The Perceptron

LIN 313 Language and Computers
UT Austin Fall 2025
Instructor: Gabriella Chronis

Admin

- HW 2 grades posted this afternoon
- HW 3 due 10/15

Overview 10/8

- Review homework
- Artificial Neurons
  - components
    - weights
    - bias
    - activation
- The Perceptron Algorithm
  - vector notation
  - dot product
  - use the algorithm to predict outputs from inputs

Problem 2: Bayesian Spelling Correction

Part 2: the bigram model

Error model probabilities

- P(korekt | correct) = .034
- P(korekt | kraken) = .007
- P(korekt | carrot) = .015

Language model probabilities

candidates

- P(correct | great) = 0/1013 = 0
- P(kraken | great) = 5/1013 = 0.004935
- P(carrot | great) = 1/1013 = 0.000987

$$\boxed{P(a|b) = \frac{P(b|a) \times P(a)}{P(b)}}$$ BAYES LAW

conditional independence assumption: $P(a|b,c) = P(a|b) \times P(a|c)$

substituting our events: $P(\text{candidate}|\text{korekt}, \text{great}) = P(\underset{a}{\text{candidate}}|\underset{b}{\text{korekt}}) \times P(\text{cand}|\text{great})$

apply Bayes Law

$$P(\text{cand}|\text{korekt}, \text{great}) = \frac{P(\text{cand}|\text{great}) \times P(\text{korekt}|\text{candidate}) \times P(\text{candidate})}{P(\text{korekt})}$$

ignore the denominator

$$\boxed{P(\text{cand}|\text{korekt}, \text{great}) = P(\text{cand}|\text{great}) \times P(\text{korekt}|\text{cand}) \times P(\text{cand})}$$

↳ apply this formula to each candidate (see next page

Problem 2: Bayesian Spelling Correction

Part 2: the bigram model

Error model probabilities

- P(korekt | correct) = .034
- P(korekt | kraken) = .007
- P(korekt | carrot) = .015

Language model probabilities

- P(correct | great) = 0/1013 = 0
- P(kraken | great) = 5/1013 = 0.004935
- P(carrot | great) = 1/1013 = 0.000987

Derive the formula we are going to use:

1. conditional independence assumption
   a. P(a | b, c) = P (a | b) x P(a | c)
2. Substitute our events
   a. P(candidate | korekt , great) = P(candidate | great) x P(candidate | korekt)
3. We already have P(candidate | great).
4. We need P(candidate | korekt). Use Bayes Law
   a. P(candidate | korekt, great) = P(candidate | great) x P(korekt | candidate) x P(candidate) / P(korekt)
5. We will be comparing these probabilities, so we can ignore the denominator because it's the same for all
   a. P(candidate | korekt, great) = P(candidate | great) x P(korekt | candidate) x P(candidate)

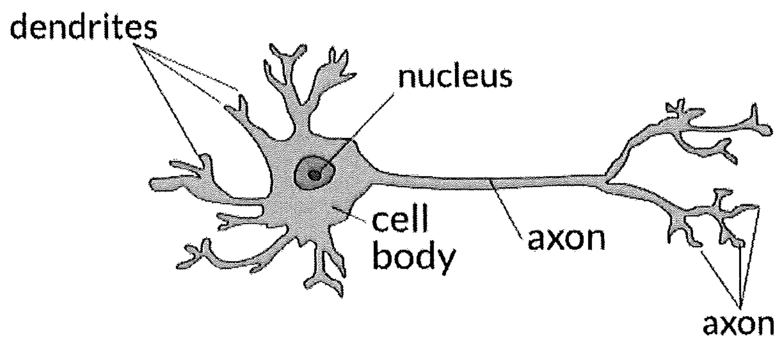
Apply the formula to each candidate:

1. correct
   a. P(correct | korekt, great) ∝ P(correct | great) x P(korekt | correct) x P(correct)
   b. P(correct | korect, great) ∝ 0 x 0.034 x 0.000845
   c. P(correct | korect, great) ∝ 0
2. kraken
   a. P(kraken | korekt, great) ∝ P(kraken | great) x P(korekt | kraken) x P(kraken)
   b. P(kraken | korekt, great) ∝ 0.004935 x 0.007 x 0.000264
   c. P(kraken | korekt, great) ∝ 9.11988e-9
3. carrot
   a. P(carrot | korekt, great) ∝ P(carrot | great) x P(korekt | carrot) x P(carrot)
   b. P(carrot | korekt, great) ∝ 0.000987 x 0.015 x 0.000211
   c. P(carrot | korekt, great) ∝ 3.123855e-9

What is a neural network?

