From data to vectors & NLP 101
Methods in AI research

Dong Nguyen
17 Sept 2019

Utrecht University

Practicalities

Literature for today:

- Hal Daumé III, A Course in Machine Learning, 3.1-3.3 (Geometry and Nearest Neighbors; http://ciml.info/dl/v0_99/ciml-v0_99-ch03.pdf)
    - Description in text for Figure 3.4 is wrong (+ and – are switched)
- Jurafsky & Martin SLP3, 6.3 (Words and vectors) and 6.4 (Cosine for measuring similarity) https://web.stanford.edu/~jurafsky/slp3/6.pdf
- Noah A. Smith (2019), Contextual Word Representations: A Contextual Introduction: https://arxiv.org/pdf/1902.06006.pdf (sec. 5 optional).

So far

- **ML concepts:**
  - Supervised learning (how to frame your task as a supervised learning problem)
  - Inductive bias
  - Overfitting and underfitting
  - Decision boundaries
  - Evaluation of supervised learning systems (don’t touch your test data!)

- **Methods**
  - Decision Trees

Today

- **ML concepts:**
  - Vector spaces
  - Distance metrics
- **ML method:**
  - K-nearest neighbors
- **NLP 101**
  - How to represent documents as vectors
  - How to represent words as vectors

RECAP !

Supervised learning

Learn a machine learning model using **labeled example instances:**

features → target ↙

$$\{<x^{(1)}, y^{(1)}>, \ldots, <x^{(N)}, y^{(N)}>\}$$

Goal: Predict the target using the features

Need to define **features**, characteristics of the instances that the model uses for predictions (words in a document, movie ratings, etc..)

Features for house price prediction:

- Neighborhood
- Number of bedrooms
- First floor square meters
- Number of schools within 2 km
- Police Label Safe Housing
- ..

Representing instances using feature vectors

$$\left.\begin{array}{l}\text{Number of bedrooms: 2}\\ \text{Plot size (square meters): 110}\\ \text{Number of schools within 2 km: 2}\end{array}\right\} \texttt{[2, 110, 2]} \longleftarrow \textit{vector}$$

Representing instances using feature vectors

Number of bedrooms: 2
Plot size (square meters): 110
Number of schools within 2 km: 2
Police Label Safe Housing: yes

vector

```
[2, 110, 2, 1]
```

We encode binary features as 1 (yes) and 0 (no)

Representing instances using feature vectors

Number of bedrooms: 2
Plot size (square meters): 110
Number of schools within 2 km: 2
Police Label Safe Housing: yes
Neighborhood: Leidsche Rijn

vector

```
[2, 110, 2, 1, ?]
```

```
East: 0
West: 1
Leidsche Rijn: 2
Overvecht: 3
```

🤔

Representing instances using feature vectors

Number of bedrooms: 2
Plot size (square meters): 110
Number of schools within 2 km: 2
Police Label Safe Housing: yes
Neighborhood: Leidsche Rijn

vector

```
[2, 110, 2, 1, 0, 0, 1, 0]
```

```
East: 0
West: 1
Leidsche Rijn: 2
Overvecht: 3
```

🤔

One Hot Encoding

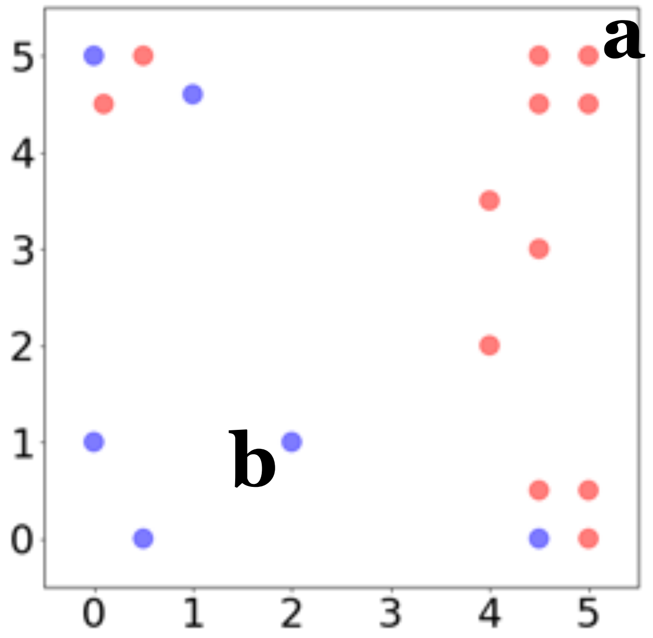
```
East? Yes = 1, No = 0
West? Yes = 1, No = 0
Leidsche Rijn? Yes = 1, No = 0
Overvecht: Yes = 1, No = 0

Neighborhood feature: [0,0,1,0]
```

😃

Types of features

- A **numerical** feature (a real number)
  - Sentence length
  - Number of likes
  - Temperature
- A **binary** feature: a yes vs. no distinction (usually 1 vs. 0)
  - Is the text capitalized?
  - Are A and B friends?
  - Employed?
  - Like Chinese restaurants?
- **Categorical** feature:
  - Country
  - Genre of a text

Vector space



```
a = [5, 5]
b = [2, 1]
```

Your instances are now represented as points in *vector space*. Each dimension represents a feature (e.g. whether the user liked a certain restaurant)

Usually *thousands of features*

Vectors & vector spaces

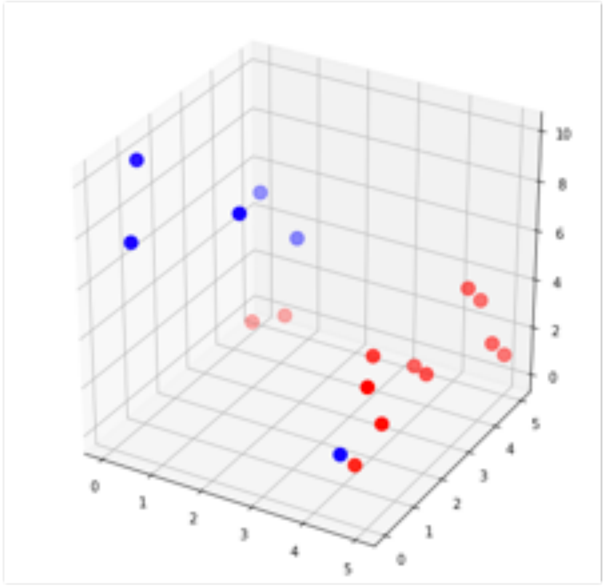
a = [5, 5]
b = [2, 1]

a is a two-dimensional vector, i.e. **a** $\in \mathbb{R}^2$

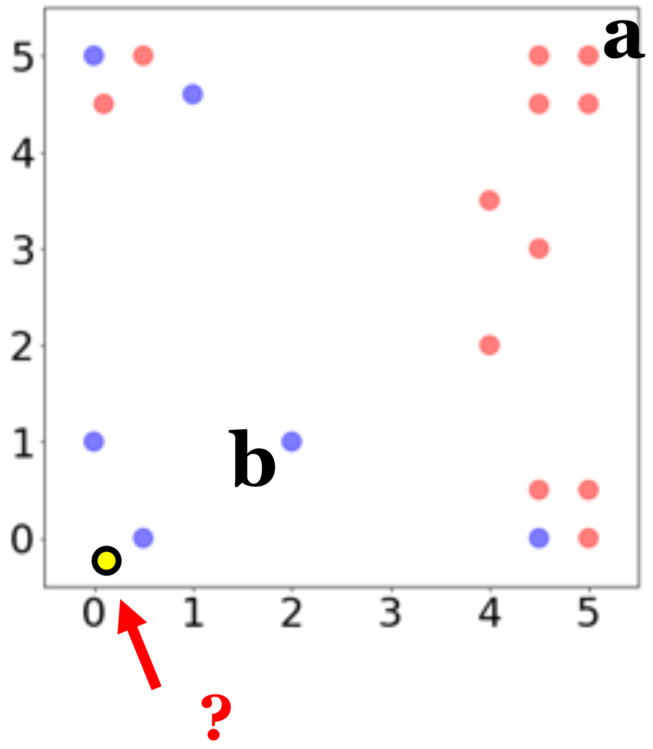
a + **b** = [5 + 2, 5 + 1] = [7, 6]

c = [c_1, ..., c_d]
c is a d-dimensional vector, i.e. **c** $\in \mathbb{R}^d$

What is a_1? *5*
What is b_2? *1*



Vector space



Q: How would you classify this point? (red or blue?)

Nearest neighbor

*This "**rule of nearest neighbor**" has considerable elementary intuitive appeal and probably corresponds to practice in many situations. For example, it is possible that much medical diagnosis is influenced by the doctor's **recollection** of the subsequent history of an earlier patient whose symptoms **resemble in some way** those of the current patient. (Fix and Hodges, 1952)*

Idea: Classify new examples based on the most similar training examples

`[CIML: chapter 3]`

Memory-based learner

This is **memory-based** learning (also called instance-based learning): look for similar instances in the training data (stored in **memory**) and fit with the local points.

Four components:

- A distance metric
- How many neighbors to look at?
- A weighting function (*optional*)
- How to fit with the local points?

Memory-based learner

This is **memory-based** learning (also called instance-based learning): look for similar instances in the training data (stored in **memory**) and fit with the local points.

Four components: `1-nearest neighbors`

- A distance metric

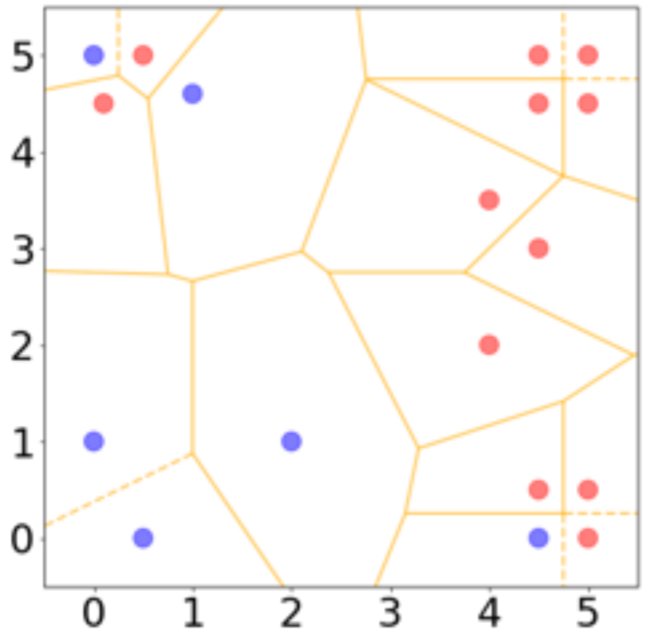
  `Many options, e.g. Euclidian`
- How many neighbors to look at?

