduction) – דוגמה- **cons**

Requires carefully controlled data:

- All faces centered in frame
- Same size
- Some sensitivity to angle

Method is completely knowledge free

- (sometimes this is good!)
- Doesn’t know that faces are wrapped around 3D
- objects (heads)
- Makes no effort to preserve class distinctions

# Data - מוטיבציה – (dimension reduction) הורדת מימדים Compression

**Motivation I: Data Compression**

- We may want to reduce the dimension of our features if we have a lot of redundant data.
- Dimensionality reduction will reduce the total data we have to store in computer memory and will speed up our learning algorithm.

# הורדת מימדים (dimension reduction) – Data Compression מוטיבציה -

- If number of observables is increased
  - More time to compute
  - More memory to store inputs and intermediate results
  - More complicated explanations (knowledge from learning)
    - Regression from 100 vs. 2 parameters
  - No simple visualization
    - 2D vs. 10D graph
  - **Need much more data (curse of dimensionality)**
    - 1M of 1-d inputs is not equal to 1 input of dimension 1M

# הורדת מימדים (dimension reduction) – מוטיבציה - Data Compression הסבר

- Some features (dimensions) bear little or nor useful information (e.g. color of hair for a car selection)
  - Can drop some features
  - Have to estimate which features can be dropped from data

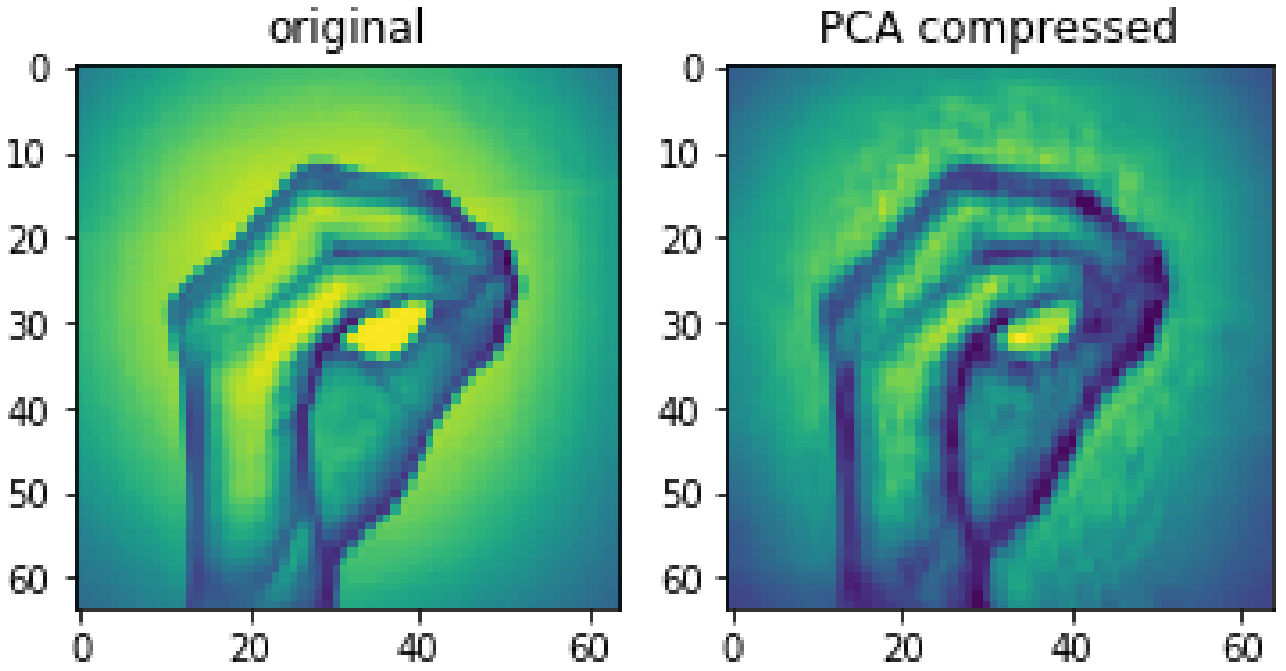
- Several features can be combined together without loss or even with gain of information (e.g. income of all family members for loan application)
  - Some features can be combined together
  - Have to estimate which features to combine from data

# הורדת מימדים (dimension reduction) – מוטיבציה - Data Compression - שימוש

- Have data of dimension d
- Reduce dimensionality to k<d
  - Discard unimportant features (we saw this also before)
  - Combine several features in one
- Use resulting k-dimensional data set for
  - Learning for classification problem (e.g. parameters of probabilities P(x|C)
  - Learning for regression problem (e.g. parameters for model y=g(x|Thetha)

# הורדת מימדים (dimension reduction) – Data
# Compression – דוגמה 2

- דוגמה – אפליקציה לזיהוי שפת הסימנים
- הורדת מימדיות מ-4096 --> 292



# הורדת מימדים (dimension reduction) –דוגמה

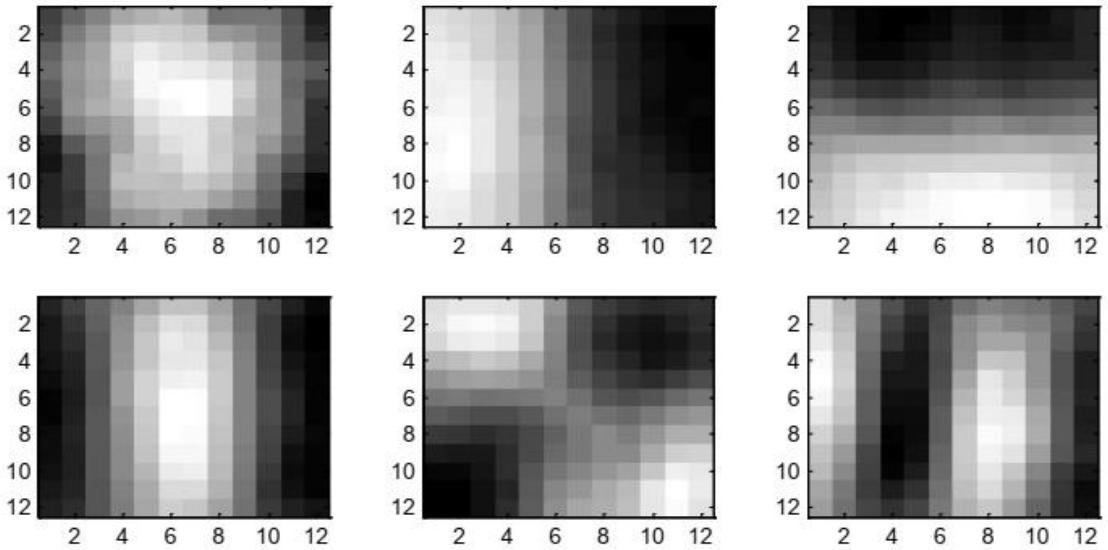
- Divide the original 372x492 image into patches:
  - Each patch is an instance that contains 12x12 pixels on a grid
- Consider each as a 144-D vector

# הורדת מימדים (dimension reduction) –דוגמה

הורדת המימדים D144 --> D6

# הורדת מימדים (dimension reduction) –דוגמה

6 most important eigenvectors:

