データと数理I
第3回

2章 ベクトルの基本/ノルム、距離、内積

武蔵野大学 データサイエンス学部データサイエンス学科
中西 崇文

ローカルルール

- 本授業はGoogle Classroom, Slack, Google Colaboratoryを使用します。
- 講義中なにかあればSlackのチャンネル「#120_データと数理」にこちらからも流しますし、みなさんもこちらに発言してください。
- グループワーク課題資料をGoogle Colaboratoryで提出する際は、その資料の表紙、および提出時のメッセージに、各メンバーがどの部分を貢献したかとその貢献度%(全体を100%として)を記述して提出してください

2章

ベクトル

教科書p.28～p.36を参照してください

- $\boldsymbol{x} = \begin{bmatrix} \text{札幌の気温} \\ \text{東京の気温} \\ \text{大阪の気温} \end{bmatrix} = \begin{bmatrix} -5.2 \\ 4.0 \\ 4.4 \end{bmatrix}$
  - $\boldsymbol{x}$をベクトルと呼ぶ
    - 大きさだけでなく、方向を持った量
    - 要素を(縦または横に)1列に並べたもの

  - スカラー
    - 大きさのみで、方向を持たない量
    - ベクトル空間においてベクトルを乗算することができる量(厳密な定義)

※ベクトルを表す$\boldsymbol{x}$はスカラーと区別するために太字で表すことに注意

列ベクトル、行ベクトル

- $\boldsymbol{x} = \begin{bmatrix} \text{札幌の気温} \\ \text{東京の気温} \\ \text{大阪の気温} \end{bmatrix} = \begin{bmatrix} -5.2 \\ 4.0 \\ 4.4 \end{bmatrix}$
  - **列ベクトル(column vector)**と呼ぶ
- $\boldsymbol{x} = [\text{札幌の気温} \quad \text{東京の気温} \quad \text{大阪の気温}] = [-5.2 \quad 4.0 \quad 4.4]$
  - **行ベクトル(row vector)**と呼ぶ
- 通常は列ベクトルで表現するが、教科書などはスペースの関係上、下記で表現する
  - $\boldsymbol{x} = [-5.2 \quad 4.0 \quad 4.4]^T = \begin{bmatrix} -5.2 \\ 4.0 \\ 4.4 \end{bmatrix}$
  - 上記のとおり、

Tを転置と呼ぶ。
(行列でも出てくる)

ベクトルの成分

- $\boldsymbol{x} = \begin{bmatrix} -5.2 \\ 4.0 \\ 4.4 \end{bmatrix}$
  - ベクトルに並んでいるそれぞれの数のことを成分(component)と呼ぶ
  - $\boldsymbol{x}$のi番目の成分を第i成分と呼び、 x_iと表記する
    - 例) 上記のベクトル$\boldsymbol{x}$の場合
      - $x_1 = -5.2$
      - $x_2 = 4.0$
      - $x_3 = 4.4$

ベクトルの基本演算

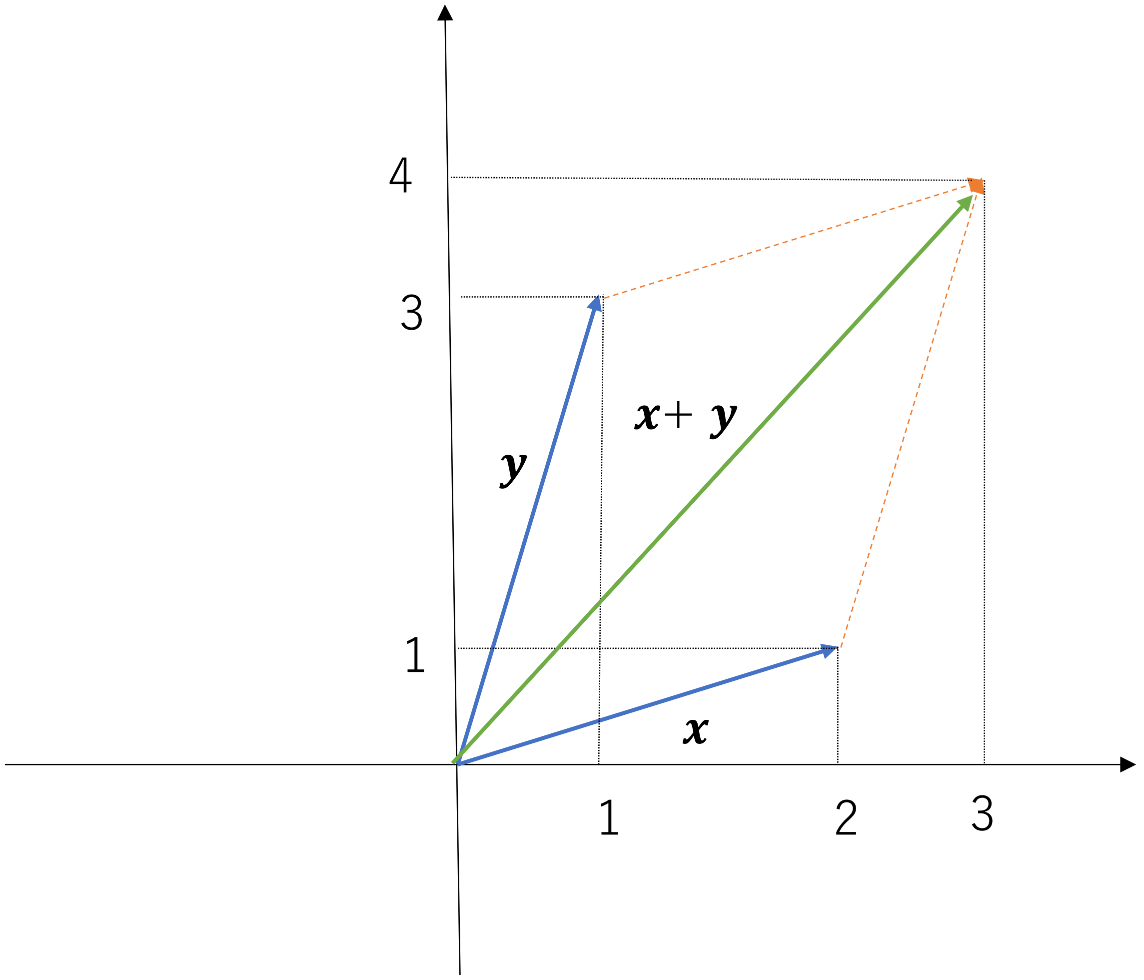
- ベクトルの和
  - ベクトルとスカラーの和は定義されないことに注意
  - 成分の数が同じもの同士しか演算できないことに注意