  `1`
- A weighting function (*optional*)

  `Unused`
- How to fit with the local points?

  ```
  Just return the label
  of the most nearest point
  ```

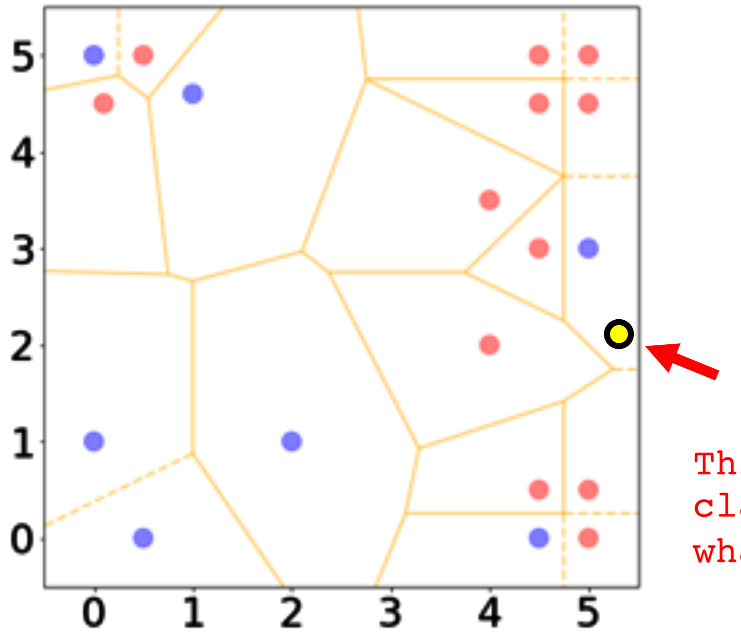
1-nearest neighbors decision boundaries



Every training example has its own neighborhood

For any point x in a training set, the Voronoi cell of x consists of all points closer to x than any other points in the training set.

1-nearest neighbors decision boundaries



1-nearest neighbors is sensitive to outliers!

Small changes in the training set can lead to large differences in the decision boundary

This point now gets classified as blue, is this what we want?

Memory-based learner

This is **memory-based** learning (also called instance-based learning): look for similar instances in the training data (stored in **memory**) and fit with the local points.

Four components: `K-nearest neighbors`

- A distance metric

    `Many options, e.g. Euclidian`

- How many neighbors to look at?

    `K`

- A weighting function (*optional*)

    `Unused`

- How to fit with the local points?

    `Just predict the majority label`

K-nearest neighbor

Training data *number of neighbors* *test instance*

```
Algorithm 3 KNN-PREDICT($\mathbf{D}$, $K$, $\hat{x}$)
S ← [ ]
for $n = 1$ to $N$ do
    $S \leftarrow S \oplus \langle d(x_n, \hat{x}), n \rangle$          // store distance to training example $n$
end for
$S \leftarrow$ SORT($S$)                                   // put lowest-distance objects first
$\hat{y} \leftarrow 0$
for $k = 1$ to $K$ do
    $\langle dist, n \rangle \leftarrow S_k$                        // $n$ this is the $k$th closest data point
    $\hat{y} \leftarrow \hat{y} + y_n$                  // vote according to the label for the $n$th training point
end for
return SIGN($\hat{y}$)                  // return $+1$ if $\hat{y} > 0$ and $-1$ if $\hat{y} < 0$
```

[CIML: chapter 3]

Choosing *K*

How many instances should we consider when making the classification?

K = 1

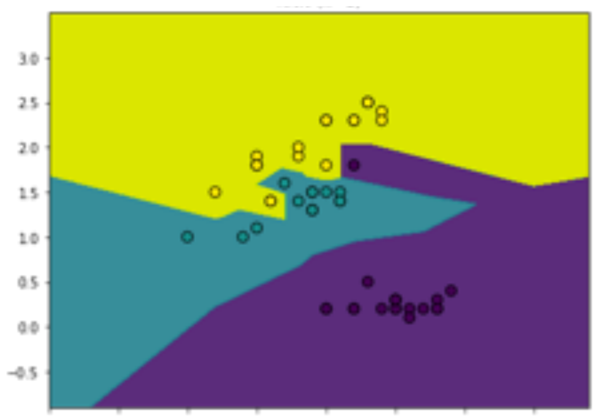
Sensitive to outliers
Could lead to overfitting

K = N (number of instances)

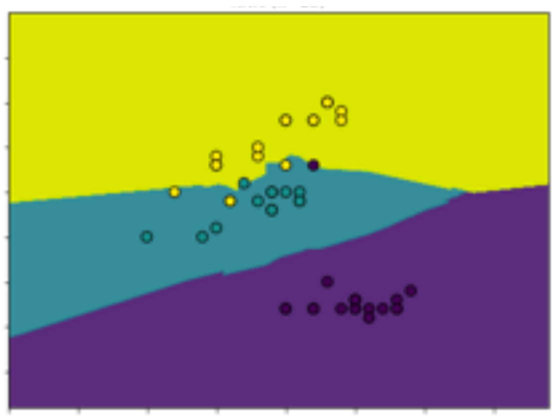
Always predict the majority label
Leads to underfitting!

Decision boundary

K=1

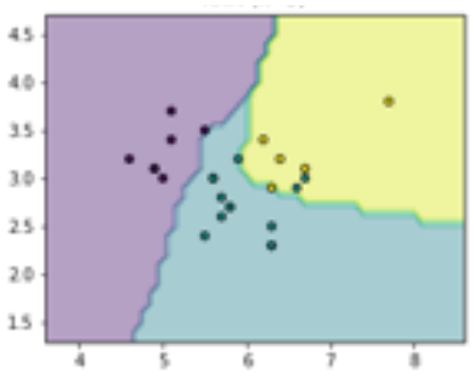


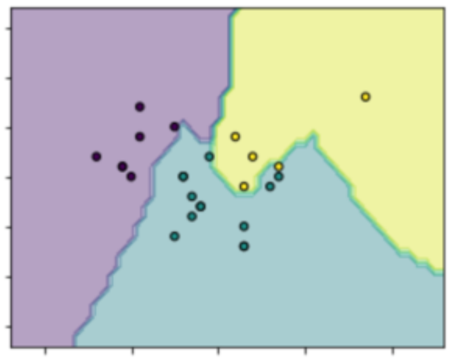
K=15

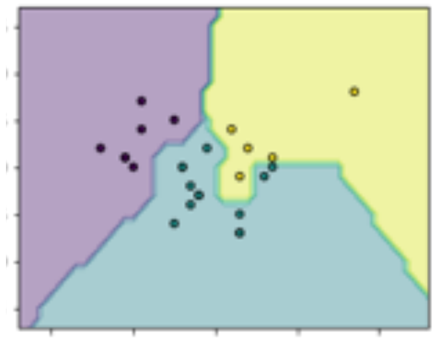


Larger K *values lead to smoother decision boundaries*

Decision boundary



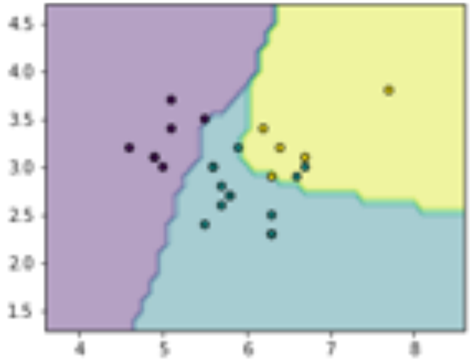


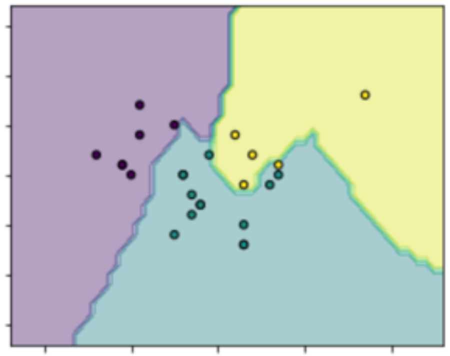


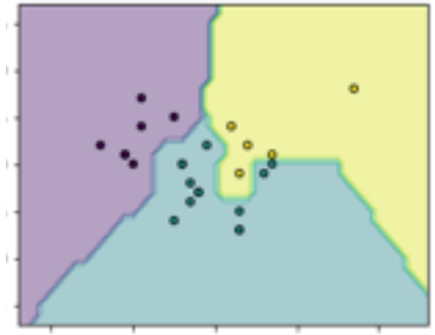
Questions:

- Which one is: K=1, K=3, K=5?
- K is usually an odd number, why?
- How would you set K?

Decision boundary







Questions:

- Which one is: K=1, K=3, K=5?
- K is usually an odd number, why?
- How would you set K?

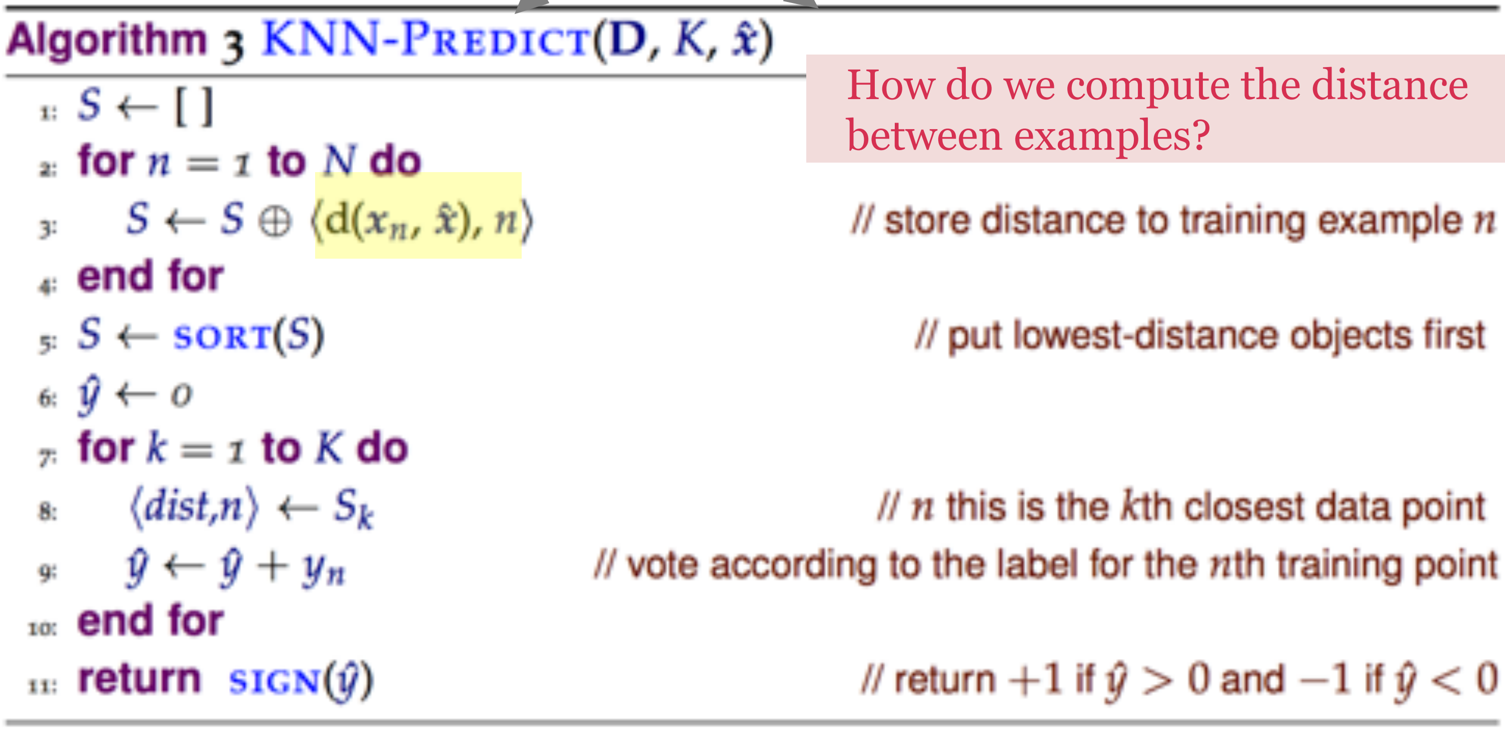
K=5, K=3, K=1

To not have ties (with binary classification)