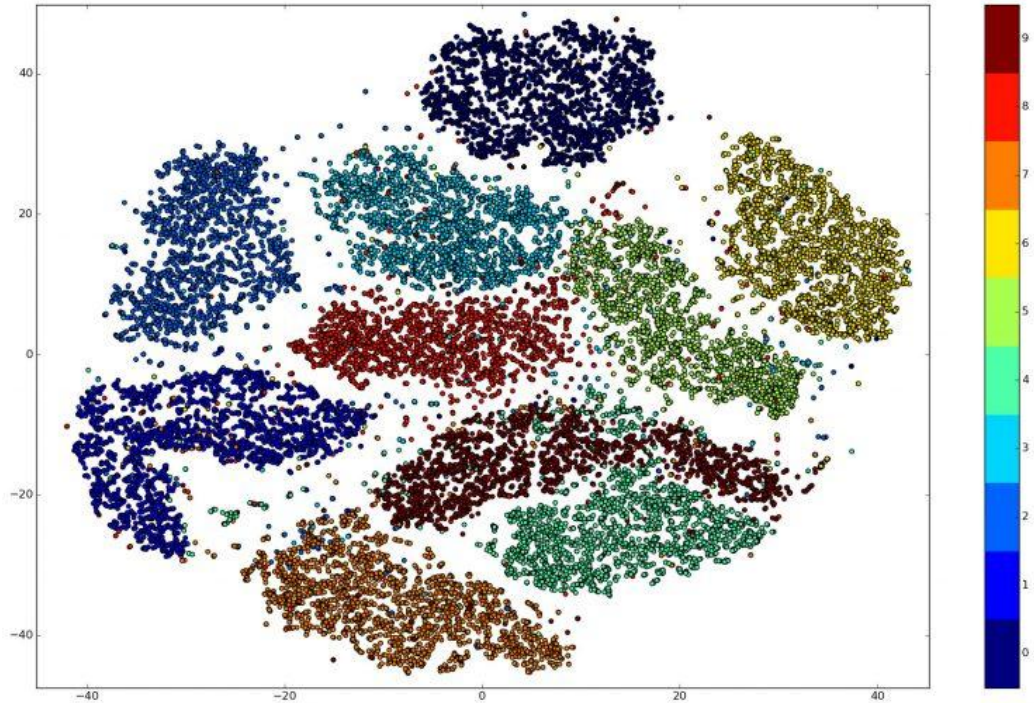


# הורדת מימדים (dimension reduction) – מוטיבציה - Visualization

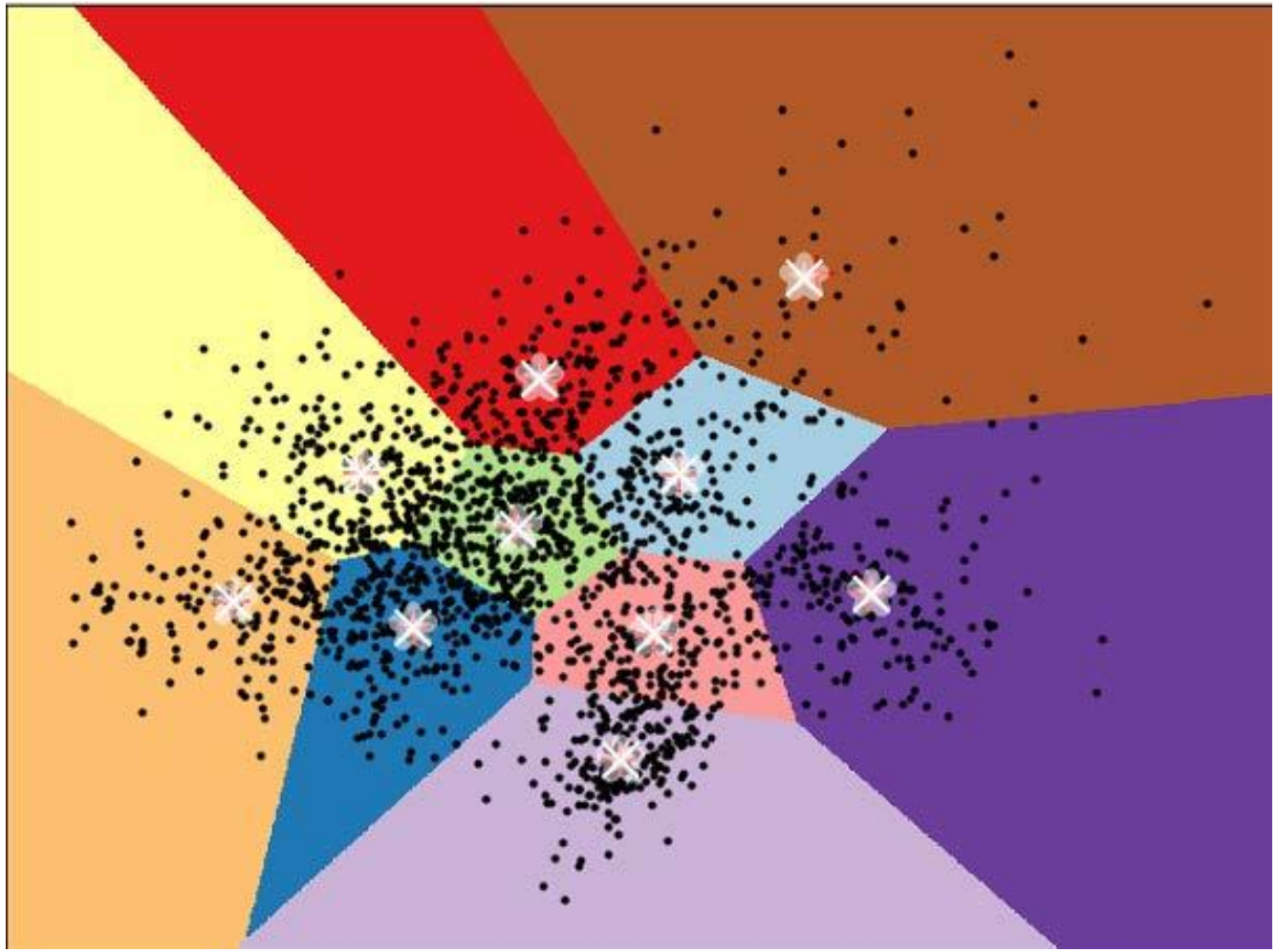
**Motivation II: Visualization**

- It is not easy to visualize data that is more than three dimensions. We can reduce the dimensions of our data to 3 or less in order to plot it
- We need that can deficiently summarize all the other features.
- Data exploration - The right visualization method may reveal problems with the experimental data.

# הורדת מימדים (dimension reduction) – Visualization – דו'
# מ- 10D ל- 2D



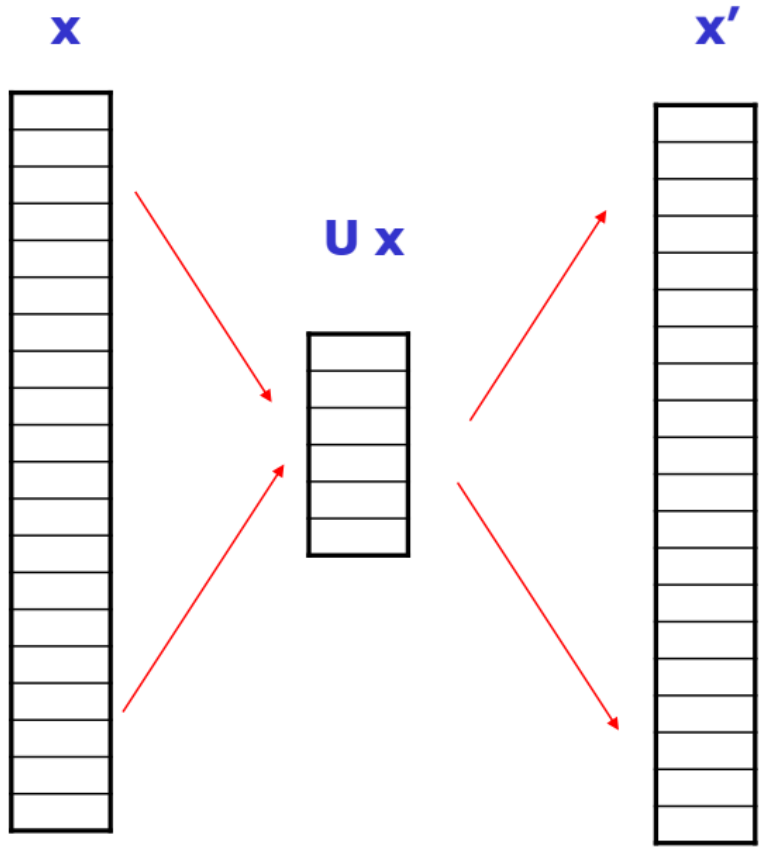
# הורדת מימדים (dimension reduction) – Visualization– דו' עבור k-mean (k=10)



# Noise - מוטיבציה – (dimension reduction) הורדת מימדים reduction

**Motivation III: Noise reduction**

- By selecting most significant eigenvectors and reproducing



# הורדת מימדים (dimension reduction) – מוטיבציה - Noise
# דוגמה - reduction

Noisy image

# הורדת מימדים (dimension reduction) – מוטיבציה - Noise
# דוגמה - reduction

De-noised image

# הורדת מימדים (dimension reduction) – מוטיבציה – Deriving new data

**Motivation IV: Deriving new data**

- Here the goal is opposite from feature selection, the goal here is to find correlation within the features in order to find new knowledge