- スカラーとの積
  - ベクトル同士の場合は内積、外積という概念がある

※スカラーは「大きさのみで、方向を持たない量」と定義したが、ベクトルを掛け合わせて大きさを変える(Scale)ことができるからScalerというと考えれば良い

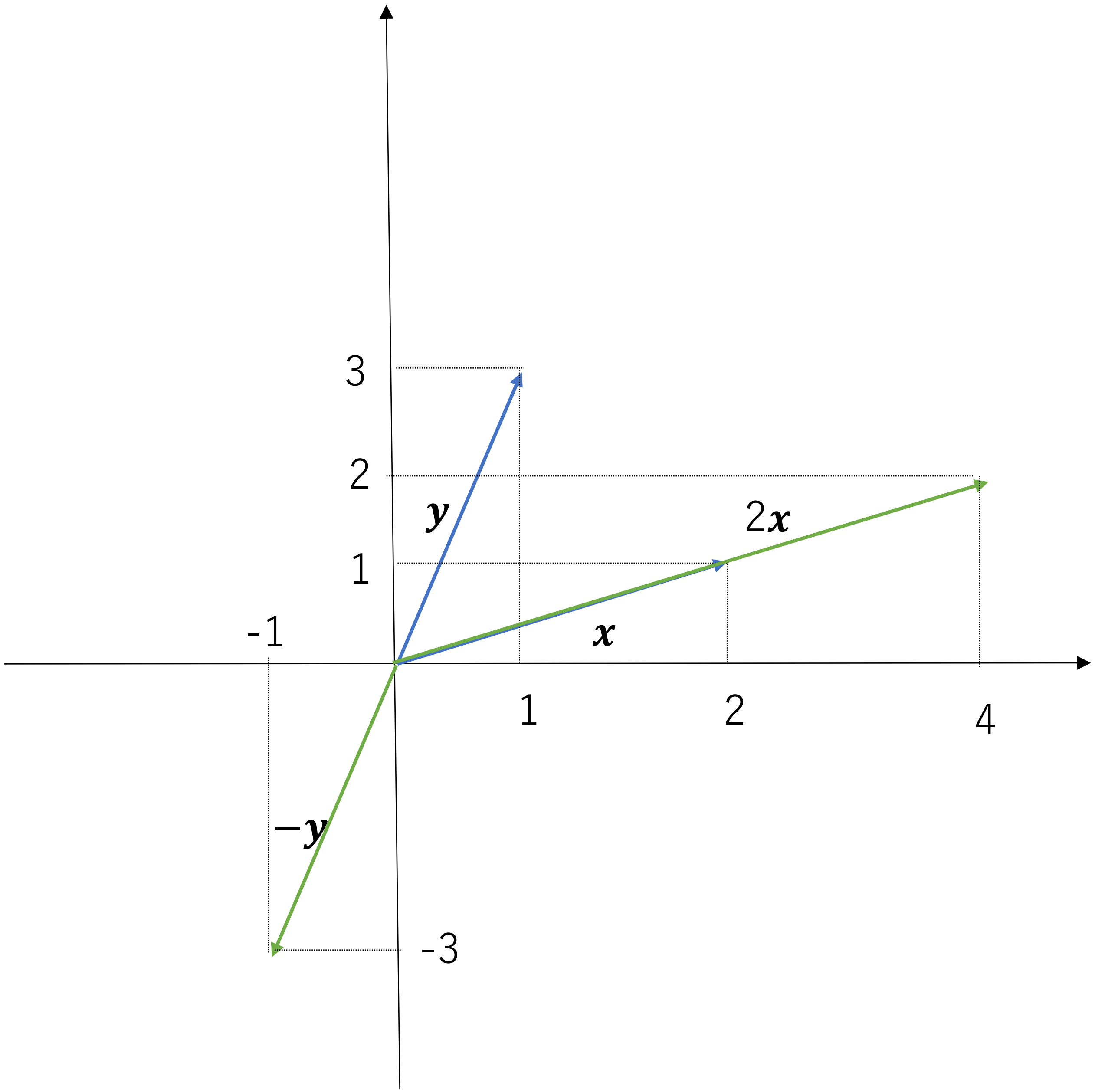
ベクトルの和

- $\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$
    - $\boldsymbol{x} + \boldsymbol{y} = \begin{bmatrix} x_1 + y_1 \\ x_2 + y_2 \\ \vdots \\ x_N + y_N \end{bmatrix}$
- 例) $\boldsymbol{x} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} 1 \\ 3 \end{bmatrix}$
    - $\boldsymbol{x} + \boldsymbol{y} = \begin{bmatrix} 2+1 \\ 1+3 \end{bmatrix} = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$



スカラーとの積

- $\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}$, スカラー$a$
  - $a\boldsymbol{x} = \begin{bmatrix} ax_1 \\ ax_2 \\ \vdots \\ ax_N \end{bmatrix}$
- 例) $\boldsymbol{x} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} 1 \\ 3 \end{bmatrix}$
  - $2\boldsymbol{x} = 2\begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 2\times 2 \\ 2\times 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$
  - $\boldsymbol{-y} = -1\begin{bmatrix} 1 \\ 3 \end{bmatrix} = \begin{bmatrix} -1\times 1 \\ -1\times 3 \end{bmatrix} = \begin{bmatrix} -1 \\ -3 \end{bmatrix}$



ベクトルの基本演算の性質

- ベクトル$\boldsymbol{x}, \boldsymbol{y}$および実数$a, b$に対して次が成り立つ
  - $\boldsymbol{x} + \boldsymbol{y} = \boldsymbol{y} + \boldsymbol{x}$
  - $(\boldsymbol{x} + \boldsymbol{y}) + \boldsymbol{z} = \boldsymbol{x} + (\boldsymbol{y} + \boldsymbol{z})$
  - $(a + b)\boldsymbol{x} = a\boldsymbol{x} + b\boldsymbol{x}$
  - $(ab)\boldsymbol{x} = a(b\boldsymbol{x}) = b(a\boldsymbol{x})$
  - $a(\boldsymbol{x} + \boldsymbol{y}) = a\boldsymbol{x} + b\boldsymbol{x}$

教科書p.33で確認してください

Pythonで計算してみよう

- $\boldsymbol{x} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} 1 \\ 3 \end{bmatrix}$
  - $\boldsymbol{x} + \boldsymbol{y}$
  - $\boldsymbol{x} - \boldsymbol{y}$
  - $2\boldsymbol{x}$
  - $-\boldsymbol{y}$
  - $2\boldsymbol{x} - \boldsymbol{y}$

- PythonではベクトルをNumpyのArrayで表現できるが、通常行ベクトルとなってしまうため、reshape(-1,1)で変形(転置)する必要がある。
  - reshapeの-1は他の次元の指定値から推測されて自動的に決定するという意味
  - reshape(-1,1)は列を1として行を自動的に決定するという意味

ベクトル同士の内積、外積、アダマール積

- $\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$とするとき、
    - ベクトル$\boldsymbol{x}, \boldsymbol{y}$の内積
        - $\boldsymbol{x} \cdot \boldsymbol{y} = \sum_{i=1}^{N} x_i y_i = \|\boldsymbol{x}\|\|\boldsymbol{y}\| \cos \boldsymbol{\theta}$
            - $\boldsymbol{x} \cdot \boldsymbol{y}$はスカラーである
    - ベクトル$\boldsymbol{x}, \boldsymbol{y}$の外積($N = 3$のとき)
        - $\boldsymbol{x} \times \boldsymbol{y} = \begin{bmatrix} x_2 y_3 - x_3 y_2 \\ x_3 y_1 - x_1 y_3 \\ x_1 y_2 - x_2 y_1 \end{bmatrix}$
            - $\boldsymbol{x} \times \boldsymbol{y}$はベクトルである
            - $\boldsymbol{x} \times \boldsymbol{y}$はベクトルの大きさが$\|\boldsymbol{x}\|\|\boldsymbol{y}\| \sin \boldsymbol{\theta}$で右ネジが進む方向に垂直
    - ベクトル$\boldsymbol{x}, \boldsymbol{y}$のアダマール積
        - $\boldsymbol{x} \otimes \boldsymbol{y} = \begin{bmatrix} x_1 y_1 \\ x_2 y_2 \\ \vdots \\ x_N y_N \end{bmatrix}$
            - $\boldsymbol{x} \otimes \boldsymbol{y}$はベクトルである

交換法則、結合法則、分配法則

- 交換法則
  - 与えられた演算の二つの引数を互いに入れ替えても結果が変わらないという法則
  - 集合E上に二つの元x, yで演算$*$が定められているとき、$x * y = y * x$のとき交換法則が満たされているという
    - 例）$5\times3 = 3\times5 = 15$のため交換法則が成り立つ
    - 交換法則を満たす演算は可換性を持つという

- 結合法則
  - 演算が結合的であるという法則
  - 集合E上に三つの元x, y, zで演算$*$が定められているとき$(x * y) * z = x * (y * z)$のとき結合法則が満たされているという
    - 例）$5\times(3\times2) = (5\times3)\times2 = 30$のため結合法則が成り立つ
    - 結合法則を満たす演算は結合性を持つという

- 分配法則
  - 集合E上に三つの元x, y, zで積$\times$と和$+$が定義されているとき$(x + y)\times z = x\times z + y\times z$および$x\times(y + z) = x\times y + x\times z$が成り立つとき、この積と和は分配法則が満たされているという
    - 例））$5\times(3 + 2) = 5\times3 + 5\times2$、$(5 + 3)\times2 = 5\times2 + 3\times2$のため分配法則が成り立つ

*元とは集合を構成する個々の数学的対象

内積、外積、アダマール積

- 内積
  - 交換法則は成立
  - 分配法則は成立
  - 結合法則は成り立たない
- 外積
  - 交換法則は成り立たない
  - 分配法則は成り立つ
  - 結合法則は成り立たない
- アダマール積
  - 交換法則は成り立つ
  - 分配法則は成り立つ
  - 結合法則は成り立つ

[コラム]外積は3次元のときしか定義がないのか？

- 1次元、3次元、7次元…のときに定義ができる
  - 複素数、四元数、八元数…

- 乗法のうまく定義できた n元数があれば，　n-1 次ベクトルの外積が定義できる

  - https://hooktail.sub.jp/vectoranalysis/SevenDCrossProd/

Concepts for mathematically creating your own world

- Space(空間)
- Metrics(メトリクス)

- By defining and preparing space and metrics, we can measure elements in the world.

Definition of Space(空間)

- A set S of elements x_i with a common mathematical structure
  - $S \in x_i$

  - Set(集合)
    - A collection of all objects that meet the given conditions
  - Element(元)
    - The individual mathematical objects that make up a set.

Definition of Metrics(メトリクス)

- A function that defines the relationship between two elements $\boldsymbol{x}_i, \boldsymbol{x}_j \in S$ in a defined space S
  - Function(関数)
    - A correspondence whereby for every element $\boldsymbol{x}_i$ of a set S, one element $\boldsymbol{y}_i$ of the set S or the other set S' is determined.
    - $\boldsymbol{y}_i = f(\boldsymbol{x}_i)$

空間とメトリクスを
もう少しイメージしやすく考える

- ものごと同士を比較するためには、そのものごとを入れておく「いれもの」とその「計算方法」が重要となる。
- 「いれもの」のことを空間(Space)と呼ぶ
- 「計算方法」のことをメトリクス(Metrics)と呼ぶ