K is a hyperparameter

K-nearest neighbor

Training data *number of neighbors* *test instance*



```
Algorithm 3 KNN-Predict(D, K, $\hat{x}$)
1:  $S \leftarrow [\,]$
2:  for $n = 1$ to $N$ do
3:      $S \leftarrow S \oplus \langle d(x_n, \hat{x}), n \rangle$        // store distance to training example $n$
4:  end for
5:  $S \leftarrow$ sort($S$)                                   // put lowest-distance objects first
6:  $\hat{y} \leftarrow 0$
7:  for $k = 1$ to $K$ do
8:      $\langle dist, n \rangle \leftarrow S_k$                   // $n$ this is the $k$th closest data point
9:      $\hat{y} \leftarrow \hat{y} + y_n$          // vote according to the label for the $n$th training point
10: end for
11: return sign($\hat{y}$)                     // return $+1$ if $\hat{y} > 0$ and $-1$ if $\hat{y} < 0$
```

How do we compute the distance between examples?

[CIML: chapter 3]

Distance measure

By representing data points as vectors, we can measure their distance (or similarity) in the vector space.

If a and b are scalars, the most straightforward way to define distance is:

$|a - b|$ *(e.g., |3-5| = 2)*

Let's generalize this idea to vectors

Length (or, norm) of a vector

Length of vector `[3, 4]`:

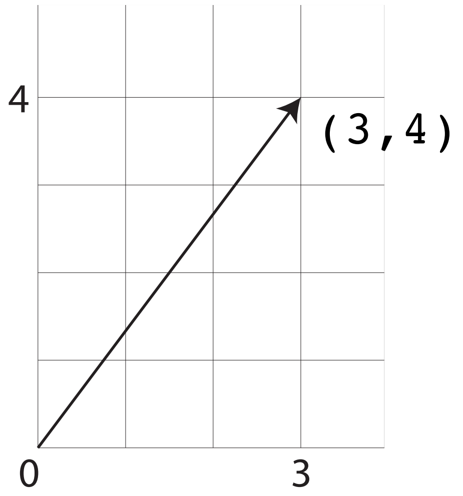
$$\sqrt{x^2 + y^2} = \sqrt{4^2 + 3^2}$$

Length of vector `[3, 4, 1]`:

$$\sqrt{4^2 + 3^2 + 1^2}$$

Length of vector **a** :

$$\|\boldsymbol{a}\|_2 = \sqrt{\sum a_i^2}$$



This is also called the l_2- norm or the Euclidian norm

Euclidian distance

Distance between endpoint of vectors

Question:
What is the Euclidian distance between a & b

```
a = [0, 3, 5]
b = [3, 1, 2]
```

`Euclidian distance:`

$$\|\boldsymbol{a} - \boldsymbol{b}\|_2 = \sqrt{\sum (a_i - b_i)^2}$$

Euclidian distance

Distance between endpoint of vectors

```
a-b     = [(0-3), (3-1), (5-2)]
        = [-3, 2, 3]
```

Euclidian distance:

$$\|\boldsymbol{a}-\boldsymbol{b}\|_{\boxed{2}} = \sqrt{\sum (a_i - b_i)^2}$$

also called L2 distance

$= \text{sqrt}((-3)^2 + 2^2 + 3^2)$

$= \text{sqrt}(9 + 4 + 9)$

$= 4{,}690$

Question:
What is the Euclidian distance between a & b

```
a = [0, 3, 5]
b = [3, 1, 2]
```

Manhattan distance

Distance between endpoint of vectors

```
a-b     = [(0-3), (3-1), (5-2)]
        = [-3, 2, 3]
```

```
Manhattan distance:
```

$$\|\boldsymbol{a} - \boldsymbol{b}\|_1 = \sum |a_i - b_i|$$

also called L1 distance

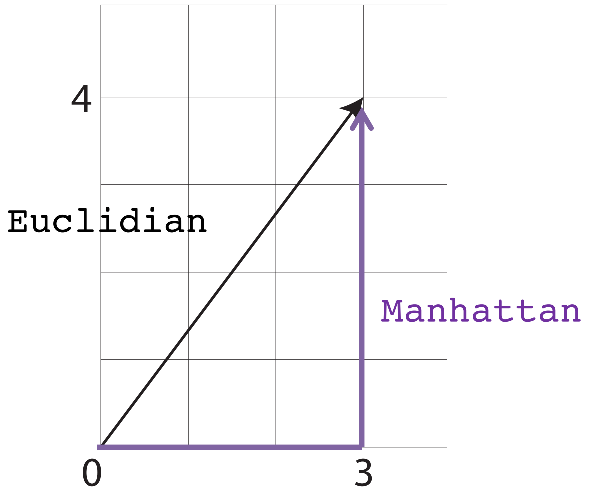
$$= |-3| + |2| + |3|$$

$$= 8$$

Question:
What is the Manhattan distance between a & b

```
a = [0, 3, 5]
b = [3, 1, 2]
```

Comparison distances

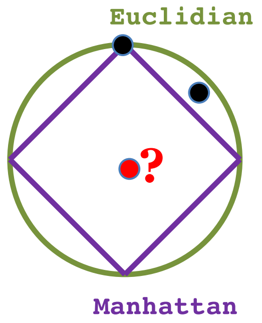


Minkowski distance:
Generalization of the Euclidian and Manhattan distance

$$\sqrt[p]{\sum |a_i - b_i|^p}$$

Many options!

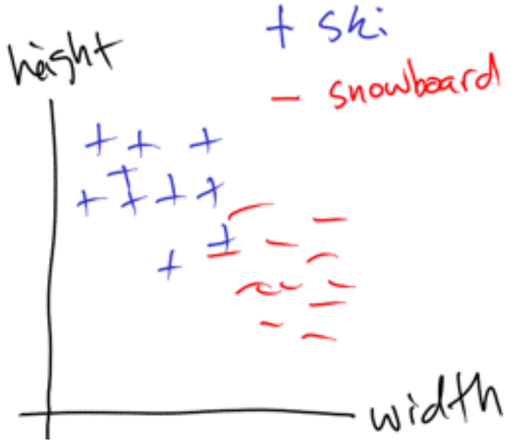
Different distance measures → different neighborhoods

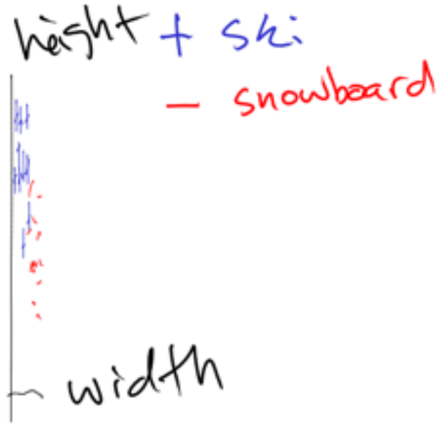


Inductive bias of k-nearest neighbors

- The label of a data point should be similar to labels of nearby points
- All features are equally important (but weighted variants exist!)

Feature Scaling





[CIML: 3.6 and 3.7]

Centering

the ith training instance

$$x_j^{(i)} = x_j^{(i)} - \mu_j$$

the jth feature

$$\text{with } \mu_j = \frac{1}{N}\sum_i x_j^{(i)}$$

```
N training examples
1 ≤ i ≤ N
d features
1 ≤ j ≤ d
```