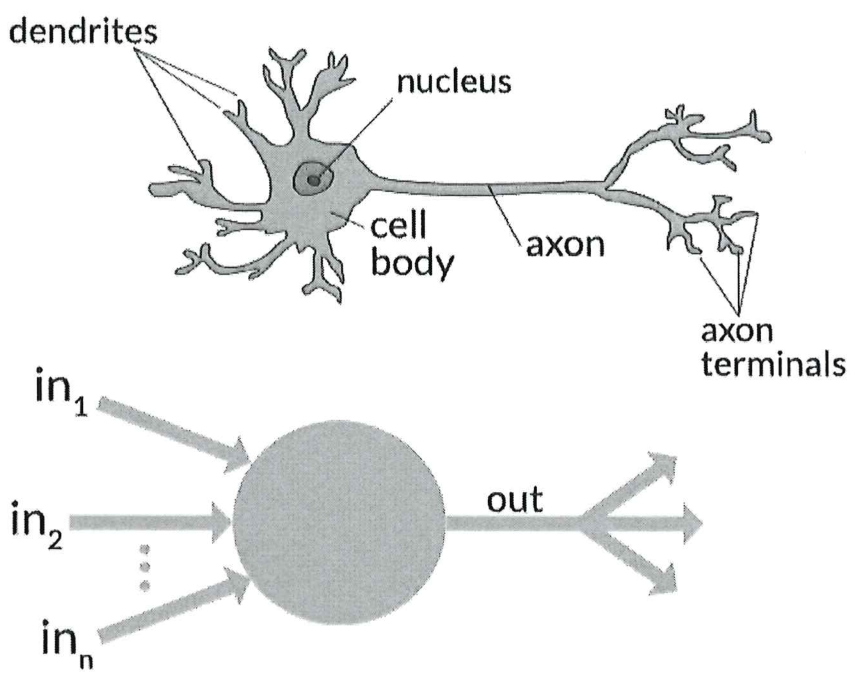
- new model for AI
- interconnected nodes weight bias
- layer
- output ← input
- logistic regression
- SGD (stochastic gradient descent)

What is a neuron?



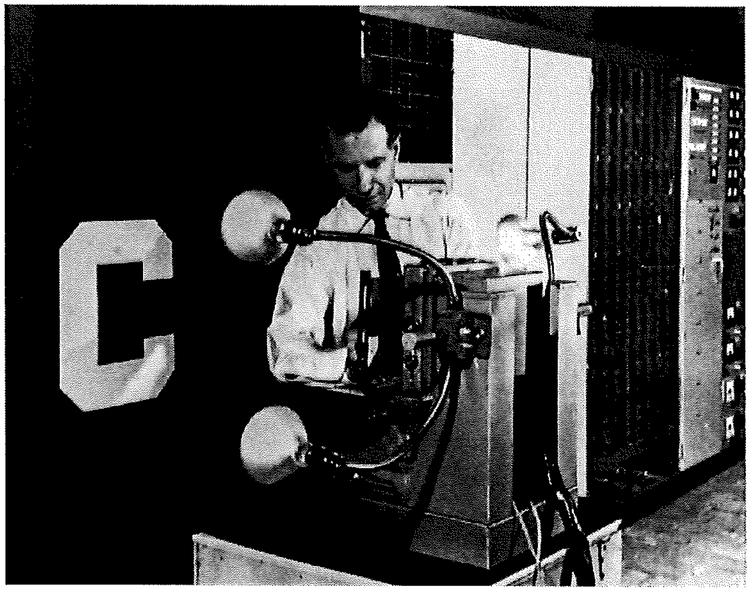
- a very long cell
- **dendrites** collect stimuli from other cells, chemical signals
- if the **action potential** is reached, the neuron fires