# הורדת מימדים (dimension reduction) – מוטיבציה – דוגמה - Deriving new data

- Vector Representation

We can define a word by a vector of counts over contexts, For Example:

| | song | cucumber | meal | black |
|---|---|---|---|---|
| **tomato** | **0** | **6** | **5** | **0** |
| book | 2 | 0 | 2 | 3 |
| **pizza** | **0** | **2** | **4** | **1** |

- Each word is associated with a vector of dimension |V| (the size of the vocabulary)
- We expect similar words to have similar vectors
- Given the vectors of two words, we can determine their similarity (more about that later)

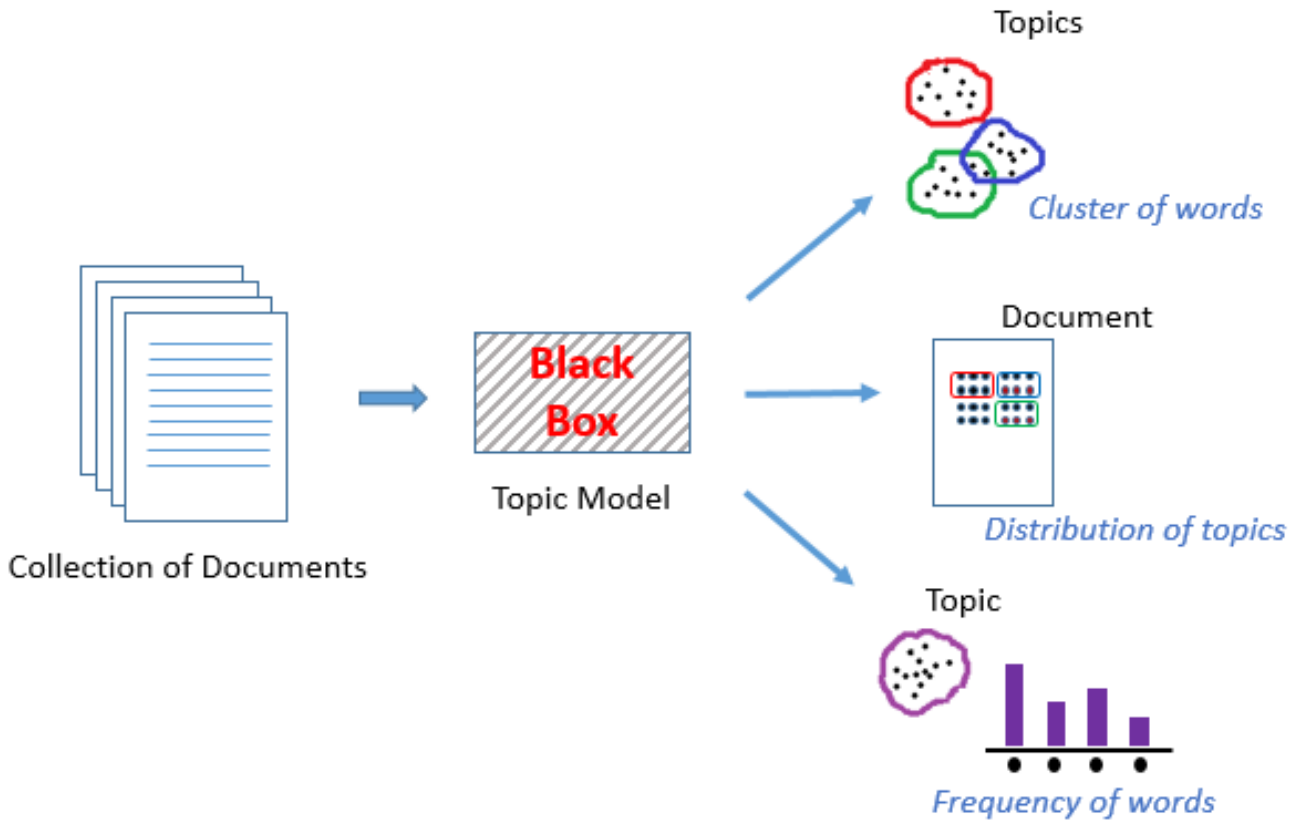
---

These vectors are:

- huge – each of dimension |V| (the size of the vocabulary ~ $100K$ +)

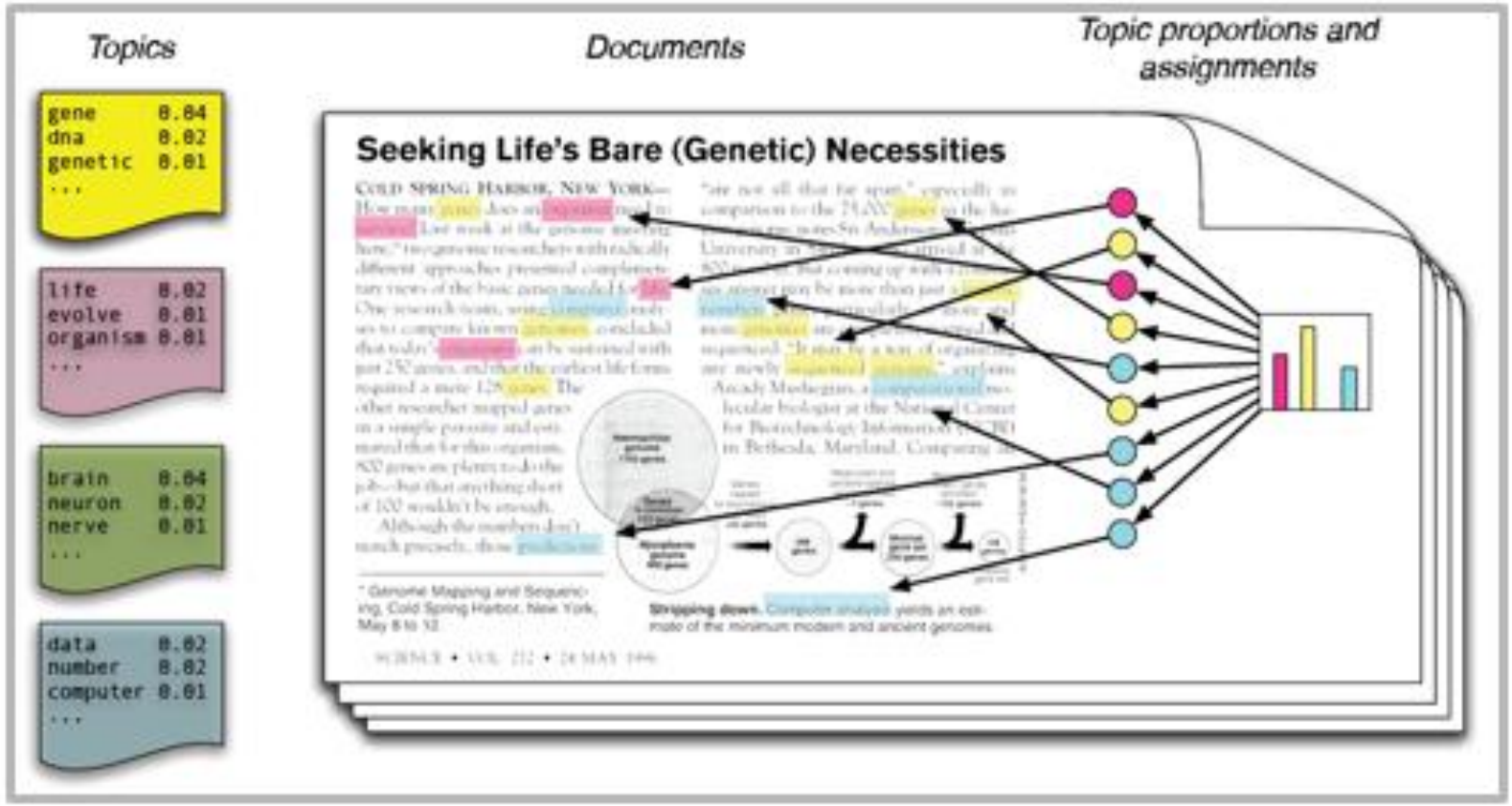
# הורדת מימדים (dimension reduction) – מוטיבציה – דוגמה - Deriving new data

Example: Topic modeling (using vector representation for similarity)



# הורדת מימדים (dimension reduction) – מוטיבציה – דוגמה - Deriving new data

Example: Topic modeling (using vector representation for similarity)



# הורדת מימדים (dimension reduction) – הגדרה

הורדת המימדים – היא בעיה הפוכה מפונקציות ה- kernel שראינו.