- 「いれもの」とその「計算方法」が決まれば、比較していくことが可能

ベクトル空間

教科書p.35を参照してください

- ベクトルの和とスカラーとの積を導入したベクトル全体の集合のことをベクトル空間(vector space)と呼ぶ
- 厳密には下記
  - 集合Vの任意の要素(ベクトル) $\boldsymbol{x}, \boldsymbol{y}$に対してその和$\boldsymbol{x}+\boldsymbol{y}$が集合$V$の要素として定義され、任意の要素(スカラー) aと集合Vの任意の要素(ベクトル) $\boldsymbol{x}$に対してスカラーとの積$a\boldsymbol{x}$が集合Vの要素として定義されていて、下記の条件が満たされるとき、集合Vはベクトル空間と呼ばれる
    - $\boldsymbol{x}+\boldsymbol{y}=\boldsymbol{y}+\boldsymbol{x}$
    - $(\boldsymbol{x}+\boldsymbol{y})+\boldsymbol{z}=\boldsymbol{x}+(\boldsymbol{y}+\boldsymbol{z})$
    - 集合Vの任意の要素$\boldsymbol{x}, \boldsymbol{y}$に対して$\boldsymbol{x}+\boldsymbol{x}'=\boldsymbol{y}$を満たす集合$V$の要素$\boldsymbol{x}'$がただ一つ存在する
    - 集合Vの任意の要素$\boldsymbol{x}$に対して$\boldsymbol{x}+\boldsymbol{0}=\boldsymbol{x}$を満たす集合$V$の要素$\boldsymbol{0}$が存在する
    - $(a+b)\boldsymbol{x}=a\boldsymbol{x}+b\boldsymbol{x}$
    - $a(\boldsymbol{x}+\boldsymbol{y})=a\boldsymbol{x}+a\boldsymbol{x}$
    - $(ab)\boldsymbol{x}=a(b\boldsymbol{x})$
    - $1\boldsymbol{x}=\boldsymbol{x}$
  - 特に$\boldsymbol{x}+\boldsymbol{y}=\boldsymbol{0}$を満たす$\boldsymbol{y}$は$-\boldsymbol{x}$と記し、これを逆ベクトルと呼ぶ

教科書p.35を参照してください

ベクトル空間ではない例

- 例) $y = 2x - 1$上の点の集合
  - 和：$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} + \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} x_1 + x_2 \\ y_2 + y_2 \end{bmatrix}$
  - スカラー倍：$a\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} ax_1 \\ ay_1 \end{bmatrix}$

    が成立していない。

  - $\begin{bmatrix} 2 \\ 3 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 4 \end{bmatrix} \rightarrow \begin{bmatrix} 3 \\ 4 \end{bmatrix}$は$y = 2x - 1$上の点ではない
  - 2$\begin{bmatrix} 2 \\ 3 \end{bmatrix} = \begin{bmatrix} 4 \\ 6 \end{bmatrix} \rightarrow \begin{bmatrix} 4 \\ 6 \end{bmatrix}$は$y = 2x - 1$上の点ではない
  - つまり、この場合ベクトル空間とは言えない

単位ベクトル

教科書p.37を参照してください

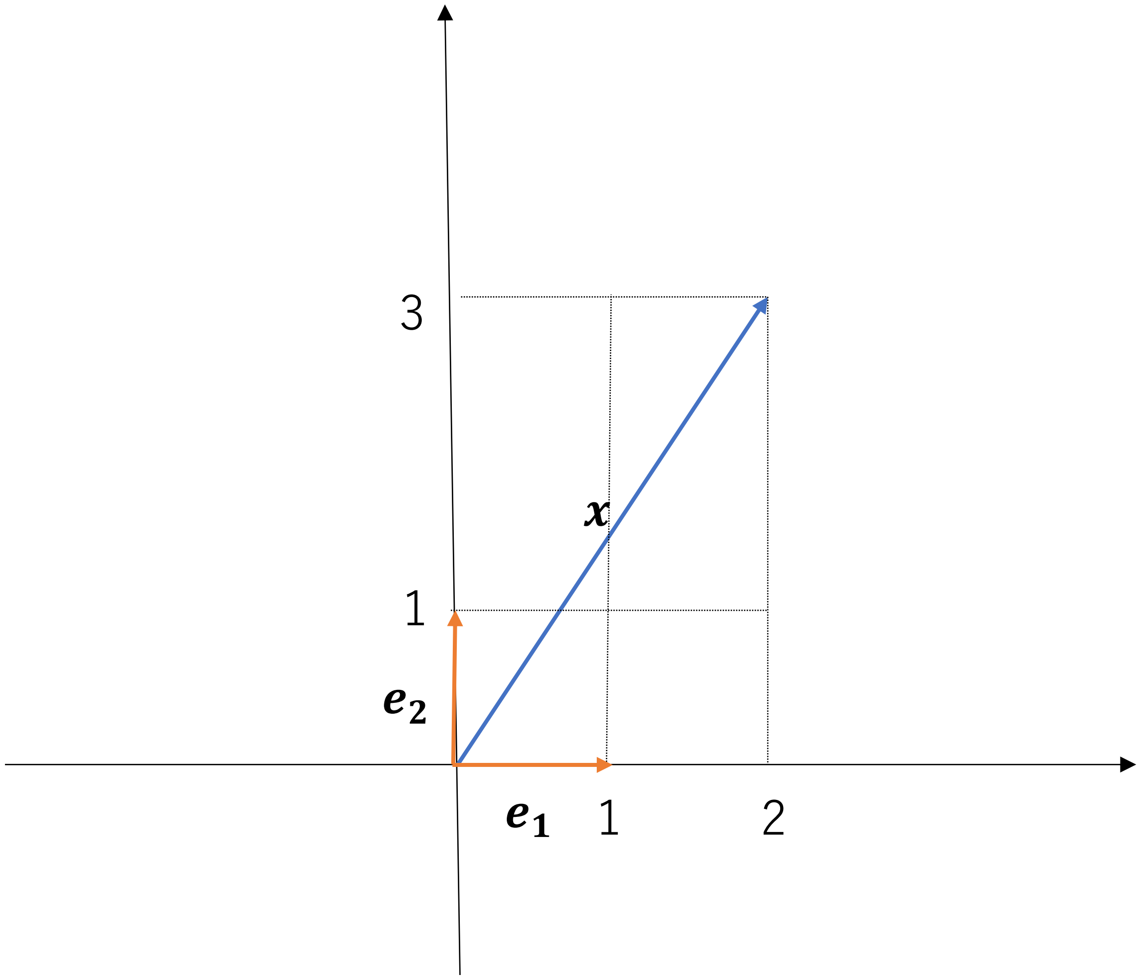
- ベクトル$\boldsymbol{x}$と同じ向きの大きさ1のベクトルのことを単位ベクトル$\boldsymbol{e}$と呼ぶ
  - $\boldsymbol{e} = \frac{1}{\|\boldsymbol{x}\|}\boldsymbol{x}$
  - $\|\boldsymbol{x}\| = \sqrt{|x_1|^2 + |x_2|^2 + \cdots + |x_N|^2}$
  - ベクトルの大きさを1にすることを正規化すると呼ぶ

教科書p.39を参照してください

ベクトルの分解[1/2]

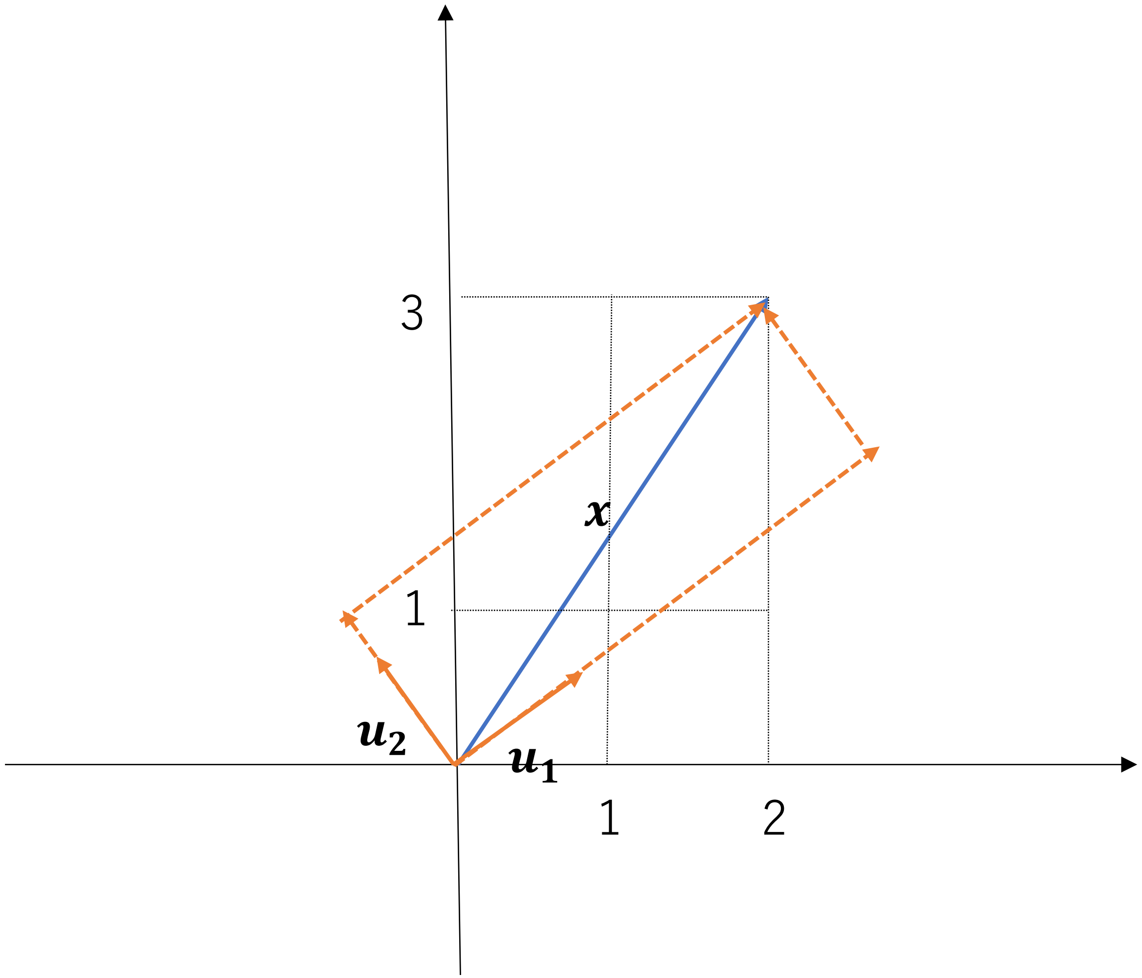
- $\boldsymbol{x} = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$ は$\boldsymbol{e}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \boldsymbol{e}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ の単位ベクトルを用いて、
$$\boldsymbol{x} = 2\boldsymbol{e}_1 + 3\boldsymbol{e}_2$$
と表現できる