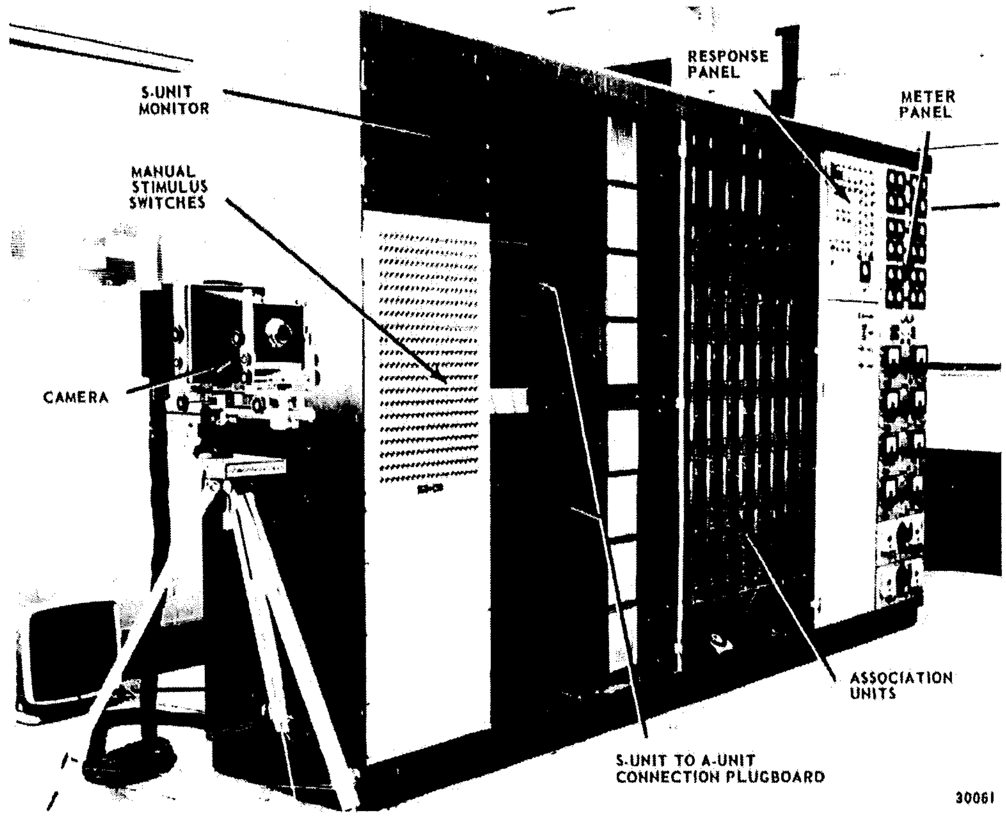
The Artificial Neuron (McCulloch-Pitts, 1943)



- binary inputs (either on or off)
- summation function / decision function
    - adds up the input values
- some *activation function*
    - decides whether to fire

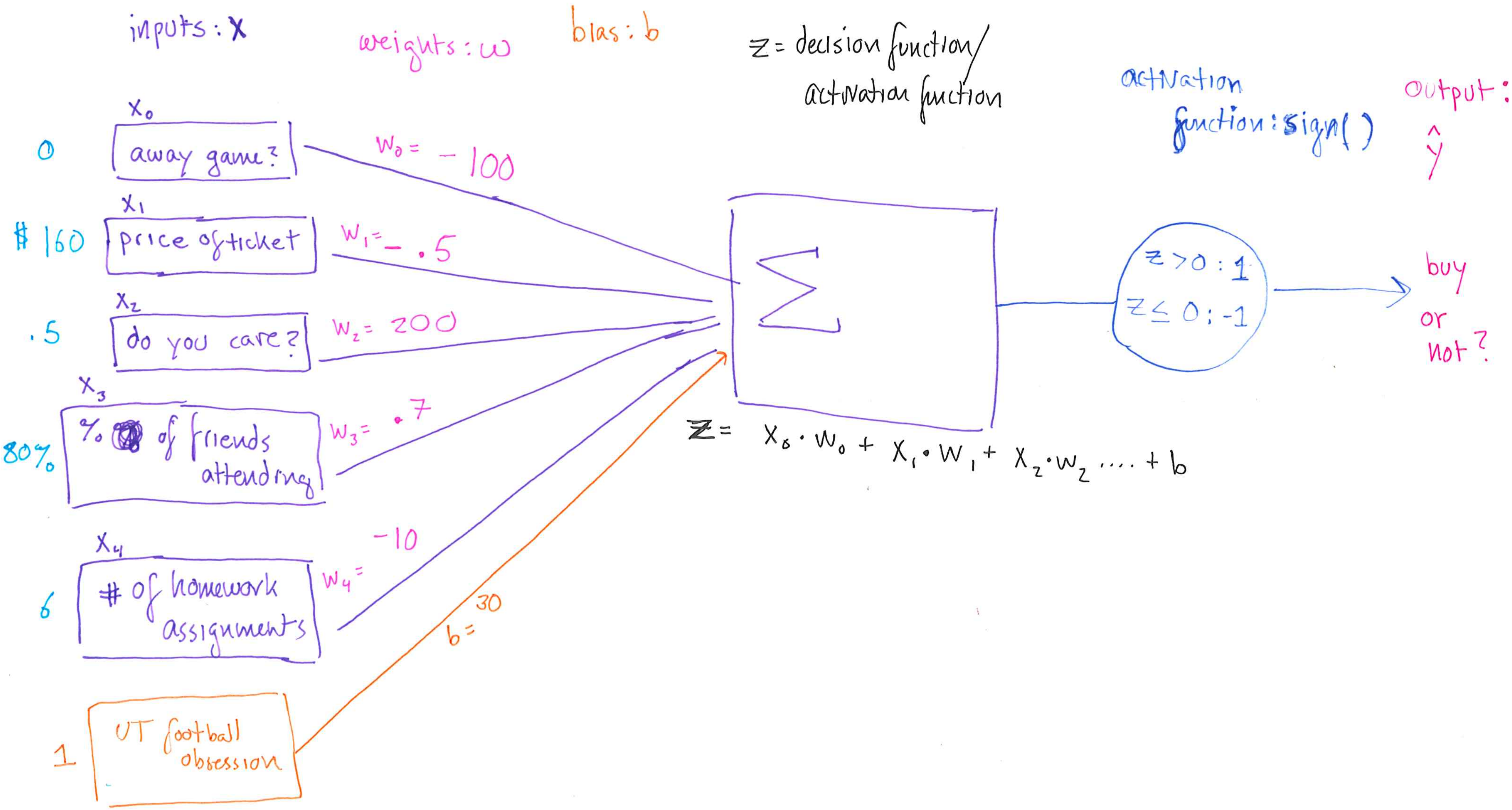
The Perceptron (Rosenblatt, 1958)