Variance Scaling

$$x_j^{(i)} = \frac{x_j^{(i)}}{\sigma_d}$$

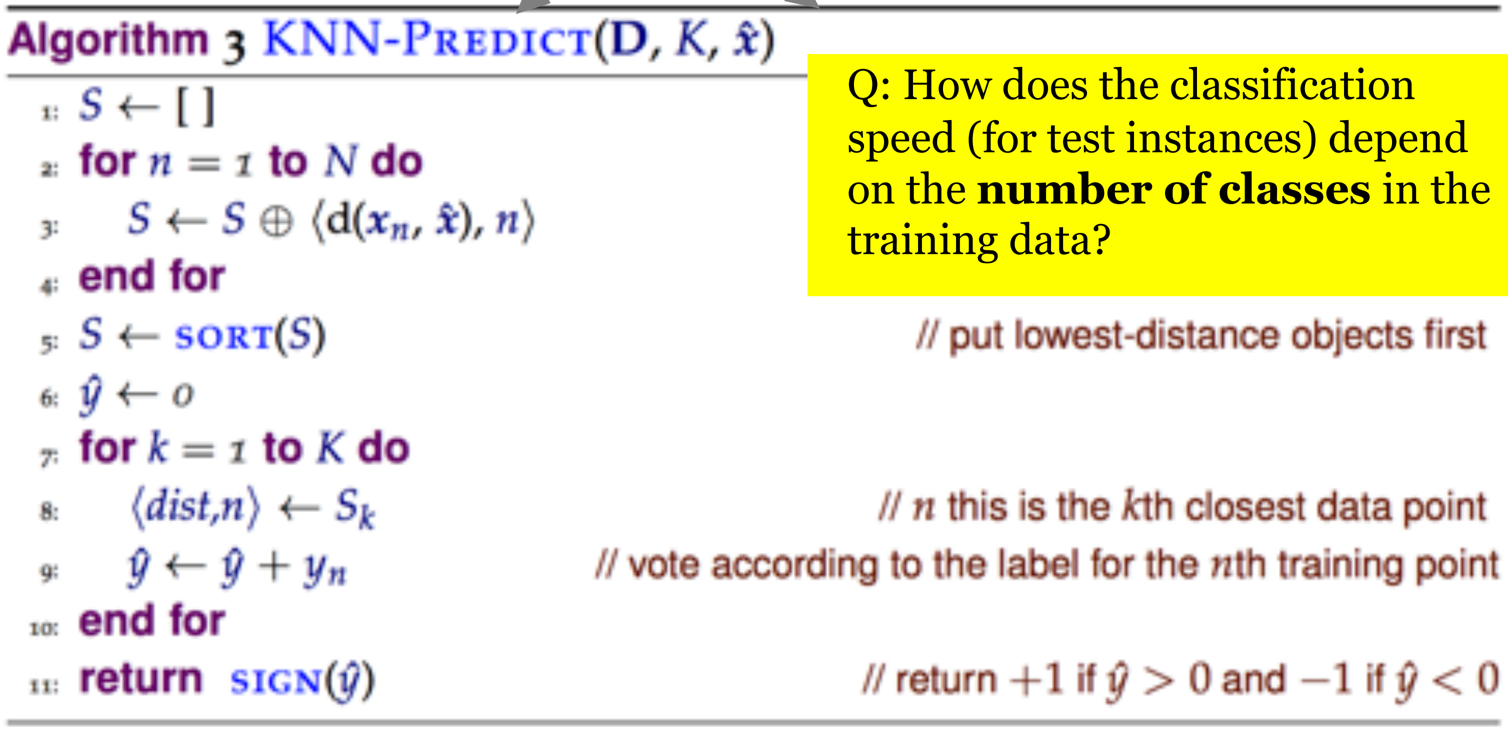
Practical considerations

- How can we use K-Nearest Neighbors for **regression?**
  - Just predict the mean of the neighbors

- What about **ties?**
  - Random
  - Use 1 -nearest neighbor to decide
  - Use the class that is most frequent in the training data (highest prior probability)

Computational Considerations

Training data *number of neighbors* *test instance*



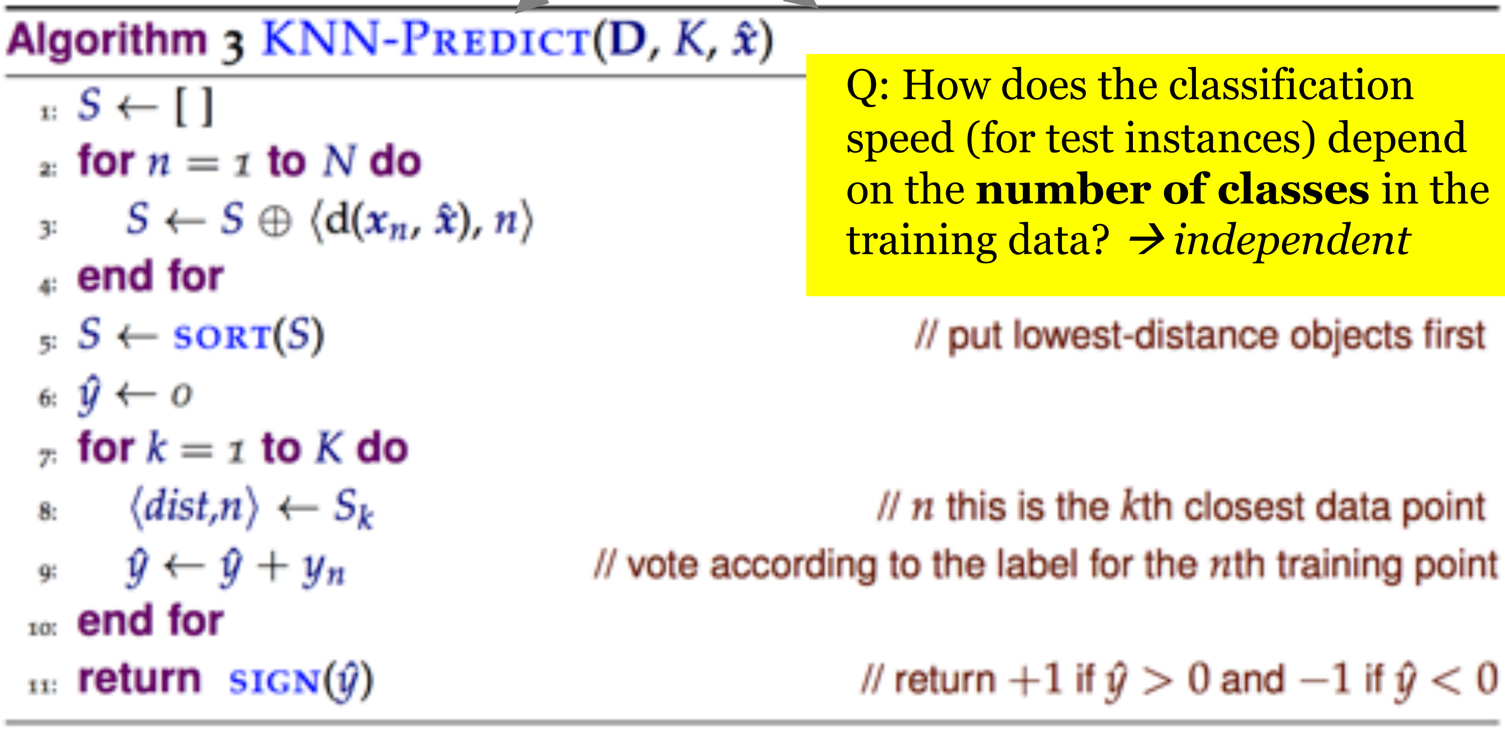
```
Algorithm 3 KNN-PREDICT($\mathbf{D}$, $K$, $\hat{x}$)
 1: $S \leftarrow [\,]$
 2: for $n = 1$ to $N$ do
 3:     $S \leftarrow S \oplus \langle \mathrm{d}(x_n, \hat{x}), n \rangle$
 4: end for
 5: $S \leftarrow \textsc{sort}(S)$                        // put lowest-distance objects first
 6: $\hat{y} \leftarrow 0$
 7: for $k = 1$ to $K$ do
 8:     $\langle dist, n \rangle \leftarrow S_k$            // $n$ this is the $k$th closest data point
 9:     $\hat{y} \leftarrow \hat{y} + y_n$                  // vote according to the label for the $n$th training point
10: end for
11: return $\textsc{sign}(\hat{y})$                     // return $+1$ if $\hat{y} > 0$ and $-1$ if $\hat{y} < 0$
```

Q: How does the classification speed (for test instances) depend on the **number of classes** in the training data?

[CIML: chapter 3]

Computational Considerations

Training data *number of neighbors* *test instance*



Algorithm 3 KNN-PREDICT($\mathbf{D}$, K, $\hat{x}$)

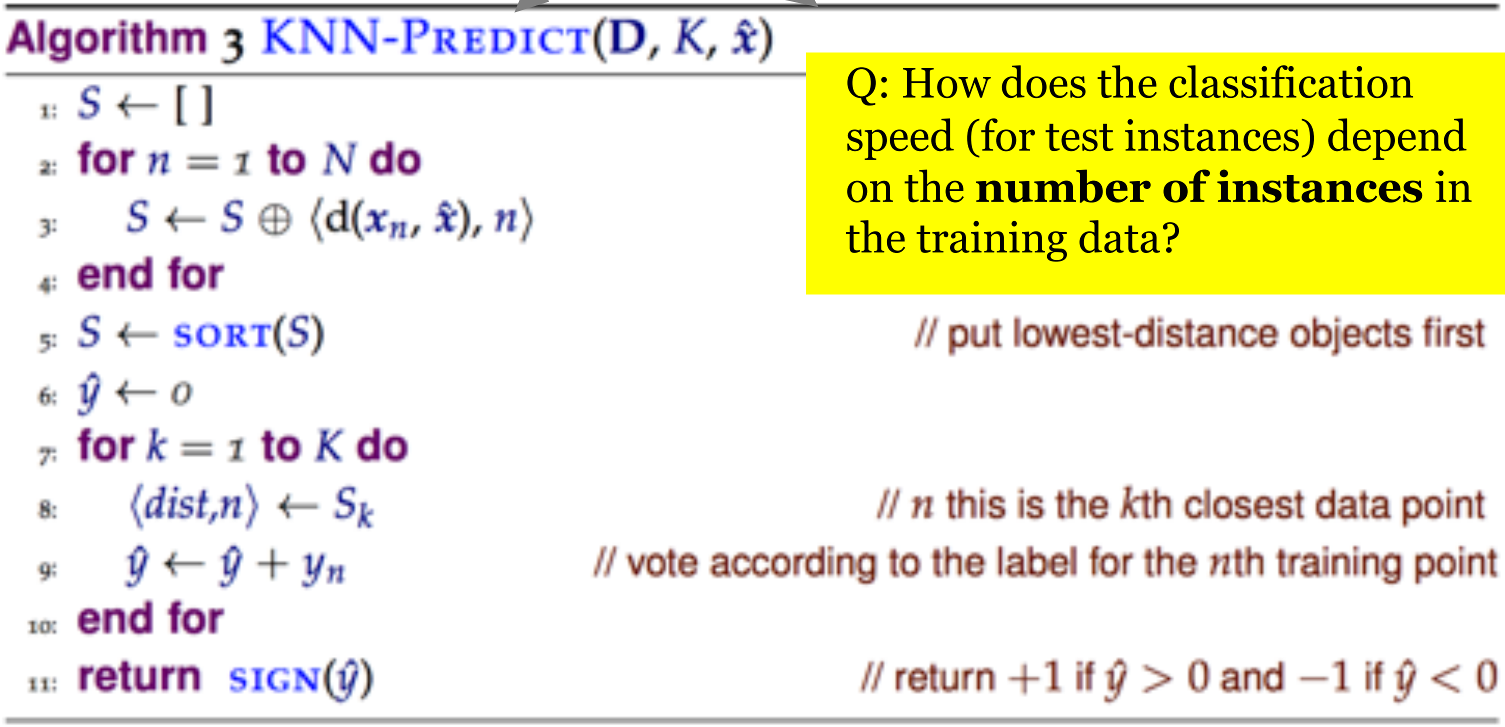
```
$S \leftarrow [\,]$
for $n = 1$ to $N$ do
    $S \leftarrow S \oplus \langle d(x_n, \hat{x}), n \rangle$
end for
$S \leftarrow$ SORT($S$)                                   // put lowest-distance objects first
$\hat{y} \leftarrow 0$
for $k = 1$ to $K$ do
    $\langle dist, n \rangle \leftarrow S_k$                          // $n$ this is the $k$th closest data point
    $\hat{y} \leftarrow \hat{y} + y_n$                    // vote according to the label for the $n$th training point
end for
return SIGN($\hat{y}$)                      // return $+1$ if $\hat{y} > 0$ and $-1$ if $\hat{y} < 0$
```

Q: How does the classification speed (for test instances) depend on the **number of classes** in the training data? → *independent*