- これをベクトルの分解と呼ぶ



ベクトルの分解[2/2]

- $x = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$ は他のベクトル$\boldsymbol{u_1}, \boldsymbol{u_2}$ でも分解可能
  - $x = \mathrm{a}_1\boldsymbol{u_1} + \mathrm{a}_2\boldsymbol{u_2}$ と表現できる
- これを線形結合と呼ぶ



教科書p.41を参照してください

線形独立/線形従属

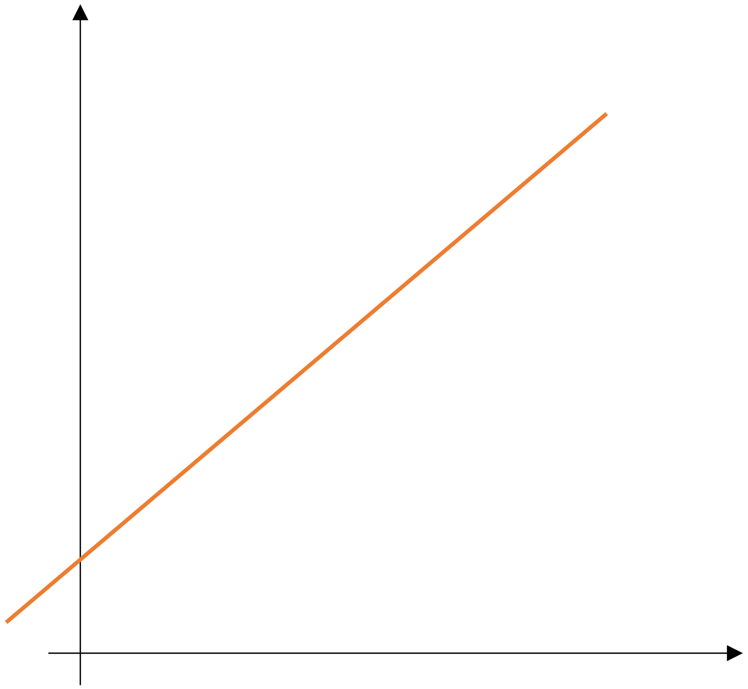
- N個のスカラー$a_1, a_2 \cdots, a_N$、N個のベクトル$\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots \boldsymbol{u}_N$に対して$a_1\boldsymbol{u}_1 + a_2\boldsymbol{u}_2 + \cdots + a_N\boldsymbol{u}_N$を線形結合($Linear\ combination$)といい

$$a_1\boldsymbol{u}_1 + a_2\boldsymbol{u}_2 + \cdots + a_N\boldsymbol{u}_N = \boldsymbol{0}$$

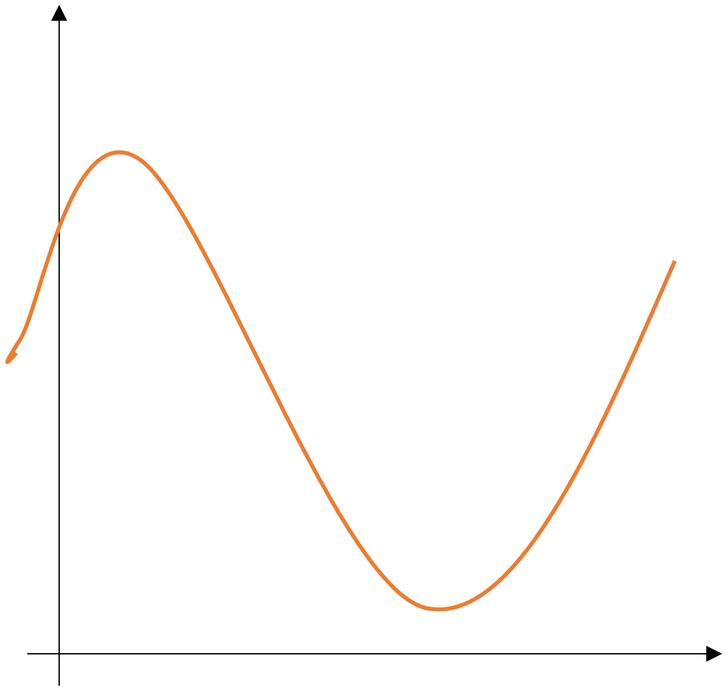
　となる必要十分条件が$a_1 = a_2 = \cdots = a_N = 0$であるとき、$\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots \boldsymbol{u}_N$は線形独立(linearly independent)といい、そうでない場合は線形従属(linearly dependent)であるという。

線形独立を一次独立、線形従属を一次従属という場合もある

線形・非線形

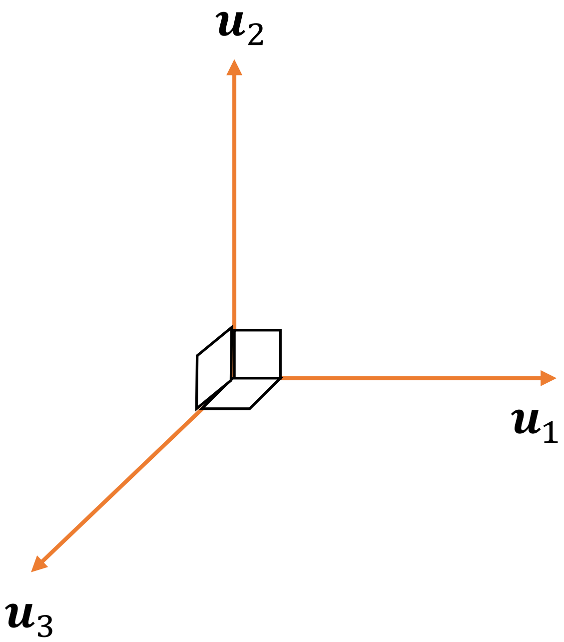


線形



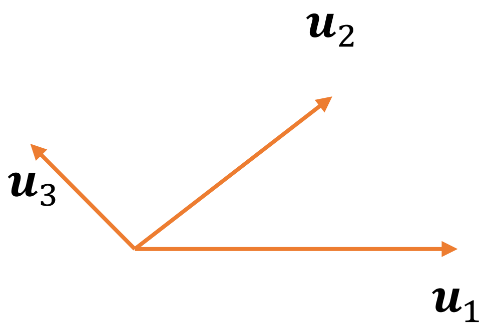
非線形

線形独立/線形従属のイメージ



$\boldsymbol{u}_3$を$a_1\boldsymbol{u}_1 + a_2\boldsymbol{u}_2$で表すことは**できない**
$\boldsymbol{u}_2$を$a_1\boldsymbol{u}_1 + a_3\boldsymbol{u}_3$で表すことは**できない**
$\boldsymbol{u}_1$を$a_2\boldsymbol{u}_2 + a_3\boldsymbol{u}_3$で表すことは**できない**

線形独立



$\boldsymbol{u}_3$を$a_1\boldsymbol{u}_1 + a_2\boldsymbol{u}_2$で表すことは**できる**
$\boldsymbol{u}_2$を$a_1\boldsymbol{u}_1 + a_3\boldsymbol{u}_3$で表すことは**できる**
$\boldsymbol{u}_1$を$a_2\boldsymbol{u}_2 + a_3\boldsymbol{u}_3$で表すことは**できる**

線形従属

線形独立の例

- $\boldsymbol{u}_1 = \begin{bmatrix} 1 \\ 0 \\ 2 \end{bmatrix}, \boldsymbol{u}_2 = \begin{bmatrix} 0 \\ 1 \\ 4 \end{bmatrix}, \boldsymbol{u}_3 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$

- $a_1\boldsymbol{u}_1 + a_2\boldsymbol{u}_2 + a_3\boldsymbol{u}_3 = 0$
  となるのは$a_1 = a_2 = a_3 = 0$のときのみ→線形独立

  - $a_1 + a_3 = 0$
  - $a_2 + a_3 = 0$
  - $2a_1 + 4a_2 + a_3 = 0$ という連立1次方程式を解けば良い

https://oguemon.com/study/linear-algebra/linearly-independent/

線形従属の例

- $\boldsymbol{u}_1 = \begin{bmatrix} 1 \\ 0 \\ 2 \end{bmatrix}, \boldsymbol{u}_2 = \begin{bmatrix} 0 \\ 1 \\ 4 \end{bmatrix}, \boldsymbol{u}_3 = \begin{bmatrix} 2 \\ -1 \\ 0 \end{bmatrix}$
- $a_1\boldsymbol{u}_1 + a_2\boldsymbol{u}_2 + a_3\boldsymbol{u}_3 = 0$
となるのは$a_1 = a_2 = a_3 = 0$以外に
$a_1 = 2\lambda, a_2 = -\lambda, a_3 = -\lambda$がある→線形従属