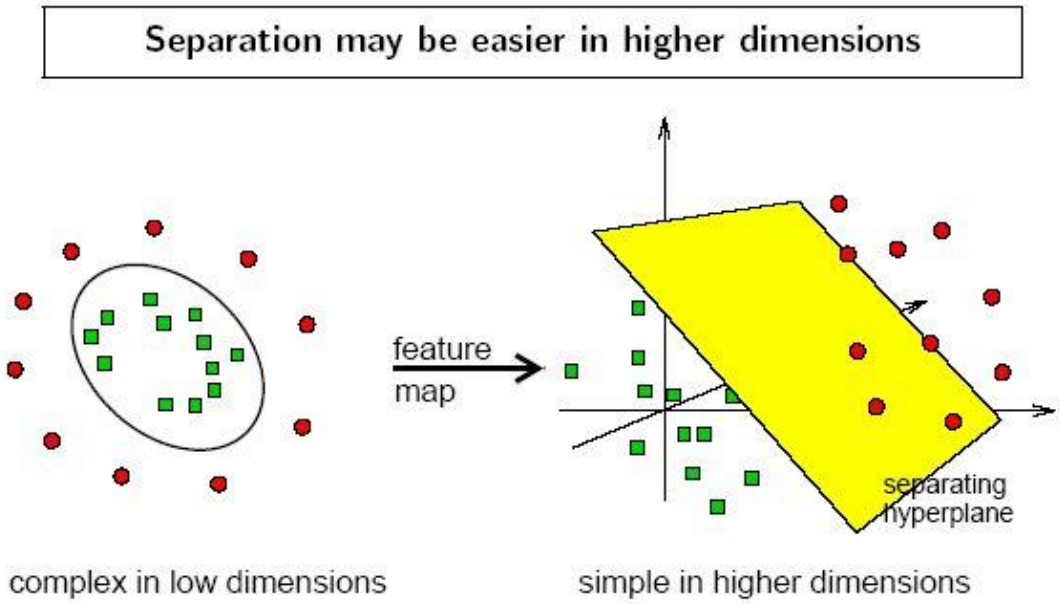
הגדרת הורדת המימדים:

- נתונות לנו ***n*** דוגמאות במימד ***d***
- נרצה למצוא יצוג לכל הדוגמאות במימד d נמוך יותר (***k<d***)

איך עושים זאת?

# תזכורת - kernel פונקציות

## The Kernel Trick Revisited




$$\text{Polynomial kernel } K(x, y) = (x' y)^2$$

$$\phi(x_1, x_2) = (x_1^2, \sqrt{2}x_1 x_2, x_2^2)$$

בפונקציות ה-kernel השתמשנו בפונקציה המתמירה את המישור של המאפיינים שלנו (ה-feature space) למישור גבוה יותר

השתמשנו בפונקציות אלו על מנת לבצע הפרדה לינארית ב-kernel k-means

# הורדת מימדים (dimensionality reduction) – הגדרה

הגדרת הורדת המימדים:

- נתונות לנו $n$ דוגמאות במימד $d$
- נרצה למצוא יצוג לכל הדוגמאות במימד d נמוך יותר ($k<d$)

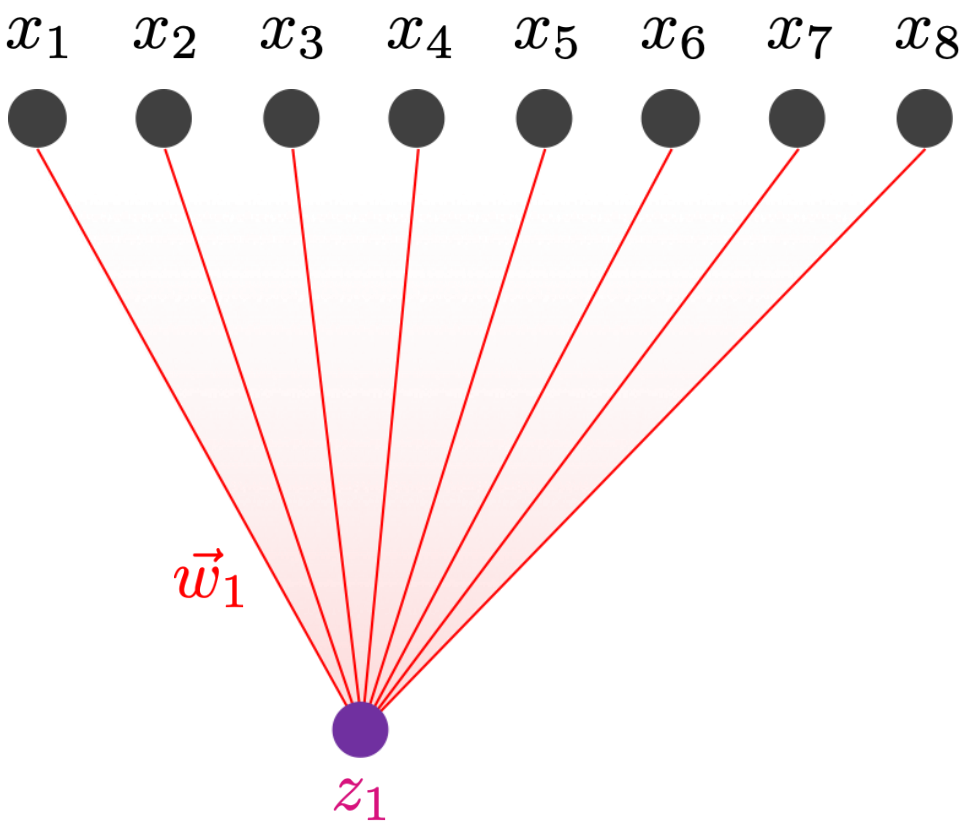
Feature selection vs. dimensionality reduction

- ב-Feature Selection בוחרים רק חלק מהמאפיינים, וחלק מסננים.
- ב-dimensionality reduction מייצגים את המאפיינים בפחות מימדים (עם איבוד מידע מנמלי).

איך עושים זאת? הטלה ממימד $d$ למימד $k$

- PCA – דו' בה ההיטל מורכב מקומבינציות לינאריות של המאפיינים
- tSNE – דו' בה ההיטל מורכב קומבינציות לא לינאריות של המאפיינים