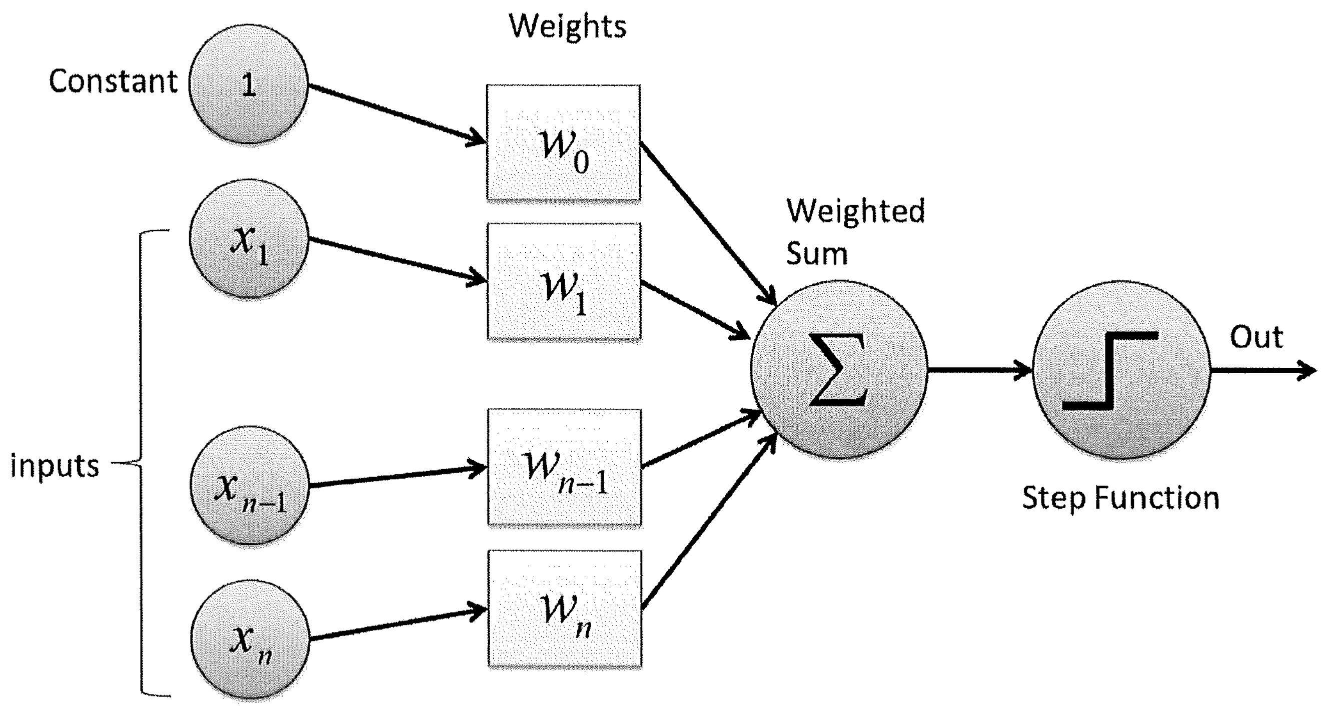


Perceptron Intuition

Should you buy a ticket to the next UT Game?