[CIML: chapter 3]

Computational Considerations

Training data *number of neighbors* *test instance*



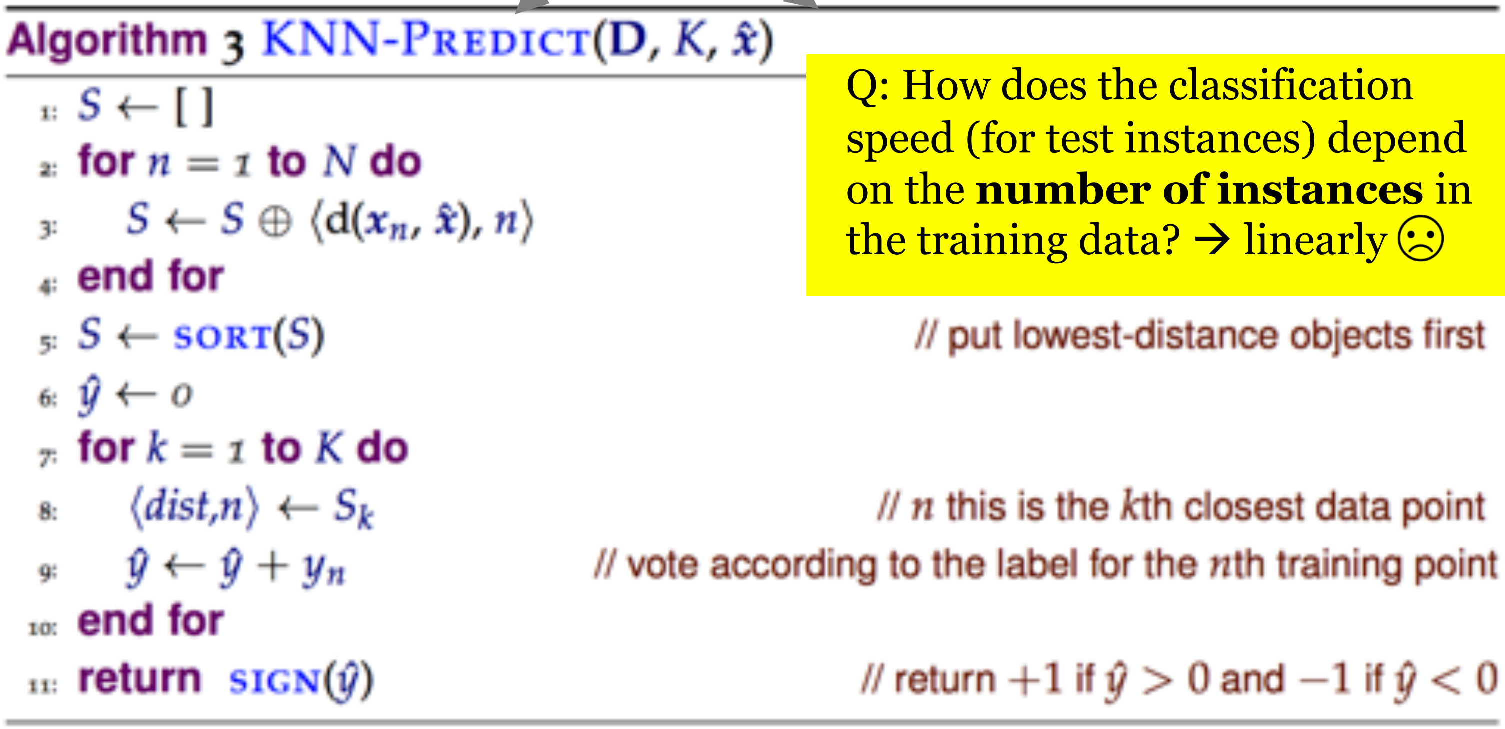
```
Algorithm 3 KNN-PREDICT($\mathbf{D}$, $K$, $\hat{x}$)
S ← [ ]
for $n = 1$ to $N$ do
    $S \leftarrow S \oplus \langle d(x_n, \hat{x}), n \rangle$
end for
$S \leftarrow$ SORT($S$)                              // put lowest-distance objects first
$\hat{y} \leftarrow 0$
for $k = 1$ to $K$ do
    $\langle dist, n \rangle \leftarrow S_k$          // $n$ this is the $k$th closest data point
    $\hat{y} \leftarrow \hat{y} + y_n$                // vote according to the label for the $n$th training point
end for
return SIGN($\hat{y}$)                                // return $+1$ if $\hat{y} > 0$ and $-1$ if $\hat{y} < 0$
```

Q: How does the classification speed (for test instances) depend on the **number of instances** in the training data?

[CIML: chapter 3]

Computational Considerations

Training data *number of neighbors* *test instance*



```
Algorithm 3 KNN-PREDICT($\mathbf{D}, K, \hat{x}$)
1: $S \leftarrow [\,]$
2: for $n = 1$ to $N$ do
3:     $S \leftarrow S \oplus \langle \mathrm{d}(x_n, \hat{x}), n \rangle$
4: end for
5: $S \leftarrow$ SORT($S$)                          // put lowest-distance objects first
6: $\hat{y} \leftarrow 0$
7: for $k = 1$ to $K$ do
8:     $\langle dist, n \rangle \leftarrow S_k$             // $n$ this is the $k$th closest data point
9:     $\hat{y} \leftarrow \hat{y} + y_n$          // vote according to the label for the $n$th training point
10: end for
11: return SIGN($\hat{y}$)             // return $+1$ if $\hat{y} > 0$ and $-1$ if $\hat{y} < 0$
```

Q: How does the classification speed (for test instances) depend on the **number of instances** in the training data? → linearly ☹

[CIML: chapter 3]

Computational Considerations

- Training is fast
- It’s easy to add new training data
  (no ‘retraining’ needed)

- Making predictions is slow
  - Takes N comparisons (number of instances) × d (number of dimensions) operations
- Need to *store* the training data

There are techniques to speed up K-nearest neighbors, such as kd-trees

Decision trees vs. nearest neighbors

Decision Trees

- Learn a model from the training data
- Apply model to new data

- Features are selected

- Decision boundaries are axis-aligned cuts

Nearest neighbors

- Store data
- Compare new data to stored data

- All features have equal weight → Sensitive to irrelevant features

- Decision boundaries can be complex

RECAP !

Chatbots Architectures

- **Rule-based**
  - Pattern-action rules (ELIZA) + a mental model (Parry)

- **Corpus-based (using a large chat corpus)**
  - Information Retrieval (IR) based. **Select** a response from a large corpus.
  - Neural networks. Train a neural network to generate text.

RECAP !

Chatbots, response selection

Return the response to the most similar turn

- Take user's turn **q** and find a similar turn **t** in the corpus **C**

  q = *"do you like Doctor Who?"*

  t= *"you like Doctor Who?"*

- Return the response to **t** in your corpus.

$$r = response\ (\underset{t \in C}{argmax}\ \text{similarity}(\mathbf{q}, \mathbf{t}))$$

1-NN approach!

Up next: How can we represent words and documents as vectors?

Natural Language Processing 101

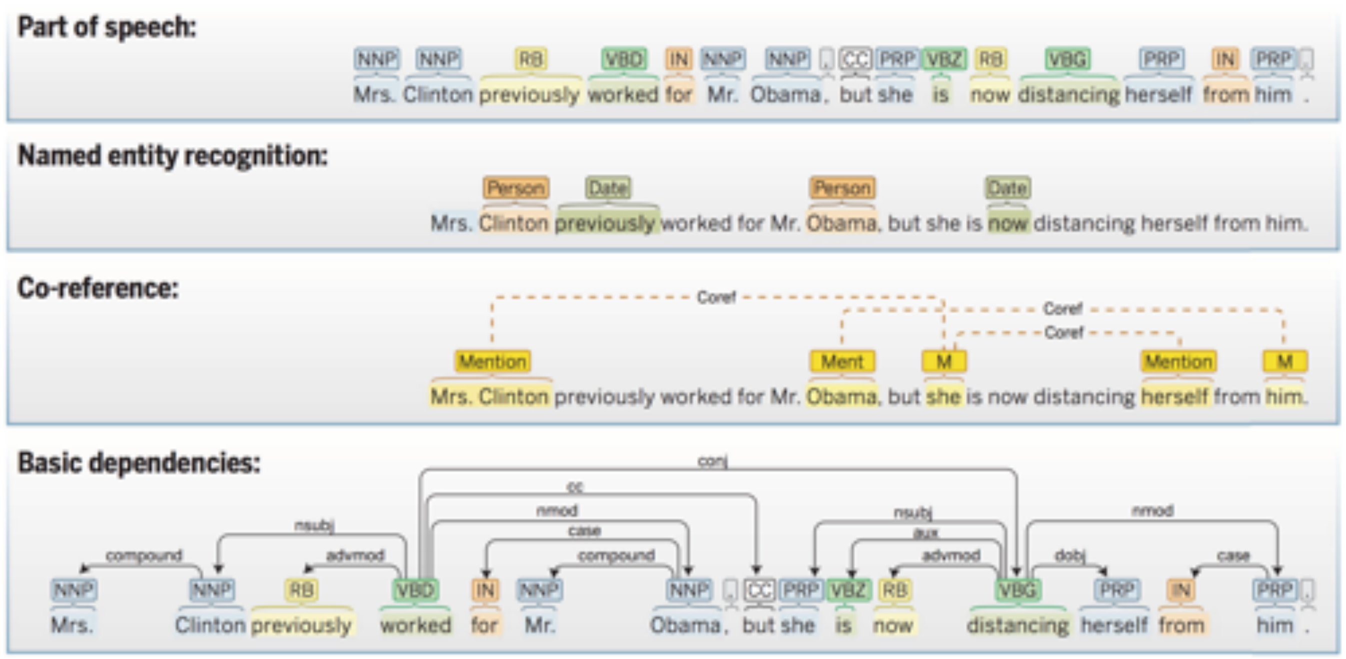
Natural Language Processing (NLP)

Automatic processing and analysis of
natural language

Dutch, English, Spanish, Hindi, Frisian, Turkish, ...