# הורדת מימדים - PCA




$$z_1 = \vec{w}_1 \cdot \vec{x}$$

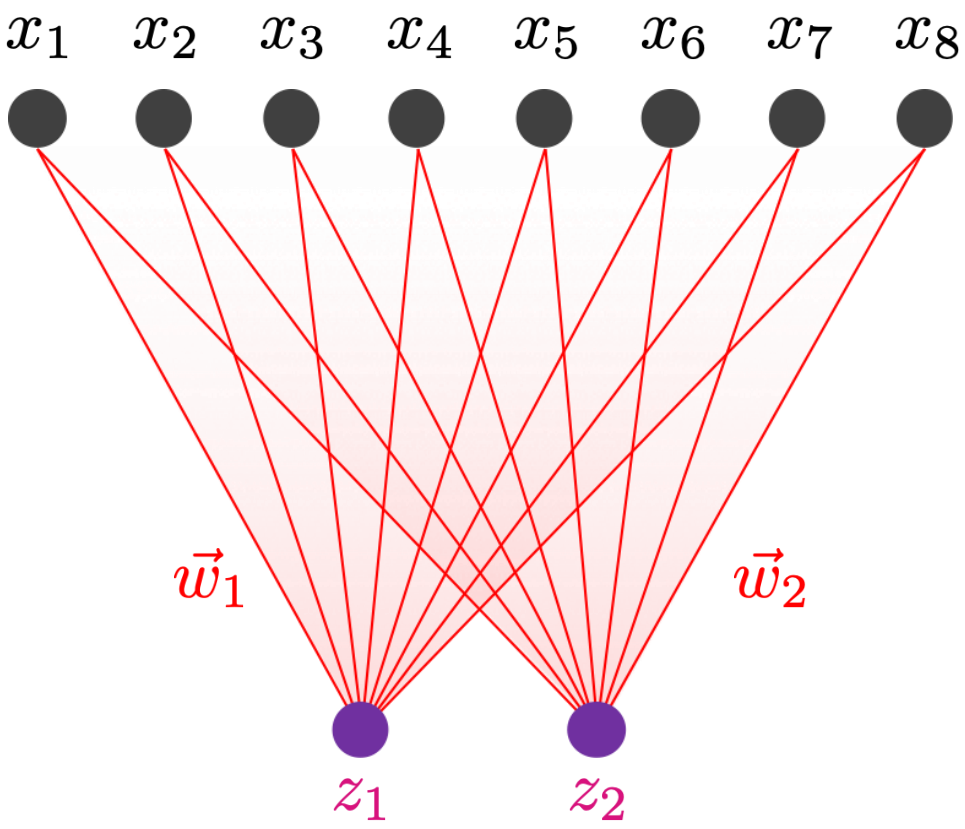
# הורדת מימדים - PCA

- PCA – Principal component analysis
  - נתונות לנו ***n*** דוגמאות במימד ***d***
  - נרצה למצוא יצוג לכל הדוגמאות במימד נמוך יותר (***k<d***)
  - האמצעי: קומבינציות לינאריות של המאפיינים
    כלומר: הטלה ממימד ***d*** למימד ***k***

$$\vec{x}_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,d}) \qquad z_{i,j} = \vec{w}_j \cdot \vec{x}_i$$

$$\vec{z}_i = \left(\vec{w}_1 \cdot \vec{x}_i\ ,\ \vec{w}_2 \cdot \vec{x}_i\ ,\ \ldots\ ,\ \vec{w}_k \cdot \vec{x}_i\right) = (z_{i,1}, z_{i,2}, \ldots, z_{i,k})$$

# הורדת מימדים - PCA



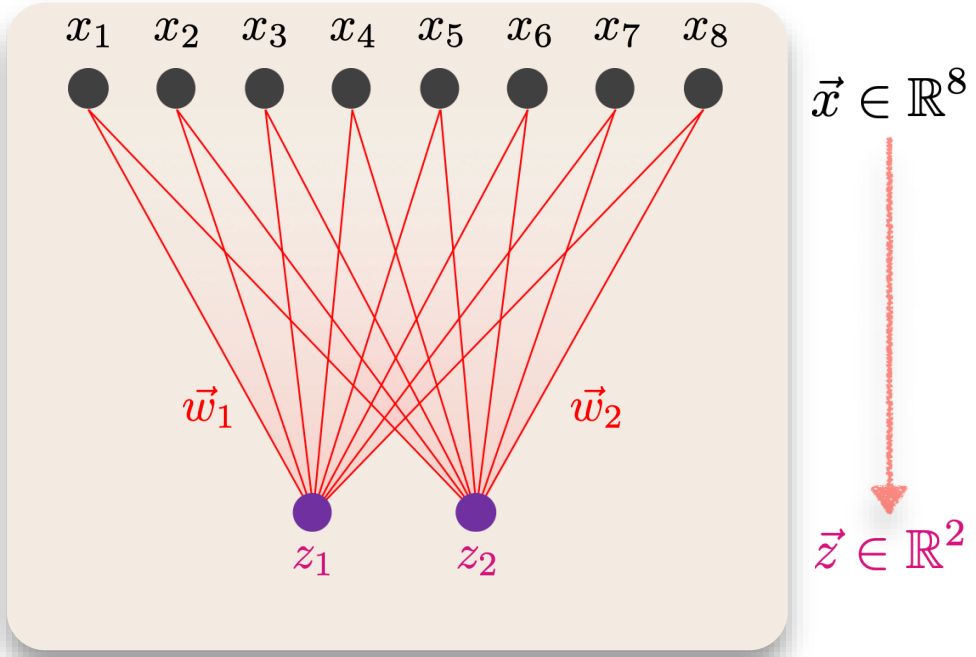

$$z_1 = \vec{w}_1 \cdot \vec{x} \qquad z_2 = \vec{w}_2 \cdot \vec{x}$$

# הורדת מימדים - PCA

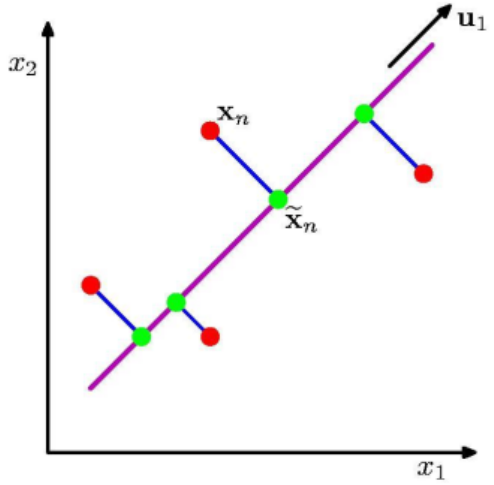
- מחפשים ליצג את $\vec{x} \in \mathbb{R}^d$ באמצעות $\vec{z} \in \mathbb{R}^k$
- ע"י שימוש בקומבינציות לינאריות של $\vec{w}_1, \ldots, \vec{w}_k$ של המאפיינים.

ש: איך נבחר את $\vec{w}_1, \ldots, \vec{w}_k$?

ת: שגיאת שחזור מינימלית.



# PCA: Motivation




**PCA:**

Orthogonal projection of the data onto a lower-dimension linear space that...

- ❑ maximizes variance of projected data (purple line)
- ❑ minimizes the mean squared distance between
  - data point and
  - projections (sum of blue lines)

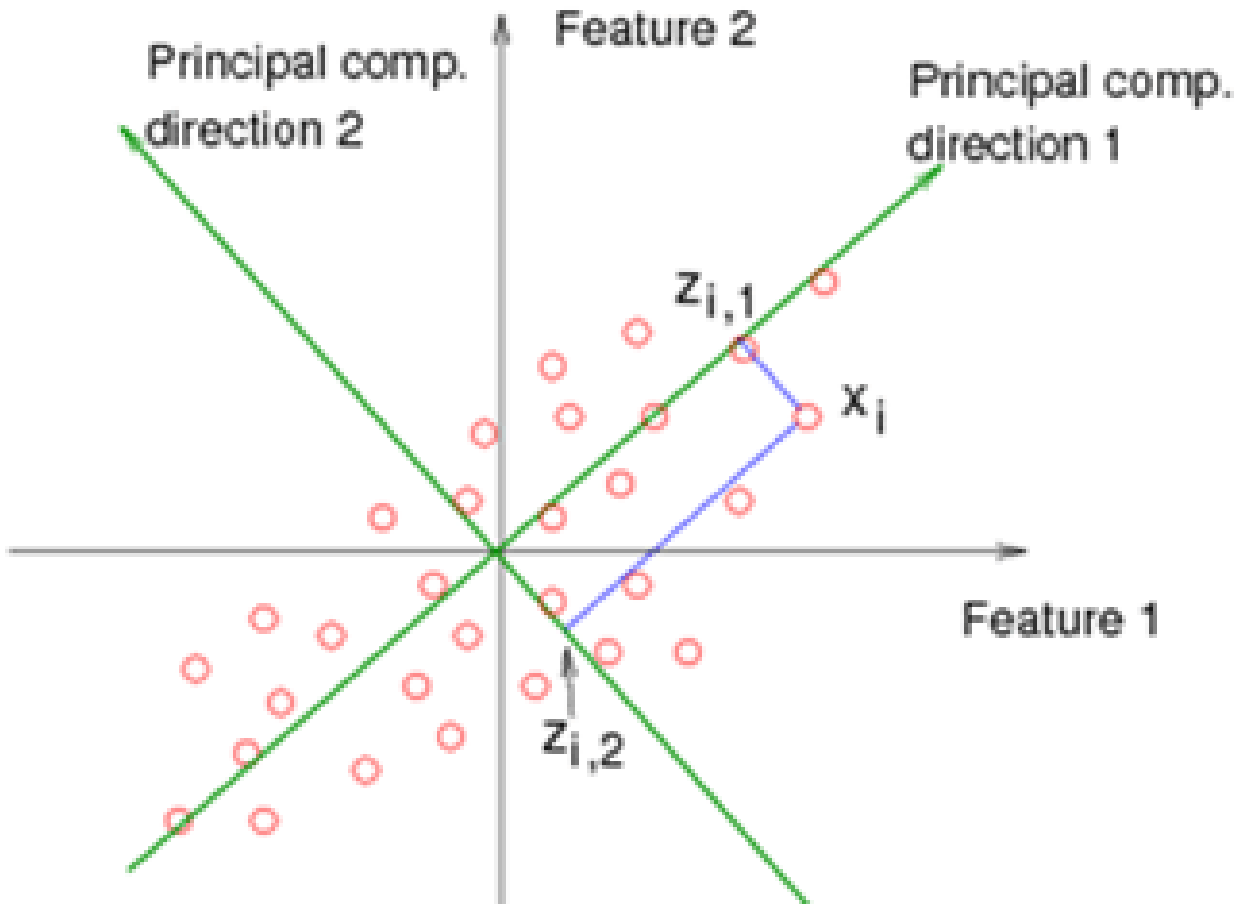
# PCA: Motivation

- Choose directions such that a total variance of data will be maximum
  - Maximize Total Variance