The Perceptron Algorithm



The Perceptron Algorithm

formalized as an equation, the Perceptron looks like this

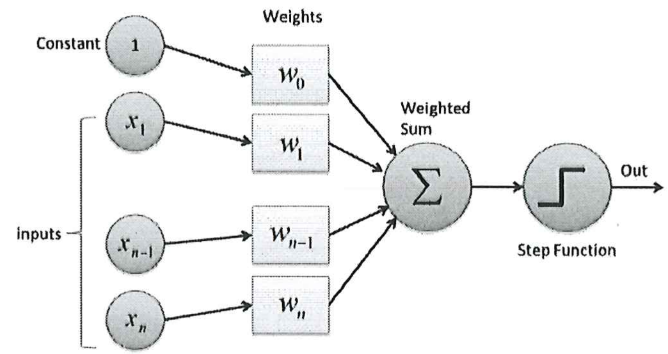
"y hat"

predicted value

weighted sum

$$\hat{y} = sign(z) = \begin{cases} 1, & \text{if } z > 0 \\ 0, & \text{if } z \leq 0 \end{cases}$$

step function



weighted sum

$$z = \left(\sum_{i=1}^{n} w_i x_i\right) + b$$

equivalent

$$z = (x_1 * w_1) + (x_2 * w_2) + (x_3 * w_3) + \ldots + (x_n * w_n) + b$$

$$\hat{y} = \vec{W} \cdot \vec{X} + b$$

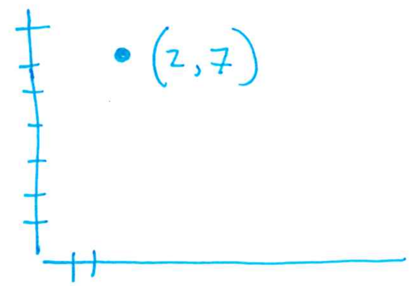
Vectors

The equation gets simpler if we think of the input as vectors.

A vector is an ordered list of numbers. $[1, -2, .3]$

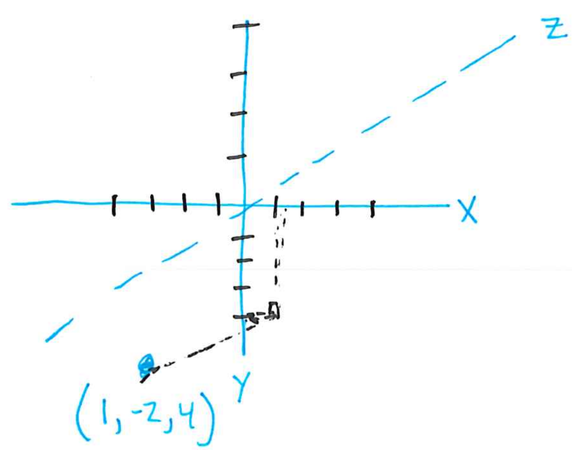
An ordered pair is a vector. $(2, 7)$

An ordered pair has a **geometric interpretation**.



A vector is an ordered pair with more numbers $\begin{array}{c}[1, -2, 4] \\ x \quad y \quad z\end{array}$

A vector also has a geometric interpretation



The Perceptron Algorithm (again)

$$\boxed{\hat{y} = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b)}$$

w is the weight vector ($w_1 w_2 \ldots w_i$)

x is the input vector ($x_1 x_2 \ldots x_i$)

b is the bias, still a regular number

sign( ) is the activation function

$\mathbf{w} \cdot \mathbf{x}$ is the dot product of **w** and **x**

the dot product is just shorthand for the weighted sum!

The Dot Product

Definition

$\mathbf{a} = \langle a_1, a_2, a_3 \rangle \quad \mathbf{b} = \langle b_1, b_2, b_3 \rangle$

$$\boxed{\mathbf{a} \cdot \mathbf{b} = a_1 b_1 + a_2 b_2 + a_3 b_3}$$