  - $a_1 + 2a_3 = 0$
  - $a_2 - a_3 = 0$
  - $2a_1 + 4a_2 = 0$ という連立1次方程式を解けば良い

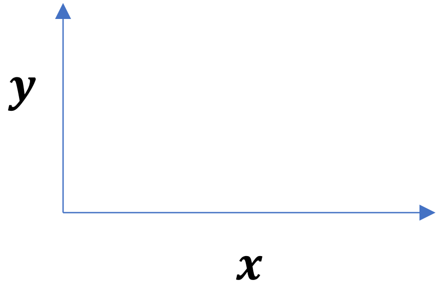
https://oguemon.com/study/linear-algebra/linearly-independent/

基底

- ベクトル空間Vの$\boldsymbol{u_1}, \boldsymbol{u_2}, \cdots \boldsymbol{u_N}$が線形独立であり、 Vの任意の要素がこれらのベクトルの線形結合で表現できるとき、集合$\{\boldsymbol{u_1}, \boldsymbol{u_2}, \cdots \boldsymbol{u_N}\}$のことを$V$の基底とよぶ。また$N$を$V$の次元と呼ぶ。
  - $dimV = N$
  - $\boldsymbol{u}_i$を基底ベクトルと呼ぶ

直交

- ベクトル$\boldsymbol{x}, \boldsymbol{y}$が内積$\boldsymbol{x} \cdot \boldsymbol{y} = \boldsymbol{0}$を満たすとき、ベクトル$\boldsymbol{x}, \boldsymbol{y}$は直交する(orthogonal)という



$$\boldsymbol{x} \cdot \boldsymbol{y} = \|\boldsymbol{x}\|\|\boldsymbol{y}\| cos\theta$$

$cos 90° = 0$だから

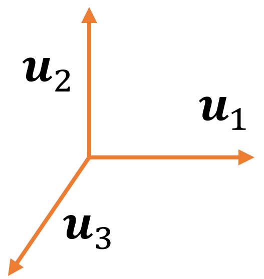
$\boldsymbol{x} \cdot \boldsymbol{y} =0$

正規直交基底

教科書p.55を参照してください

- ベクトル空間Vの$\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots \boldsymbol{u}_N$が線形独立であり、$V$の任意の要素がこれらのベクトルの線形結合で表現できるとき、集合$\{\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots \boldsymbol{u}_N\}$のことを$V$の基底という。

- 上記の基底となる$\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots \boldsymbol{u}_N$のどの2つも互いに直交し、かつ$\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots \boldsymbol{u}_N$それぞれのノルム が1に等しいとき正規直交基底(orthonormal basis)であるという



正規直交基底だと何がうれしいか？[1/2]

- 2つのベクトル$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$のとき基底が$\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3$であれば、
  $\boldsymbol{x} = x_1\boldsymbol{u}_1 + x_2\boldsymbol{u}_2 + x_3\boldsymbol{u}_3$,
  $\boldsymbol{y} = y_1\boldsymbol{u}_1 + y_2\boldsymbol{u}_2 + y_3\boldsymbol{u}_3$と表せる。

正規直交基底だと何がうれしいか？[2/2]

- $\boldsymbol{x} = x_1\boldsymbol{u}_1 + x_2\boldsymbol{u}_2 + x_3\boldsymbol{u}_3$, $\boldsymbol{y} = y_1\boldsymbol{u}_1 + y_2\boldsymbol{u}_2 + y_3\boldsymbol{u}_3$のとき、
- 内積は
- $\boldsymbol{x}\cdot\boldsymbol{y} = (x_1\boldsymbol{u}_1 + x_2\boldsymbol{u}_2 + x_3\boldsymbol{u}_3)\,(y_1\boldsymbol{u}_1 + y_2\boldsymbol{u}_2 + y_3\boldsymbol{u}_3)$

$$
\begin{aligned}
&= x_1y_1\boldsymbol{u}_1\boldsymbol{u}_1 + x_2y_1\boldsymbol{u}_2\boldsymbol{u}_1 + x_3y_1\boldsymbol{u}_3\boldsymbol{u}_1 \\
&\qquad + x_1y_2\boldsymbol{u}_1\boldsymbol{u}_2 + x_2y_2\boldsymbol{u}_2\boldsymbol{u}_2 + x_3y_2\boldsymbol{u}_3\boldsymbol{u}_2 \\
&\qquad + x_1y_3\boldsymbol{u}_1\boldsymbol{u}_3 + x_2y_3\boldsymbol{u}_2\boldsymbol{u}_3 + x_3y_3\boldsymbol{u}_3\boldsymbol{u}_3
\end{aligned}
$$

ごっちゃり

もし、基底$\boldsymbol{u}_1, \boldsymbol{u}_2, \boldsymbol{u}_3$が正規直交基底だとしたら、

$$
\boldsymbol{u}_1\boldsymbol{u}_1 = \boldsymbol{u}_2\boldsymbol{u}_2 = \boldsymbol{u}_3\boldsymbol{u}_3 = 1
$$

$$
\boldsymbol{u}_1\boldsymbol{u}_2 = \boldsymbol{u}_2\boldsymbol{u}_3 = \boldsymbol{u}_1\boldsymbol{u}_3 = 0
$$

なので

$$
\boldsymbol{x}\cdot\boldsymbol{y} = x_1y_1 + x_2y_2 + x_3y_3
$$

すっきり

正規直交基底では

- 正規直交基底では、内積やベクトルの大きさの計算が標準基底(軸上の単位ベクトルからなる基底)と同じように計算ができる

つまり計算が楽になる！

部分空間

- ベクトル空間Vの部分集合Wが次の条件を満たすとき、 Wは部分空間(subspace)であるといい、 $W \subset V$と表記する。
  - （1）部分集合Wの任意の$\boldsymbol{x}, \boldsymbol{y}$に対して、 $\boldsymbol{x} + \boldsymbol{y}$も部分集合$W$の要素
  - （2）部分集合Wの任意の$\boldsymbol{x}$とスカラー aに対して$a\boldsymbol{x}$も部分集合Wの要素

ベクトル同士を比べる

教科書p.42を参照してください

- ノルム→ベクトルの大きさ
- 距離→ベクトル間の差
- 内積→ベクトル間の作用関係
  - コサイン類似度→ベクトルの向きの類似の比較

ベクトルの大きさ

- $\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}$のときベクトルの大きさは下記のように定義
  - $\|\boldsymbol{x}\| = \sqrt{|x_1|^2 + |x_2|^2 + \cdots + |x_N|^2}$
  - 実はこれはL2ノルムと呼ばれるもの

- 例)
  - $\boldsymbol{x} = \begin{bmatrix} 4 \\ -3 \end{bmatrix}$のベクトルの大きさ$\|\boldsymbol{x}\| = \sqrt{|4|^2 + |-3|^2} = \sqrt{16 + 9} = \sqrt{25}$=5
  - Pythonでの計算 np.linalg.norm(x)

L1ノルム,L2ノルム,L∞ノルム

- $\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}$のときベクトルのノルムは下記のように定義
  - $\|\boldsymbol{x}\|_p = \sqrt[p]{|x_1|^p + |x_2|^p + \cdots + |x_N|^p}$

  - p=1のとき、各成分の絶対値の和
    - Pythonでは、np.linalg.norm(x,1)
  - p=2のとき、いわゆるベクトルの長さ
    - Pythonでは、np.linalg.norm(x,2)
  - p=∞のとき、 x_i(1≤i≤n) の中で一番絶対値が大きいものの一つを $\boldsymbol{x_k}$
    - Pythonでは、np.linalg.norm(x,np.inf)

距離(高校で習ったもの➔ユークリッド距離)

- $\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$のときベクトル同士のユークリッド距離は下記のように定義
  - $d(\boldsymbol{x}, \boldsymbol{y}) = d(\boldsymbol{y}, \boldsymbol{x}) = \sqrt{|x_1 - y_1|^2 + |x_2 - y_2|^2 + \cdots + |x_N - y_N|^2}$
- 例)
  - $\boldsymbol{x} = \begin{bmatrix} 4 \\ -3 \end{bmatrix}, \mathbf{y} = \begin{bmatrix} 2 \\ 4 \end{bmatrix}$のユークリッド距離
  - $d(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{|4-2|^2 + |-3-4|^2} = \sqrt{|2|^2 + |-7|^2} = \sqrt{4+49} = \sqrt{53} = 7.28..$
- Pythonでの計算 np.linalg.norm(x-y) または、

```
from scipy.spatial import distance
distance.euclidean(x,y)
```