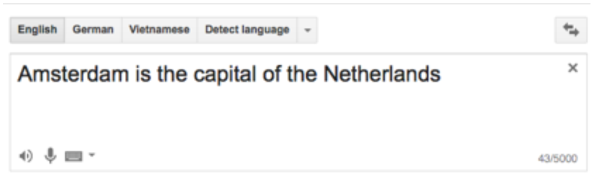
7,111 known living languages.
https://www.ethnologue.com/

Analysis of linguistic structure

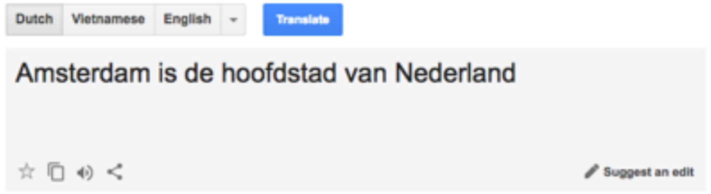

Part of speech:
NNP NNP RB VBD IN NNP NNP , CC PRP VBZ RB VBG PRP IN PRP .
Mrs. Clinton previously worked for Mr. Obama , but she is now distancing herself from him .
Named entity recognition:
Person Date Person Date
Mrs. Clinton previously worked for Mr. Obama, but she is now distancing herself from him.
Co-reference:
Coref
Coref
Coref
Mention Ment M Mention M
Mrs. Clinton previously worked for Mr. Obama, but she is now distancing herself from him.
Basic dependencies:
conj
cc
nmod
case
compound
nsubj
advmod
compound
nsubj
aux
advmod
nmod
dobj
case
NNP NNP RB VBD IN NNP NNP , CC PRP VBZ RB VBG PRP IN PRP .
Mrs. Clinton previously worked for Mr. Obama , but she is now distancing herself from him .

Advances in natural language processing,
Hirschberg and Manning, Science 2015

machine translation



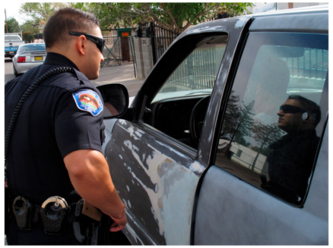
Amsterdam is the capital of the Netherlands



Amsterdam is de hoofdstad van Nederland

IBM Watson (question answering)

https://www.youtube.com/watch?v=WFR3lOm_xhE



uncovering racial disparities in analyzing language from police body camera's (PNAS 2017)

What makes language understanding hard? Polysemy

Today, I went to the **bank** to deposit a check.

The hut is located near the **bank** of the river.

Ardougne North **Bank** is near the **bank** of the River Dougne in the north part of the city.

Words can have multiple meanings

What makes language understanding hard? Syntactic ambiguity

“One morning, I shot an elephant in my pajamas.”

“How he got in my pajamas, I don't know.”

(Groucho Marx)

Who wore the pajamas?

What makes language understanding hard?

i know right	shake my head			for	your		
ikr	smh	he	asked	fir	yo	last	name
inter-jection	acronym	pronoun	verb	preposition	determiner	adjective	noun

				you		facebook	
so	he	can	add	u	on	fb	lololol
preposition	pronoun	verb	verb	pronoun	preposition	proper noun	inter-jection

http://www.cs.cmu.edu/~ark/TweetNLP/

What makes language understanding hard?

A string may have many possible interpretations in different contexts, and resolving ambiguity correctly may rely on knowing a lot about the world. (Noah Smith)

- Linguistic **diversity**: languages, dialects, registers, styles
- Language is constantly **changing**

ML for NLP: What assumptions should the machine learning model have? (‘*How does language work?*’)
How should we represent language data in our models?

Today: Vector representations of words and documents

Represent *documents*, *words*, phrases, sentences, etc.. as vectors

📄 → `[0.20, 0.80, 0.10, ..., -0.10]`

dog → `[0.10, 0.90, -0.20, ..., -0.40]`

- What are the dimensions of the vector space?

`One dimension for each word.`

- How do we measure distances in the vector space?

To find similar documents, find similar words, as input to ML models, …

Tokenization

```
A tokenizer segments text into a
sequence of tokens
```

```
A
tokenizer
segments
text
into
a
sequence
of
tokens
```

We often take words as the basic level of analysis

Tokenization: tokens vs types

```
A tokenizer segments text into a
sequence of tokens
```

Token:	**Type:**
A	a
tokenizer	tokenizer
segments	segments
text	text
into	into
a	sequence
sequence	of
of	tokens
tokens	

Tokenization challenges

始めまして。お元気ですか。

Japanese: No spaces between words

New York, European Union

Multiword expressions

I'm, don't

Contractions

Note: Many libraries provide tokenization methods (e.g. nltk, spacy, scikit-learn)Ω

Pre-processing steps

Stop word removal
- *a, an, the, it, ..*
- Using a stop word list or by filtering words that appear in many documents

Lemmatization
- *sang, sung, sings* → *sing*

Stemming (strip endings of the word)
- E.g., *running* to *run*

Lowercasing
- E.g., *Running* to *running*

Removing infrequent words
- E.g., occurring in less than 10 (or 25, or …) documents

Overall goal: reduce the number of *types* (usually thousands to millions in a large dataset!)

But could remove useful signals!

Example: function words (the, a, and, however, on, etc..) are not so important for topic classification, but strong signals for authorship identification.

Edit distance

The minimum **edit distance** between two strings:

the minimum number of editing operations (insertion, deletion, substitution) needed to transform one string into another

```
I N T E * N T I O N
| | | | | | | | | |
* E X E C U T I O N
d s s   i s
```

```
d = deletion
s = substitution
i = insertion
```

J&M: Fig. 2.14

Levenshtein distance:
each of these operators has cost 1

Bag of words representation

Only look at the presence (or frequency) of words, i.e. ignore the order. (*Often a surprisingly hard baseline to beat!*)

```
this is a
nice restaurant
```

this
restaurant
a
nice
is

But..

```
Dog bites man
```

```
Man bites dog
```

Question: Come up with a task for which bag of words is probably sufficient. And another task for which it is not.

```
Sufficient: topic classification
Not sufficient: machine translation
```

Jaccard similarity

Jaccard similarity

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

```
this is not bad

this is really bad
```

```
Common (3): this, is, bad
Union (5):  this, is, not, bad, really

Jaccard similarity: 3/5
```

Simple 😃

Infrequent words (e.g. *bad*) versus frequent words (e.g. *is*) 😔

Frequency and order of words are ignored

Vector representations of documents

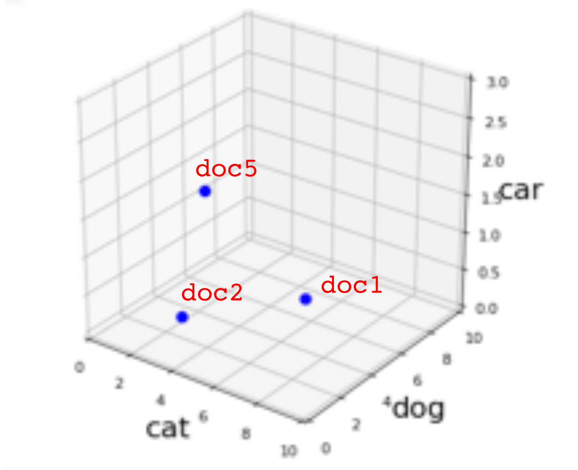
word-document matrix

$$\begin{array}{c|ccccccc}
 & doc_1 & doc_2 & doc_3 & doc_4 & doc_5 & doc_6 & doc_7 \\
\hline
cat & 5 & 2 & 0 & 1 & 4 & 0 & 0 \\
dog & 7 & 3 & 1 & 0 & 2 & 0 & 0 \\
car & 0 & 0 & 1 & 3 & 2 & 1 & 1
\end{array}$$

vector

Euclidian distance

$$\sqrt{\sum (a_i - b_i)^2}$$



doc$_1$ and doc$_2$
Jaccard similarity = 1
Euclidian distance = 5

doc$_1$ and doc$_5$
Jaccard similarity = 2/3
Euclidian distance = 5.477

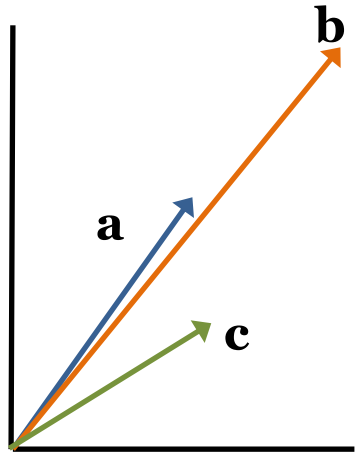
RECAP !

Euclidian distance

Distance between endpoint of vectors

Euclidean distance:

$$\|\boldsymbol{a} - \boldsymbol{b}\|_2 = \sqrt{\sum (a_i - b_i)^2}$$



a is closer (more similar) to **c** than to **b**, is this what we want?

Extreme example:

Concatenate two documents.

Dot product

Dot product:

$$\boldsymbol{a} \bullet \boldsymbol{b} = \sum a_i b_i$$

```
a = [0, 3, 5]
b = [3, 1, 2]
```

a•b$= 0 \times 3 + 3 \times 1 + 5 \times 2 = 13$

```
c = [0, 1, 0]
d = [1, 1, 1]
```

c•d$= 0 \times 1 + 1 \times 1 + 0 \times 1 = 1$

If we have a **set-based** representation, i.e. ***a*** and ***b*** are **binary** vectors (1 = word is present, 0=absence), then the dot product is the number of words present in both documents (intersection)

Compare to Jaccard Similarity! (normalization by dividing by the union of words)

Dot product

Dot product:

$$\boldsymbol{a} \bullet \boldsymbol{b} = \sum a_i b_i$$

Length of a vector:

RECAP !

$$\|\boldsymbol{a}\|_2 = \sqrt{\sum a_i^2}$$

Therefore:

$$\|\boldsymbol{a}\|_2 = \sqrt{\boldsymbol{a} \bullet \boldsymbol{a}}$$

Normalization of a vector to unit length

$$\frac{\boldsymbol{a}}{\|\boldsymbol{a}\|_2}$$

```
a = [0, 3, 5]
```

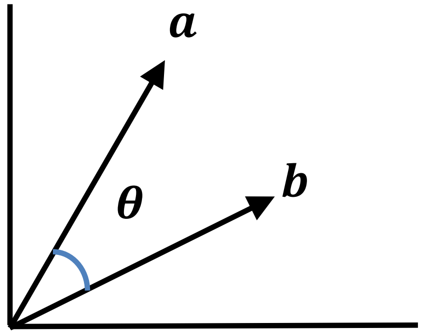
$\|\boldsymbol{a}\|_2$ `= sqrt(9 + 25) = 5.831`

$\frac{\boldsymbol{a}}{\|\boldsymbol{a}\|_2}$ `= [0, 0.5145, 0.8575]`

Cosine similarity

Cosine similarity

$$\cos(\boldsymbol{\theta}) = \frac{\boldsymbol{a} \bullet \boldsymbol{b}}{\|\boldsymbol{a}\|\|\boldsymbol{b}\|} = \frac{\sum \boldsymbol{a}_i \boldsymbol{b}_i}{\sqrt{\sum \boldsymbol{a}_i^2}\sqrt{\sum \boldsymbol{b}_i^2}}$$



Question:
What is the cosine distance between a & b

```
a = [0, 3, 5]
b = [3, 1, 2]
```

$\boldsymbol{a} \bullet \boldsymbol{b}$ = 13

$\|\boldsymbol{a}\|$ =sqrt(0^2 + 3^2 + 5^2) = sqrt(34)