- Choose directions that are orthogonal
  - Minimize correlation

- Choose k<d orthogonal directions which maximize total variance

# PCA: Motivation



# PCA – the idea

**Idea:**

- Given data points in a d-dimensional space, project them into a lower dimensional space while preserving as much information as possible.
  - Find best 2D approximation of 3D data
  - Find best 12-D approximation of $10^4$-D data

- In particular, choose projection that minimizes squared error in reconstructing the original data.

# PCA - properties

**Properties:**

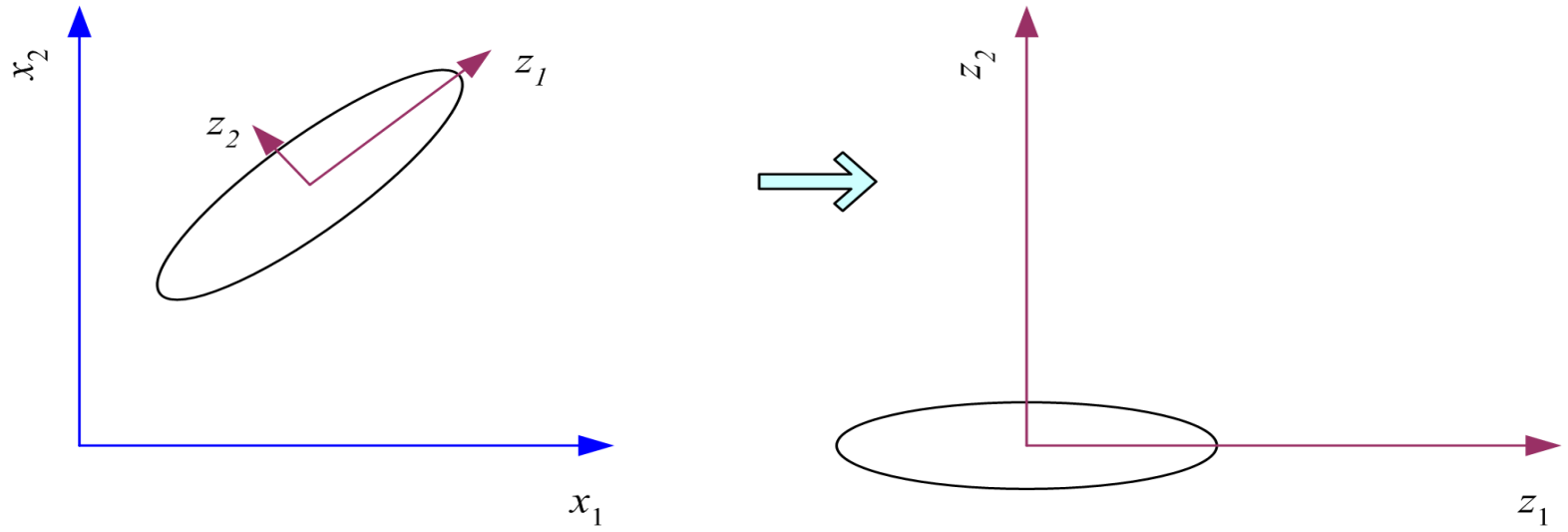
- **PCA Vectors** originate from the center of mass.
- Principal component #1: points in the direction of the **largest variance**.
- Each subsequent principal component
  - is **orthogonal** to the previous ones, and
  - points in the directions of the **largest variance of the residual subspace**

# What PCA does

$$\boldsymbol{z} = \mathbf{U}^T(\boldsymbol{x} - \boldsymbol{m})$$

where the columns of **U** are the eigenvectors of **∑**, and ***m*** is sample mean

Centers the data at the origin and rotates the axes



# What PCA does

**PCA:**

- linear transformation of $d$ dimensional input $x$ to M dimensional feature vector $z$ such that $M < d$ under which the retained variance is maximal.
- Task independent

**Fact:**

- A vector $x$ can be represented using a set of orthonormal vectors $u$

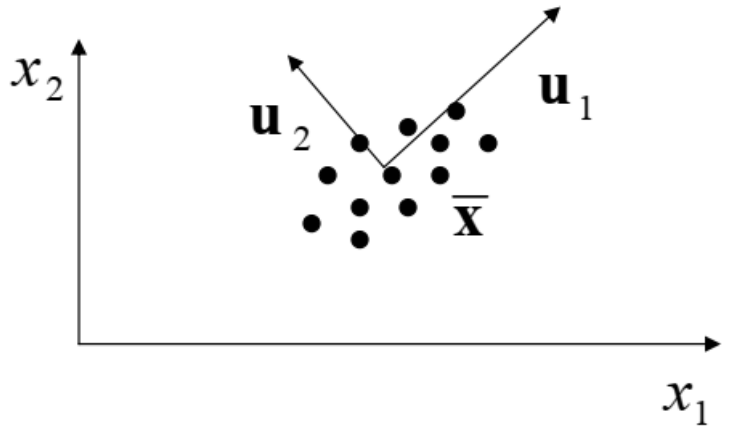
$$\mathbf{x} = \sum_{i=1}^{d} z_i \mathbf{u}_i$$

- Leads to transformation of coordinates (from $x$ to $z$ using $u$'s)

$$z_i = \mathbf{u}_i^T \mathbf{x}$$

# PCA

Once eigenvectors $\mathbf{u}_i$ with largest eigenvalues are identified, they are used to transform the original *d*-dimensional data to *M* dimensions




To find the "true" dimensionality of the data *d'* we can just look at eigenvalues that contribute the most (small eigenvalues are disregarded)

**Problem:** PCA is a linear method. The "true" dimensionality can be overestimated. There can be non-linear correlations.

# PCA

- d-dimensional feature space
- d by d symmetric covariance matrix estimated from samples

$$\mathrm{Cov}(\boldsymbol{x}) = \boldsymbol{\Sigma},$$

- Select k largest eigenvalue of the covariance matrix and associated k eigenvectors
- The first eigenvector will be a direction with largest variance

# PCA via SVD

$\mathrm{Cov}(\boldsymbol{x}) = \boldsymbol{\Sigma}$ - Covariance Centered data matrix x - $\Sigma = \frac{1}{m}\sum_{i=1}^{n}(x^{(i)})(x^{(i)})^T$