マンハッタン距離、ユークリッド距離、チェビシェフ距離

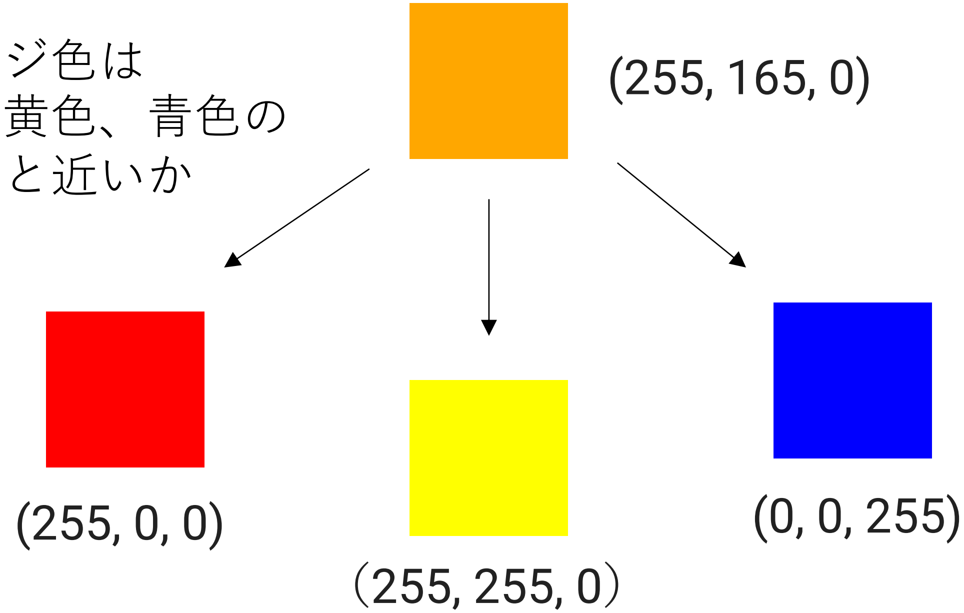
- $\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$のときベクトル同士の距離は下記のように定義
  - $d_p(\boldsymbol{x}, \boldsymbol{y}) = d_p(\boldsymbol{y}, \boldsymbol{x}\,) = \sqrt[p]{|x_1 - y_1|^p + |x_2 - y_2|^p + \cdots + |x_N - y_N|^p}$
  - p=1のとき、マンハッタン距離
    - python ではfrom scipy.spatial.distanceのcityblock(x, y)
  - p=2のとき、ユークリッド距離
    - python ではfrom scipy.spatial.distanceのeuclidean(x, y)
  - p=∞のとき、チェビシェフ距離
    - python ではfrom scipy.spatial.distanceのchebyshev(x, y)

その他、マハラノビス距離というのがよく使われる。調べてみよう。（レポート出せば加点します）

Pythonで計算してみよう
オレンジ色は赤色と黄色と青色とどちらに近いの？(距離編)

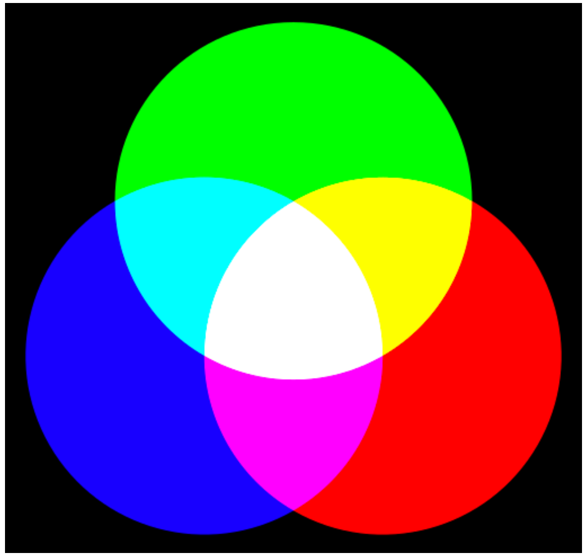
Q

オレンジ色は
赤色、黄色、青色の
どの色と近いか



それぞれの色のRGB値をベクトルとして、
ユークリッド距離で計算し、**<u>値が小さいほど</u>**
<u>近いこと</u>になる。

RGB



色の表現法の一種で、赤 (**R**ed) 、緑 (**G**reen)、青 (**B**lue) の三つの原色を混ぜて幅広い色を再現する
https://ja.wikipedia.org/wiki/RGB
それぞれの色の明度は0から255までで表現

応用：類似画像検索、画像認識、画像クラスタリングなど

内積

- $\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$のときベクトル同士の内積は下記のように定義
  - $\boldsymbol{x} \cdot \boldsymbol{y} = x_1 y_1 + x_2 y_2 + \cdots + x_N\, y_N$

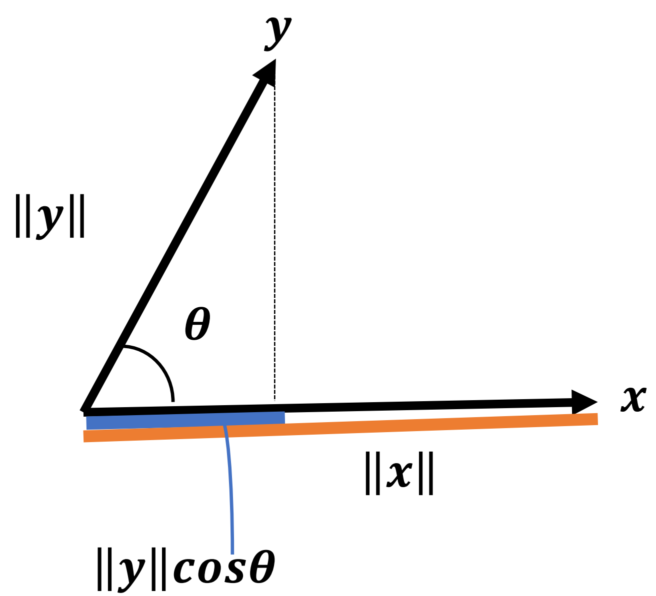
- 例)
  - $\boldsymbol{x} = \begin{bmatrix} 4 \\ -3 \end{bmatrix}, \mathbf{y} = \begin{bmatrix} 2 \\ 4 \end{bmatrix}$の内積
  - $\boldsymbol{x} \cdot \boldsymbol{y} \ = 4 \times 2 + (-3) \times 4 = -4$

- Pythonでの計算 np.dot(x.T,y)

.Tは転置の意味。
Pythonのdotの計算は行ベクトルと列ベクトルで計算
実際、内積は厳密には行ベクトルと列ベクトルの演算と考えることができる
(なぜそうなるかは行列の積で解説する)

内積とコサイン類似度

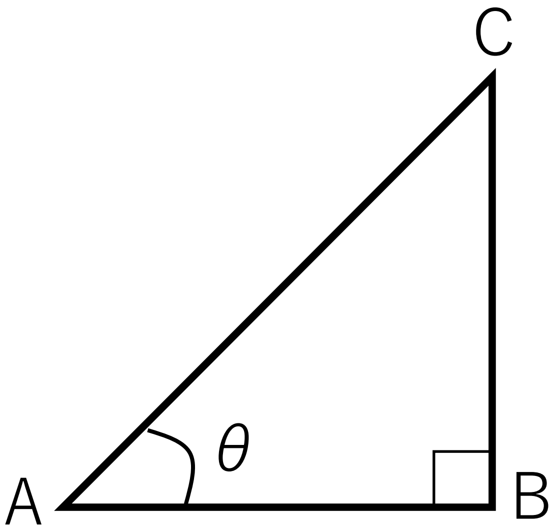
- $\boldsymbol{x} \cdot \boldsymbol{y} = \|\boldsymbol{x}\|\|\boldsymbol{y}\|cos\theta$
  - ベクトル$\boldsymbol{x}$の大きさと
    ベクトル$\boldsymbol{y}$のベクトル$\boldsymbol{x}$方向の大きさ
    の掛け算が内積



- コサイン類似度
  - (ベクトル$\boldsymbol{x}$と$\boldsymbol{y}$の類似度を計算するための尺度)
  - $cos\theta = \frac{\boldsymbol{x} \cdot \boldsymbol{y}}{\|\boldsymbol{x}\|\|\boldsymbol{y}\|}$

(忘れた人用)三角関数を復習しよう

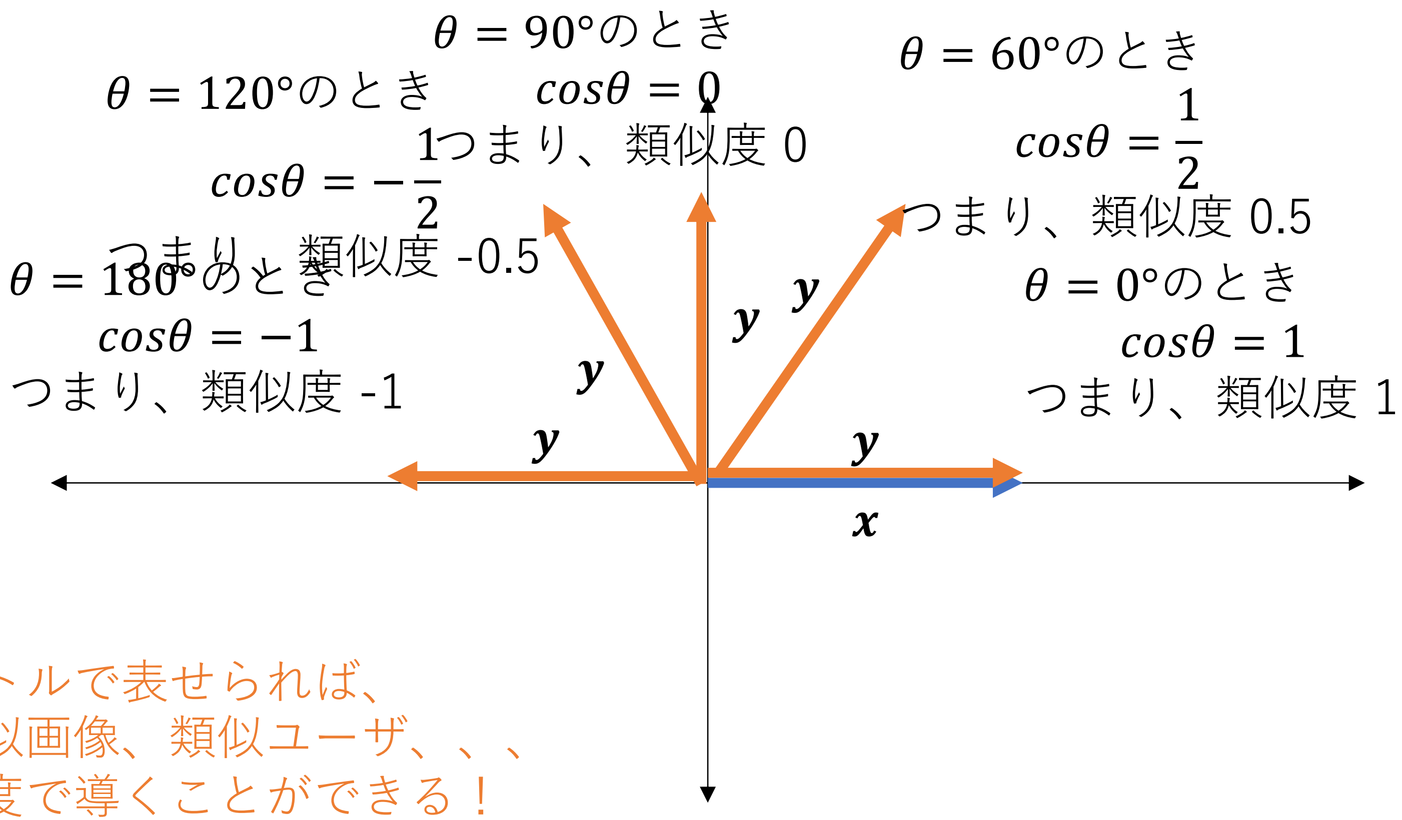
- $sin\theta = \frac{BC}{AC}$
- $cos\theta = \frac{AB}{AC}$
- $tan\theta = \frac{BC}{AB}$