$\|\boldsymbol{b}\|$ =sqrt(3^2 + 1^2 + 2^2) = sqrt(14)

→ 13/(sqrt(34) * sqrt(14)) ≈ 0.60

Cosine similarity

Cosine similarity

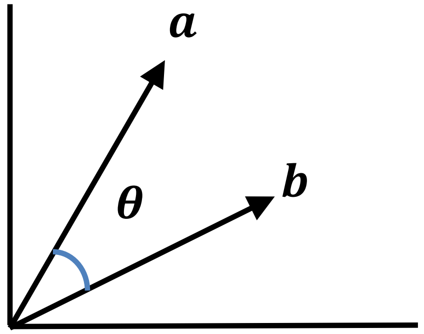
$$\cos(\boldsymbol{\theta}) = \frac{\boldsymbol{a} \bullet \boldsymbol{b}}{\|\boldsymbol{a}\|\|\boldsymbol{b}\|} = \frac{\sum \boldsymbol{a_i}\,\boldsymbol{b_i}}{\sqrt{\sum \boldsymbol{a_i^2}}\,\sqrt{\sum \boldsymbol{b_i^2}}}$$

When $\boldsymbol{a}$ and $\boldsymbol{b}$ are normalized:

$$\cos(\boldsymbol{\theta}) = \boldsymbol{a} \bullet \boldsymbol{b}$$

If the vectors are orthogonal:

$$\boldsymbol{a} \bullet \boldsymbol{b} = 0$$



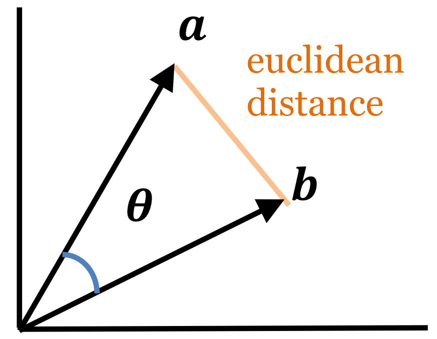
Cosine ranges from -1 (vectors pointing in opposite directions) to 0 (orthogonal) to 1 (vectors pointing in the same direction).

Raw frequencies (non-negative): 0-1

Comparisons

Cosine similarity

$$\cos(\boldsymbol{\theta}) = \frac{\boldsymbol{a} \bullet \boldsymbol{b}}{\|\boldsymbol{a}\|\|\boldsymbol{b}\|} = \frac{\sum \boldsymbol{a}_i \boldsymbol{b}_i}{\sqrt{\sum \boldsymbol{a}_i^2}\sqrt{\sum \boldsymbol{b}_i^2}}$$



Euclidean distance

$$\|\boldsymbol{a} - \boldsymbol{b}\|_2 = \sqrt{\sum (a_i - b_i)^2}$$

If $\boldsymbol{a}$ and $\boldsymbol{b}$ have unit length, then Euclidean distance and cosine similarity will result in the same ordering (but reversed)

Today: Vector representations of words and documents

Represent *documents*, *words*, phrases, sentences, etc.. as vectors

📄 → `[0.20, 0.80, 0.10, ..., -0.10]`

dog → `[0.10, 0.90, -0.20, ..., -0.40]`

Document representations

- What are the dimensions of the vector space?

One dimension for each word. Values: binary (presence or absence), frequency, weighting schemes (e.g. tf-idf, not discussed)

- How do we measure distances in the vector space?

Cosine similarity

Today: Vector representations of words and documents

Represent *documents*, *words*, phrases, sentences, etc.. as vectors

📄 → `[0.20, 0.80, 0.10, ..., -0.10]`

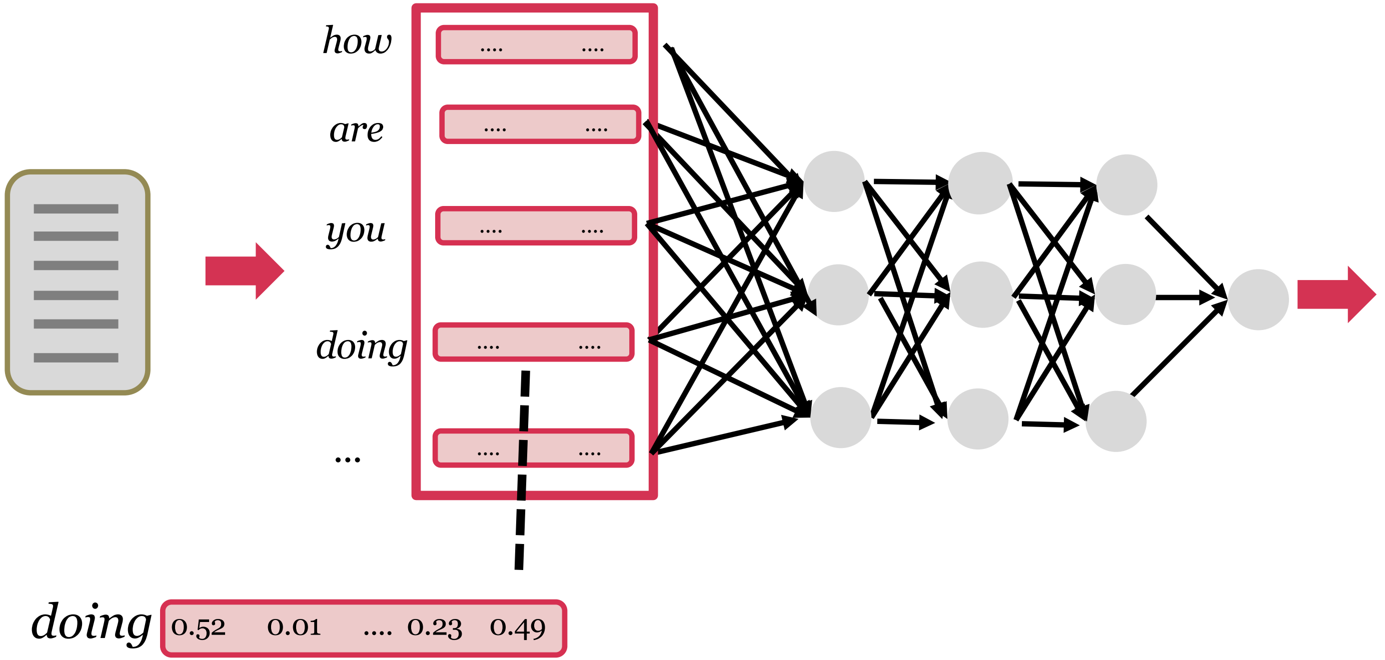
dog → `[0.10, 0.90, -0.20, ..., -0.40]`

- What are the dimensions of the vector space?
- How do we measure distances in the vector space?

```
dog and puppy,
Monday and Tuesday,
buy and purchase
```

Word overlap is not enough!
Can we also map
words to vectors??

Word representations



Spam classification

Topic classification

Translation

Sentiment

Political affiliation

…

The standard way to represent the meaning of words in (neural) NLP

Word representations

when your	**cat**	leaves a mangled mouse
to care for your	**cat**	or kitten including food
the domestic	**cat**	is a small, typically furry
choosing to get a	**cat**	or dog is a

You shall know a word by the company it keeps
(Firth, J. R. 1957:11)

To assign similar vectors to similar words a notion of similarity is needed.

The distributional hypothesis: Words that occur in similar contexts tend to have similar meanings.

Vector representations of words

documents as context

word-document matrix

$$\begin{array}{c|ccccccc|l}
 & doc_1 & doc_2 & doc_3 & doc_4 & doc_5 & doc_6 & doc_7 & \\
\text{cat} & 5 & 2 & 0 & 1 & 4 & 0 & 0 & \}\ \text{vector} \\
\text{dog} & 7 & 3 & 1 & 0 & 2 & 0 & 0 & \\
\text{car} & 0 & 0 & 1 & 3 & 2 & 1 & 1 &
\end{array}$$

Vector representations of words

documents as context
word-document matrix

	doc$_1$	doc$_2$	doc$_3$	doc$_4$	doc$_5$	doc$_6$	doc$_7$
cat	5	2	0	1	4	0	0
dog	7	3	1	0	2	0	0
car	0	0	1	3	2	1	1

neighboring words as context
word-word matrix

	cat	dog	car	bike	book	house	tree
cat	0	3	1	1	1	2	3
dog	3	0	2	1	1	3	1
car	1	2	0	3	2	2	0

Vector representations of words

Properties:

- Vectors are sparse: Many zero entries
- Values are based on frequencies (sometimes weighted)

documents as context

word-document matrix

	doc_1	doc_2	doc_3	doc_4	doc_5	doc_6	doc_7
cat	5	2	0	1	4	0	0
dog	7	3	1	0	2	0	0
car	0	0	1	3	2	1	1

neighboring words as context

word-word matrix

	cat	dog	car	bike	book	house	tree
cat	0	3	1	1	1	2	3
dog	3	0	2	1	1	3	1
car	1	2	0	3	2	2	0

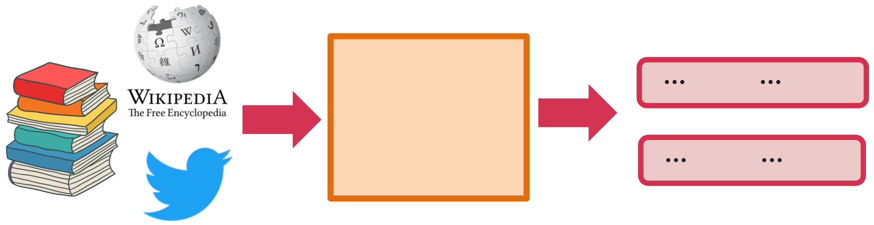
Word embeddings (Dense word vectors)

Word embeddings:

- Vectors are dense
- The values are learned by **training a classifier** to distinguish nearby and far away words
- Individual dimensions are less interpretable.

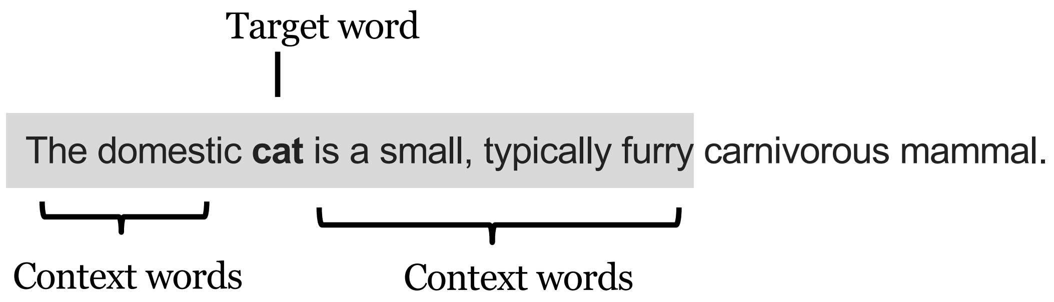
Dense real-valued vectors

cat	0.52	0.84	0.01	....	0.23
dog	0.40	0.90	0.10	....	0.40



How are word embeddings learned?