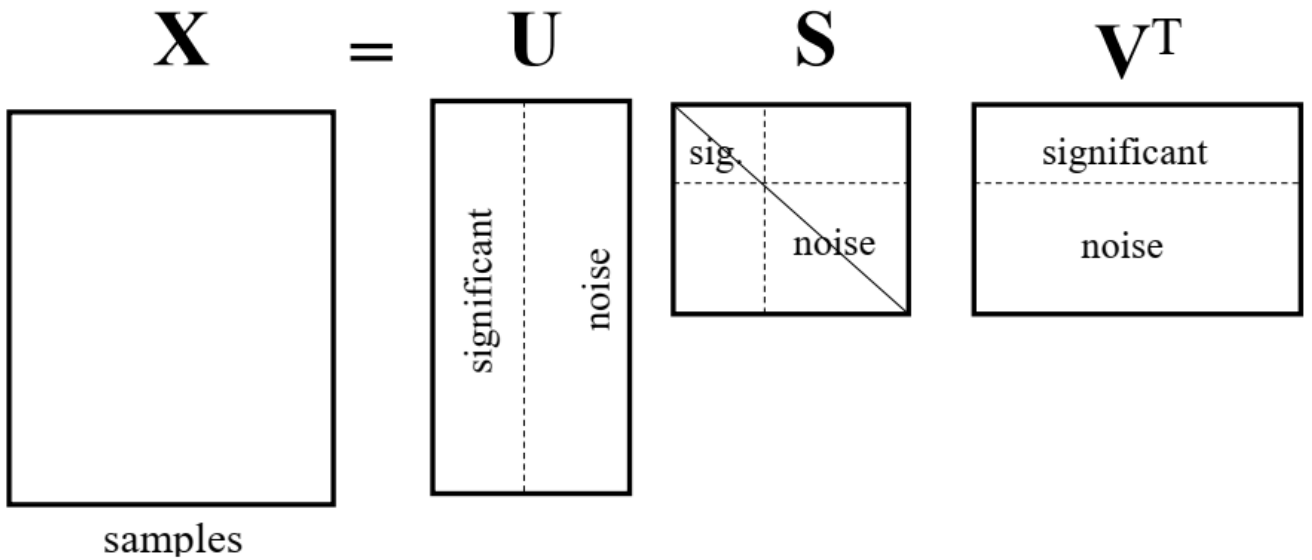
Singular Value Decomposition of the **centered** data matrix **X**.

$\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_m] \in \mathbb{R}^{N \times m}$,

$m$: number of instances,

$N$: dimension

$$\mathbf{X}_{\text{features} \times \text{samples}} = \mathbf{USV}^T$$



# PCA via SVD

**Columns of U**

- the principal vectors, $\{ \mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(k)} \}$
- orthogonal and has unit norm – so $U^T U = I$
- Can reconstruct the data using linear combinations of $\{ \mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(k)} \}$

**Matrix S**

- Diagonal
- Shows importance of each eigenvector

**Columns of $V^T$**

- The coefficients for reconstructing the samples

# PCA – פעולות מרכזיות

**PCA does the following**:

- finds orthonormal basis for data
- Sorts dimensions in order of “importance”
- Discard low significance dimensions

**Explanations**:

- Principal components – the $W_i$ vectors
- Singular values – the coefficients of the principal components
  - higher coefficients mean more important principal components
- $\lambda_i$ - eigenvalues – square of singular values

# PCA - How to choose k ?

Principal components – the $W_i$ vectors

Singular values – the coefficients of the principal components

- $\lambda_i$ - eigenvalues – square of singular values

**How do we choose k?**

**Use the following proportion:**

$$\frac{\lambda_1 + \lambda_2 + \cdots + \lambda_k}{\lambda_1 + \lambda_2 + \cdots + \lambda_k + \cdots + \lambda_d}$$

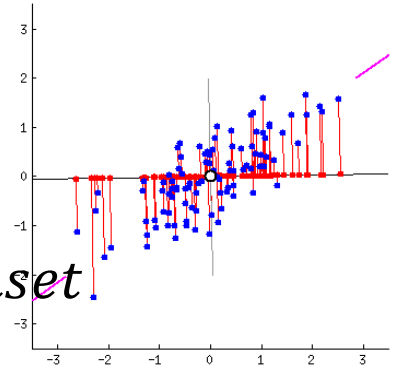
when $\lambda_i$ are sorted in descending order

- Typically, stop when proportion>0.9
- K could be also predefined

# Using PCA

Notations

- Reduced dataset – Z
- W – principal components
- $X^{scaled}$ = $standartized\ original\ dataset$




**PCA Flow**

- Calculate symmetric covariance matrix
- Calculate eigen vectors & eigen values (representing the variance) of the covariance matrix - **out of scope**
- Sort eigen vectors, by their eigen values
- Select *principal components* - the most significant eigen vectors

Transfer dataset in the following way:

- Z = $W^T * X^{scaled^T}$

# PCA - How to choose k ?

Principal components – the $W_i$ vectors

Singular values – the coefficients of the principal components

- $\lambda_i$ - eigenvalues – square of singular values

**How do we choose k?**

**Use the following proportion:**

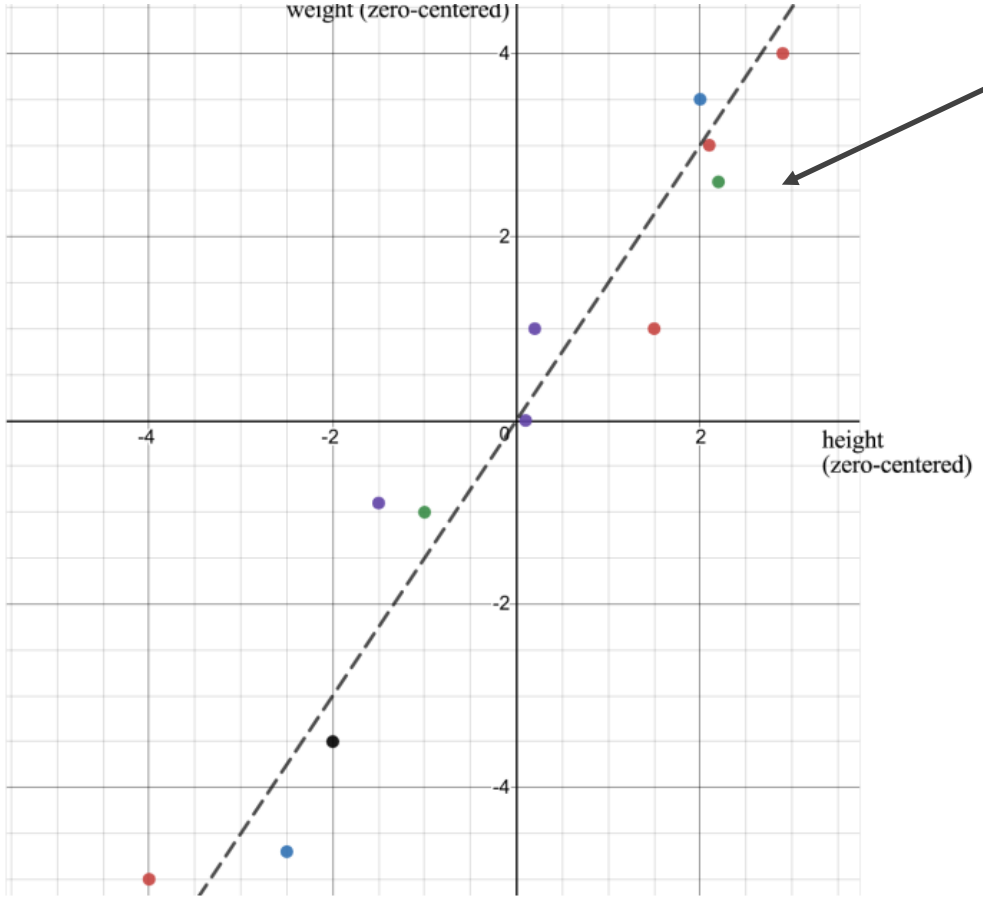
$$\frac{\lambda_1 + \lambda_2 + \cdots + \lambda_k}{\lambda_1 + \lambda_2 + \cdots + \lambda_k + \cdots + \lambda_d}$$

when $\lambda_i$ are sorted in descending order

- Typically, stop when proportion>0.9
- K could be also predefined

# PCA via SVD - example

$$A = \begin{bmatrix} 2.9 & -1.5 & 0.1 & -1.0 & 2.1 & -4.0 & -2.0 & 2.2 & 0.2 & 2.0 & 1.5 & -2.5 \\ 4.0 & -0.9 & 0.0 & -1.0 & 3.0 & -5.0 & -3.5 & 2.6 & 1.0 & 3.5 & 1.0 & -4.7 \end{bmatrix} \begin{matrix} \text{height} \\ \text{weight} \end{matrix}$$