度数法(度:°)	0°	30°	45°	60°	90°	120°	135°	150°	180°
弧度法(ラジアン:rad)	0	$\frac{\pi}{6}$	$\frac{\pi}{4}$	$\frac{\pi}{3}$	$\frac{\pi}{2}$	$\frac{2}{3}\pi$	$\frac{3}{4}\pi$	$\frac{5}{6}\pi$	π
$\sin\theta$	0	$\frac{1}{2}$	$\frac{1}{\sqrt{2}}$	$\frac{\sqrt{3}}{2}$	1	$\frac{\sqrt{3}}{2}$	$\frac{1}{\sqrt{2}}$	$\frac{1}{2}$	0
$\cos\theta$	1	$\frac{\sqrt{3}}{2}$	$\frac{1}{\sqrt{2}}$	$\frac{1}{2}$	0	$-\frac{1}{2}$	$-\frac{1}{\sqrt{2}}$	$-\frac{\sqrt{3}}{2}$	-1
$\tan\theta$	0	$\frac{1}{\sqrt{3}}$	1	$\sqrt{3}$	$-$	$-\sqrt{3}$	-1	$-\frac{1}{\sqrt{3}}$	0

$\boldsymbol{cos\theta}$がなぜ類似度計算となるのか？

ベクトル$\boldsymbol{x}$をXさんの趣味、ベクトル$\boldsymbol{y}$をYさんの趣味と設定すると、コサイン類似度で相性診断ができる



つまり、ベクトルで表せれば、
類似文章、類似画像、類似ユーザ、、、
コサイン類似度で導くことができる！

コサイン類似度

- $\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$のときベクトル間のコサイン類似度は下記のように定義される
  - $cos\theta = \frac{\boldsymbol{x}\cdot\boldsymbol{y}}{\|\boldsymbol{x}\|\|\boldsymbol{y}\|} = \frac{x_1y_1+x_2y_2+\cdots+x_Ny_N}{\sqrt{|x_1|^2+|x_2|^2+\cdots+|x_N|^2}\sqrt{|y_1|^2+|y_2|^2+\cdots+|y_N|^2}}$

各ベクトルの成分を使うだけで$cos\theta$が導出できる

- 例）
  - $\boldsymbol{x} = \begin{bmatrix} 4 \\ -3 \end{bmatrix}, \mathbf{y} = \begin{bmatrix} 2 \\ 4 \end{bmatrix}$のコサイン類似度
  - $cos\theta = \frac{\boldsymbol{x}\cdot\boldsymbol{y}}{\|\boldsymbol{x}\|\|\boldsymbol{y}\|} = \frac{4\times2+(-3)\times4}{\sqrt{|4|^2+|-3|^2}\sqrt{|2|^2+|4|^2}} = \frac{-4}{\sqrt{25}\sqrt{20}} = -0.178 \ldots$

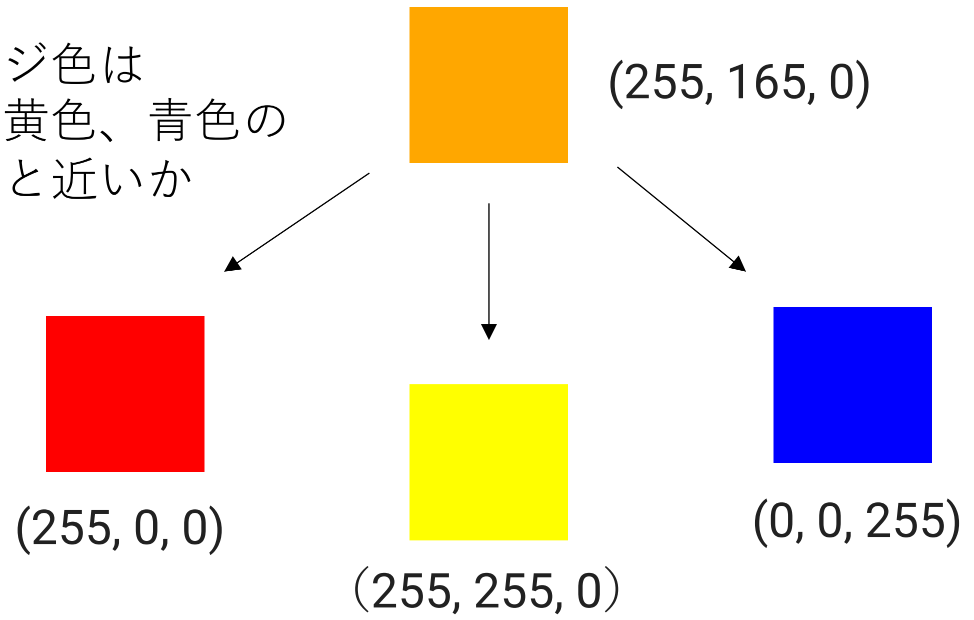
Pythonでの計算の場合、下記の関数を定義

```
import numpy as np
def cos_similarity(x,y):
        return np.dot(x.T, y)[0][0] / (np.linalg.norm(x) * np.linalg.norm(y))
```

Pythonで計算してみよう
オレンジ色は赤色と黄色と青色とどちらに近いの？(コサイン類似度編)

Q

オレンジ色は
赤色、黄色、青色の
どの色と近いか



それぞれの色のRGB値をベクトルとして、
コサイン類似度で計算し、
値が大きいほどほど近いことになる。

距離の場合は
値が小さいほど近い

類似度の場合は
値が大きいほど近い

応用：類似画像検索、画像認識、画像クラスタリングなど

Embedding

- An operation to convert words, sentences, image data, image data, etc., into vector representations
  - Word2vec, BERT, AugNet, pyannote.audio

Concept of realizing embedded expressions

- Dimensional reduction
  - Eigenvalue decomposition, Singular value decomposition
  - Encoder-Decoder model

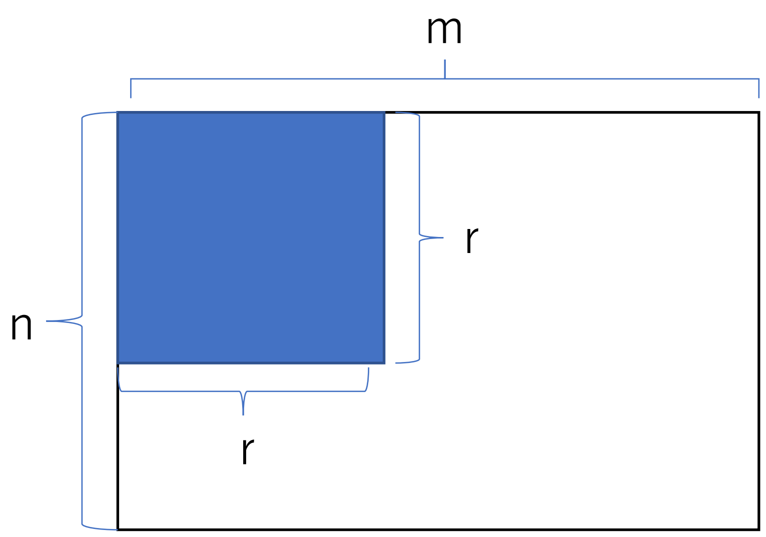
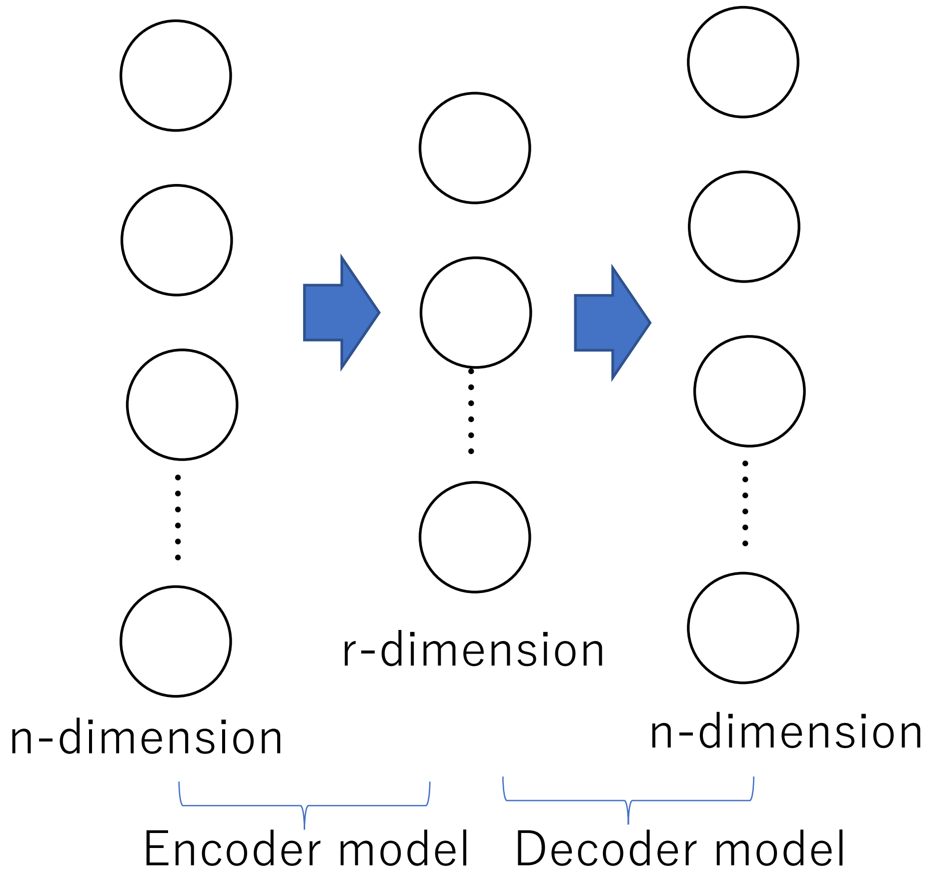