(the skipgram model)


Target word
The domestic **cat** is a small, typically furry carnivorous mammal.
Context words
Context words

Embedding vectors are essentially a by-product!

target word	context word	label
cat	small	1
cat	furry	1
cat	car	0
…	…	….

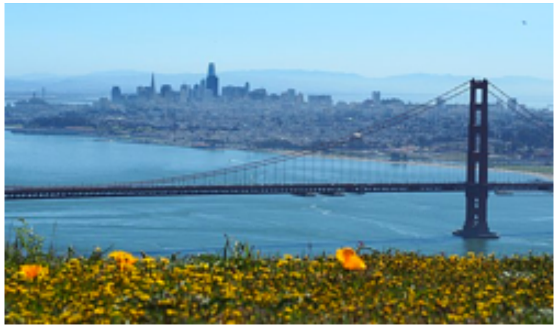
Different methods based on similar ideas:

- Word2vec (Mikolov et al. 2013), skipgram & continuous bag of words. (https://code.google.com/archive/p/word2vec/, another implementation in https://radimrehurek.com/gensim/)
- GloVe (https://nlp.stanford.edu/projects/glove/)
- FastText (https://fasttext.cc/)

Properties of word embeddings

We can use cosine similarity to find similar words in the vector space.

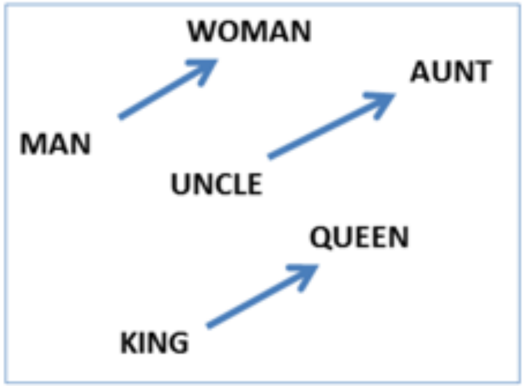
```
san francisco

los_angeles       0.666175
golden_gate       0.571522
oakland           0.557521
california        0.554623
```

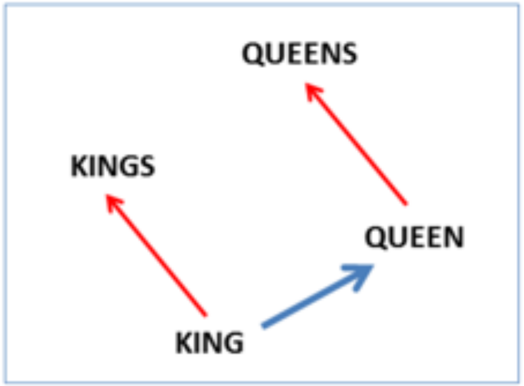


https://code.google.com/archive/p/word2vec/
https://en.wikipedia.org/wiki/San_Francisco

Properties of word embeddings

We can look at analogies in the vector space





king - man + woman ≈ queen

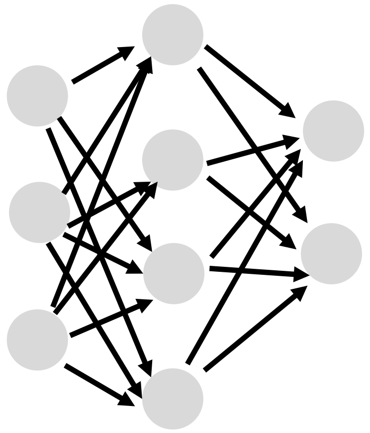
Linguistic Regularities in Continuous Space Word Representations, Mikolov et al. 2013

Word embeddings (Dense word vectors)

Dense real-valued vectors

cat	0.52	0.84	0.01	....	0.23
dog	0.40	0.90	0.10	....	0.40

How are these word representations used?



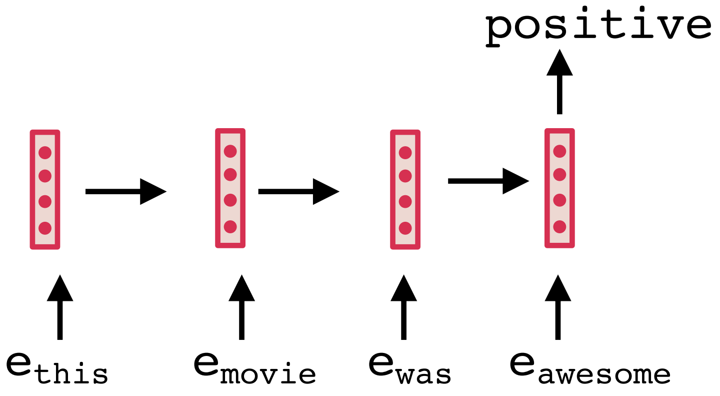
In neural networks (text classification, sequence tagging, etc..)

As research objects

Using embeddings in ML models

$e_{\texttt{this}}$	0.52	-0.20	-0.10	...	0.68
$e_{\texttt{movie}}$	0.90	0.91	0.30	...	-0.40
$e_{\texttt{was}}$	0.22	-0.12	-0.19	...	0.58
$e_{\texttt{awesome}}$	-0.12	-0.75	-0.8	...	0.28

Bag of words model (surprisingly effective). Average over the vectors! (resulting in a vector of size 7)



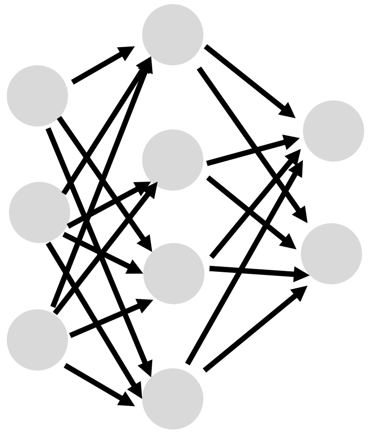
Most commonly the word vectors are used as input in **neural networks**

Word embeddings (Dense word vectors)

Dense real-valued vectors

cat	0.52	0.84	0.01	....	0.23
dog	0.40	0.90	0.10	....	0.40

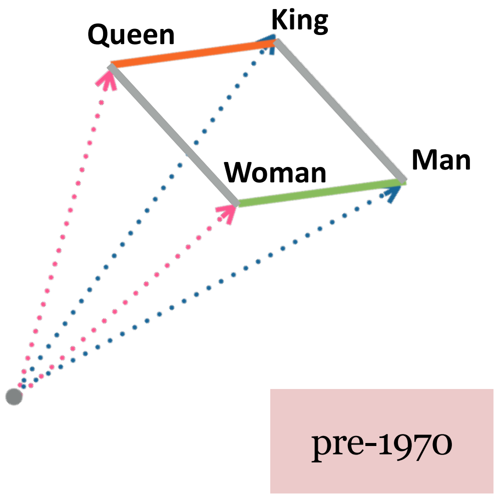
How are these word representations used?

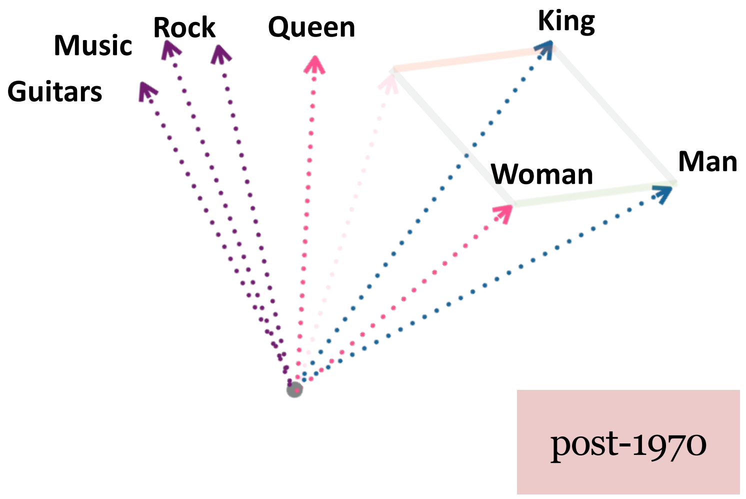


In neural networks (text classification, sequence tagging, etc..)

As research objects

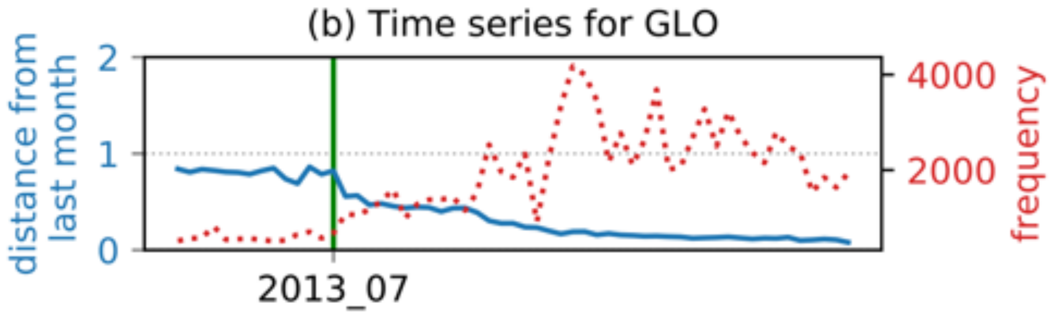
Measuring change using embeddings





Empirical data: glo

August 2013 rapper Chief Keef released "Gotta Glo Up One Day."



[Shoemark, Liza, et al. EMNLP 2019]

What do you need to know

- K-nearest neighbor method (algorithm, pros and cons, effect of K, distance measures)
- You should be able to compute Manhattan/L1 and Euclidian/L2 distance, cosine similarity, dot product
- The frequently used preprocessing steps in natural language processing
- What makes processing and analyzing language difficult?
- Representing documents (from documents to vectors)
- Representing words (from words to vectors)

NLP tools

- **Natural Language Toolkit (NLTK)** http://www.nltk.org/

python™
Natural Language Analyses with NLTK

- **Spacy** https://spacy.io/

- Many machine learning libraries (e.g. scikit-learn) also implement basic NLP methods.

spaCy

Books (drafts online!)

- Speech & language processing (3rd edition, to appear in 2019) by Jurafsky and Martin
  - https://web.stanford.edu/~jurafsky/slp3/
- Introduction to Natural Language Processing (1st edition, to appear in 2019) by Jacob Eisenstein
  - https://github.com/jacobeisenstein/gt-nlp-class/blob/master/notes/eisenstein-nlp-notes.pdf

Thanks

Some slides based on (or inspired by) slides by Matt Gormley, Carlos Guestrin, Soheil Feizi, Victor Lavrenko, Marine Carpuat, Noah Smith, Marijn Schraagen