**Data is centered (reduce mean)**

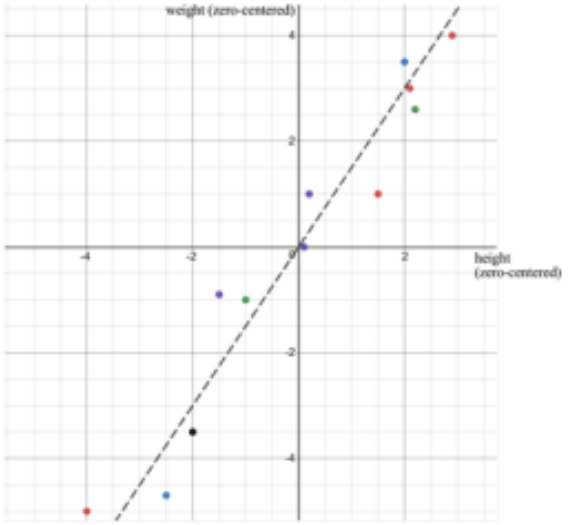
sample variance of height

$$\frac{1}{11}\begin{bmatrix} 53.46 & 73.42 \\ 73.42 & 107.16 \end{bmatrix}$$

sample covariance between height & weight

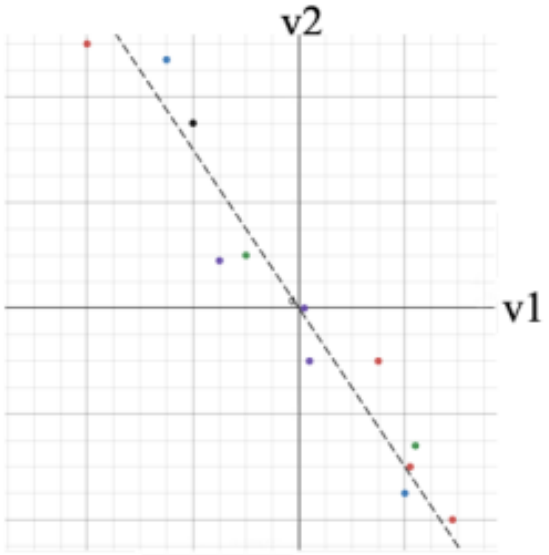
sample variance of weight

# PCA via SVD – example – meaning of correlation - reminder



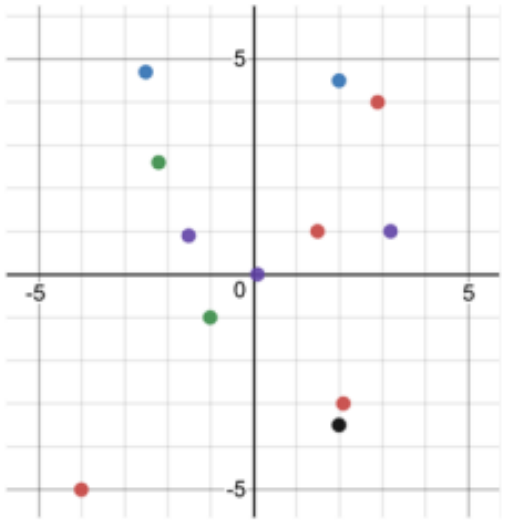

$$\frac{1}{11}\begin{bmatrix} 53.46 & 73.42 \\ 73.42 & 107.16 \end{bmatrix}$$

weight and height are positively correlated




$$\frac{1}{11}\begin{bmatrix} 53.46 & -73.42 \\ -73.42 & 107.16 \end{bmatrix}$$

variable 1 & 2 are negatively correlated

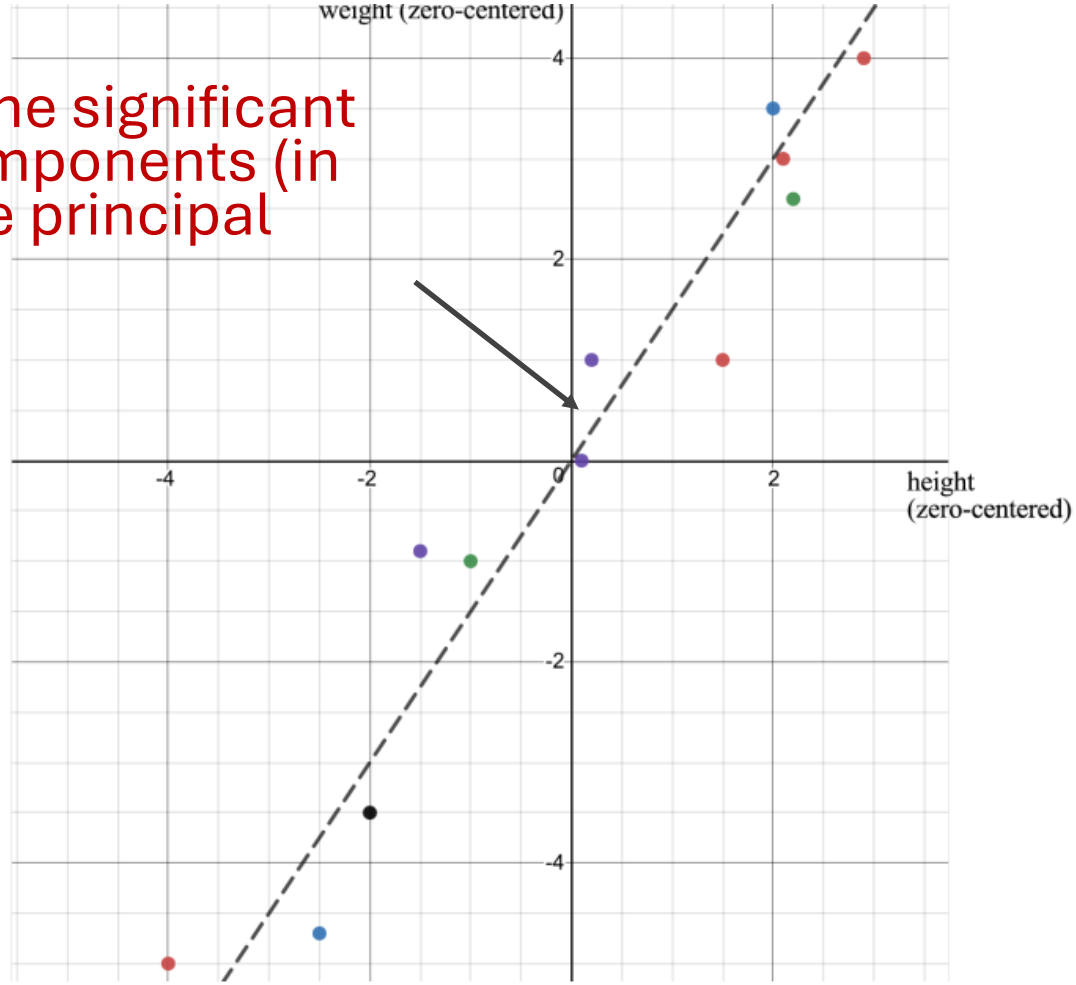



$$\frac{1}{11}\begin{bmatrix} 43.5 & 0 \\ 0 & 78.3 \end{bmatrix}$$

variable 1 & 2 are not correlated

# PCA via SVD – example – Selecting significant principal components

Select only the significant principal components (in this case one principal component)



# Preprocessing for PCA

Feature scaling:

- Features with different orders of scales prevent PCA from computing the best principal components.
  - Common scaling method – standardization

Categorical Values:

- Need to transfer to numerical values (explained next lesson & demonstrated in the tutorial notebook)

Data Cleansing before PCA:

- Missing values - PCA assumes there are no missing values (also duplicative values).
- Outliers - PCA is sensitive to outliers, need to filter outliers.
  - Large noise can become new dimension/largest PC

# PCA vs Feature selection

- Feature selection
  - Supervised: drop features which don’t introduce large errors (validation set)
  - Unsupervised: keep only uncorrelated features (drop features that don’t add much information)
- Dimensionality Reduction (Unsupervised)
  - PCA - Linearly combine feature into smaller set of features
  - PCA – Can be used to assist supervised data - explains most of the total variability

# PCA – pros and cons

**Pros**

- Dramatic reduce in size of data
  - Improve performance, reduce overfitting
- Interested in resulting uncorrelated variables which explain large portion of **total** sample variance
- Sometimes interested in explained shared variance (common factors) that affect data

**Cons**

- Assumes dependency of features – usually this is not a problem
- **PCA** is limited to linear dimensionality reduction
- Doesn’t know class labels
- PCA Does not try to explain noise
- Too expensive for some applications
- In cases of sparse data, there are better ways to deal with the dimensionality

# Until the next time ☺