Image of dimensional reduction for a matrix



ノルムの公理(厳密に)

- ベクトル空間の一つの元$\boldsymbol{x}, \boldsymbol{y}$に下記の性質を満たす$|\boldsymbol{x}|$を決めることができるとき、これをノルムと呼ぶ
  - 任意の実数aに対して$\|a\boldsymbol{x}\| = |a|\|\boldsymbol{x}\|$ (線形性)
  - $\|\boldsymbol{x}\| \geq 0$,$\|\boldsymbol{x}\|$ =0であれば$\boldsymbol{x} = \boldsymbol{0}$(正値性)
  - $\|\boldsymbol{x} + \boldsymbol{y}\| \leq \|\boldsymbol{x}\| + \|\boldsymbol{y}\|$(三角不等式)

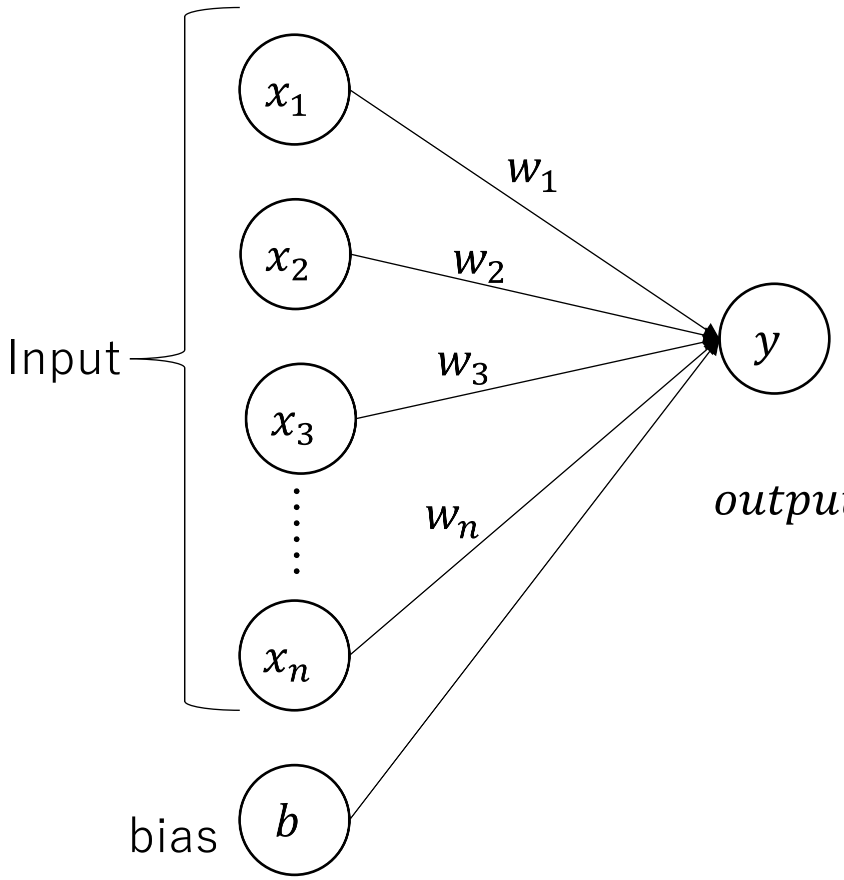
距離の公理(厳密に)

- ベクトル空間の元$\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$間に下記の性質を満たす$\rho(\boldsymbol{x}, \boldsymbol{y})$を決めることができるとき、これを距離と呼ぶ
  - $\rho(\boldsymbol{x}, \boldsymbol{y}) = \rho(\boldsymbol{y}, \boldsymbol{x})$ (対称性)
  - $\rho(\boldsymbol{x}, \boldsymbol{y}) \geq 0, (\rho(\boldsymbol{x}, \boldsymbol{y}) = 0 \leftrightarrow \boldsymbol{x} = \boldsymbol{y})$ (正値性)
  - $\rho(\boldsymbol{x}, \boldsymbol{y}) \leq \rho(\boldsymbol{x}, \boldsymbol{z}) + \rho(\boldsymbol{z}, \boldsymbol{y})$ (三角不等式)

内積の公理(厳密に)

- ベクトル空間の元$\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$間に下記の性質を満たす$\boldsymbol{x} \cdot \boldsymbol{y}$を決めることができるとき、これを内積と呼ぶ
  - 任意の実数a, bに対して$a\boldsymbol{x} + b\boldsymbol{y} \cdot \boldsymbol{z} = a(\boldsymbol{x} \cdot \boldsymbol{z}) + b(\boldsymbol{y} \cdot \boldsymbol{z})$(線形性)
  - $\boldsymbol{x} \cdot \boldsymbol{y} = \boldsymbol{y} \cdot \boldsymbol{x}$ (対称性)
  - $\boldsymbol{x} \cdot \boldsymbol{x} \geq 0,\ \boldsymbol{x} \cdot \boldsymbol{x} = 0$であれば$\boldsymbol{x} = \boldsymbol{0}$ (正値性)

[Application]Simple perceptron



$y = \sum_{i=1}^{n} w_i x_i$+b= $w_1 x_1 + w_2 x_2 + \cdots w_n x_n + b$

When $\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$, $\boldsymbol{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}$, the following equation is obtained:

$$\boldsymbol{y} = \boldsymbol{w} \cdot \boldsymbol{x} + b$$

The bias term is disturbing.

$$output = \begin{cases} -1 \ (if\ y < 0) \\ 1 \ (if\ y \geq 0) \end{cases}$$

個人課題3へ

グループワーク

1. ベクトル同士の距離、内積、またはコサイン類似度を使って実現できそうな応用例を考えてみよう
   - どういうベクトルを用意して、どのような演算、処理をするかを示すこと
   - 具体的なベクトルの例を挙げて実際に計算結果も示すこと
   - プログラミング結果も示すこと

- 各グループ発表5分、質疑応答2分

- グループワーク課題資料をGoogle Colaboratoryで提出する際は、その資料の表紙、および提出時のメッセージに、各メンバーがどの部分を貢献したかとその貢献度%(全体を100%として)を記述して提出してください

次週授業後半で各グループごとに発表

個人課題

1. YouTubeのprof.Gilbert StrangのLinear Algebraの授業 「1. The Geometry of Linear Equations」
   - https://youtu.be/J7DzL2_Na80 を視聴し、教科書p.62～p.106ページを参考にして、行列について予習した上で、要点をA4 1枚以内にまとめてください。

2. Lec02.ipynbの「入力画像ファイルから指定したディレクトリの中の画像から一番類似性が高い画像ファイル名を検索するプログラム」の部分の
   comp_sim(qvec,tvec)にqvecとtvecコサイン類似度を計算する
   プログラムを作成し、意図するプログラムを完成してください。

3. 単純パーセプトロンが$\boldsymbol{y} = \boldsymbol{w} \cdot \boldsymbol{x} + \boldsymbol{b}$が表せることがわかったが、バイアス項$(+\boldsymbol{b})$がついているのが美しくない。
   $\boldsymbol{y} = \boldsymbol{w} \cdot \boldsymbol{x}$と表現できるようにするためには、どのように式表現すればよいのか、A4枚以内に説明してください。

4. (加点対象)マンハッタン距離、ユークリッド距離、チェビシェフ距離を整理した上で<u>マハラノビス距離</u>を説明し、具体的な実用例をあげ、 A4 1枚にまとめてください。

- 1、3、4 についてはdocx又はpdfファイル、2についてはipynbファイルを提出すること(つまり、一人3ファイルは必ず提出すること)
- Google Classroomで提出のこと
  (締切はGoogle Classroom参照)