How to choose w to maximize $\frac{1}{n}\sum_{i=1}^{n}(w^T x^{(i)})^2$?

(we restrict w to have $\|w\| = 1$)

$$\frac{1}{n}\sum_{i=1}^{n}\underbrace{(w^T x^{(i)})}_{1\times 1}\underbrace{(x^{(i)T} w)}_{1\times 1}$$

$$= \frac{1}{n}\sum_{i=1}^{n}\underbrace{w^T}_{1\times d}\underbrace{(x^{(i)} x^{(i)T})}_{d\times d}\underbrace{w}_{d\times 1}$$

$$= \frac{1}{n} w^T\left(\sum_{i=1}^{n} x^{(i)} x^{(i)T}\right) w$$

$$X X^T = \begin{bmatrix} x_1^2 & x_1x_2 & \cdots \\ x_1x_2 & \ddots & \\ \vdots & & x_d^2 \end{bmatrix}$$

↑ symmetric

Covariance of data $\{x^{(1)}, \ldots, x^{(n)}\} = \frac{1}{n}\sum_{i=1}^{n} x^{(i)} x^{(i)T}$

because mean of $x^{(i)}$'s is 0

$$= w^T \Sigma w$$

↑ covariance matrix, symmetric

Every symmetric matrix can be written as

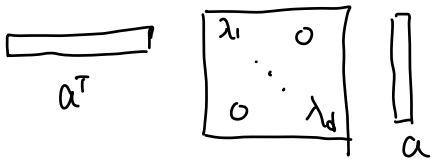
$$\Sigma = U D U^T \quad \text{where} \quad D = \begin{bmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_d \end{bmatrix} \text{ diagonal}$$

U is orthonormal $\begin{bmatrix} | & & | \\ u_1 & \cdots & u_d \\ | & & | \end{bmatrix}$ each u_i is unit vector and $u_i^T u_j = 0$ for $i \neq j$ (they are orthogonal)

$$\underset{w}{\text{maximize}}\ w^T \Sigma w = \underbrace{w^T U}_{a^T} D \underbrace{U^T w}_{a}$$

define $a = U^T w$ — a is still unit vector since w is unit vector, we just changed the basis

same as $\underset{a}{\text{maximize}}\ a^T D a$



$$= \sum_{j=1}^{d} \lambda_j a_j^2$$

constraint: $\sum_{j=1}^{d} a_j^2 = 1$

assume $\lambda_1 \geq \lambda_2 \cdots \geq \lambda_d$

optimal solution: $a_1 = 1$, other entries of $a = 0$

$$a = U^T w = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$\Rightarrow \quad w = u_1$

Recap: Given $\{x^{(1)}, \ldots, x^{(n)}\}$

① mean-center data

② compute $\Sigma = \frac{1}{n} \sum_{i=1}^{n} x^{(i)} x^{(i)T}$

③ decompose Σ as UDU^T

④ choose w to be eigenvector corresponding to largest eigenvalue

What if dimension > 1?

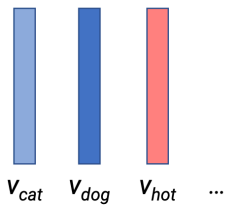
e.g. $x^{(i)} \in \mathbb{R}^{1000}$

Want to visualize the data, so reduce it to 2 dimensions.

solution: choose eigenvalues for 2 largest eigenvalues

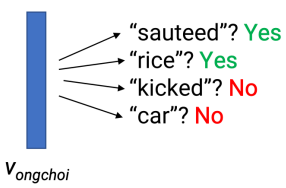
Lexical Semantics

- Goal: For each word *w*, have vector v_w that represents word's meaning
  - Lexical = word-level
  - Semantics = meaning
- What do we want to represent?
  - Synonymy (*car*/*automobile*) or antonymy (*cold*/*hot*)
  - Hypernymy/Hyponymy (*animal*/*dog*)
  - Similarity (*cat*/*dog*, *coffee*/*cup*, *waiter*/*menu*)
  - Various features
    - Sentiment (positive/negative)
    - Formality
    - All sorts of properties (Is a city? Is an action that a person can do?)



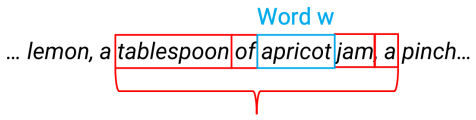
Word vectors as a learning problem

- Want to learn vector v_w for each word *w*
- What makes a vector good?
- Idea: v_w should help you predict which words co-occur with w
  - Captures **distribution** of context words for *w*
  - Think of it as N binary classification problems, where N is size of vocabulary



Creating a dataset

- Given: Raw dataset of text (unsupervised)
- We will create N "fake" supervised learning problems!
  - We don't really care about these supervised learning problems
  - We just care that we learn good vectors
- Task i: Did word *w* co-occur with the i-th word?
  - Positive examples: Real co-occurrences within sliding window
  - Negative examples: Random samples



Word *w* ("input")	Context *w'* ("task")	y (label)
apricot	tablespoon	+1
apricot	of	+1
apricot	jam	+1
apricot	a	+1