機械学習の回帰分析

2024年5 月22日

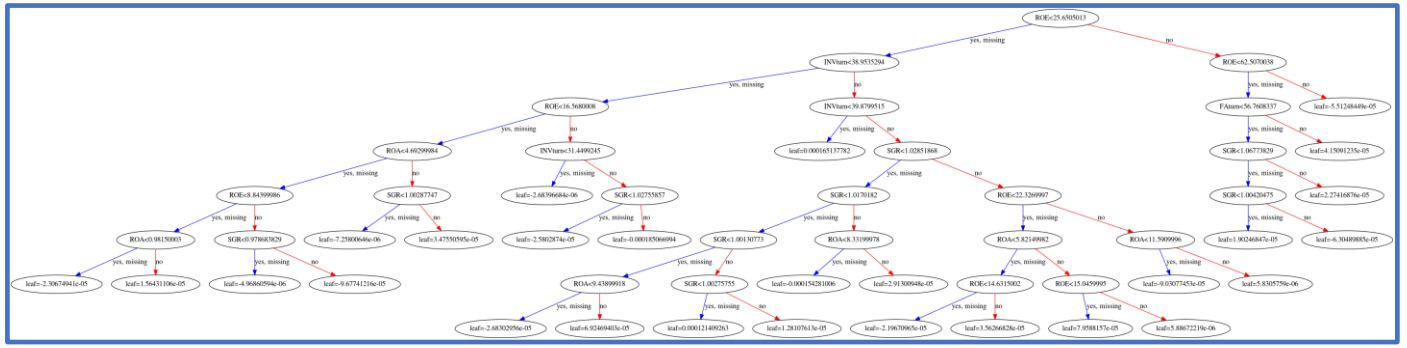
学習院大学経済学部経営学科教授

白田　由香利

機械学習による分析手法

- 回帰
  - 線形回帰
  - 樹木系AI手法
    - 決定木 decision tree
    - ランダムフォレスト
    - 勾配ブースティング系
      XGBOOST, LightGBM
  - ニューラルネットワーク

- 分類 classification
  - ロジスティック回帰
  - 樹木系AI手法
    - 決定木 decision tree
    - ランダムフォレスト
    - 勾配ブースティング系
      XGBOOST, LightGBM
  - ニューラルネットワーク



機械学習による分析手法

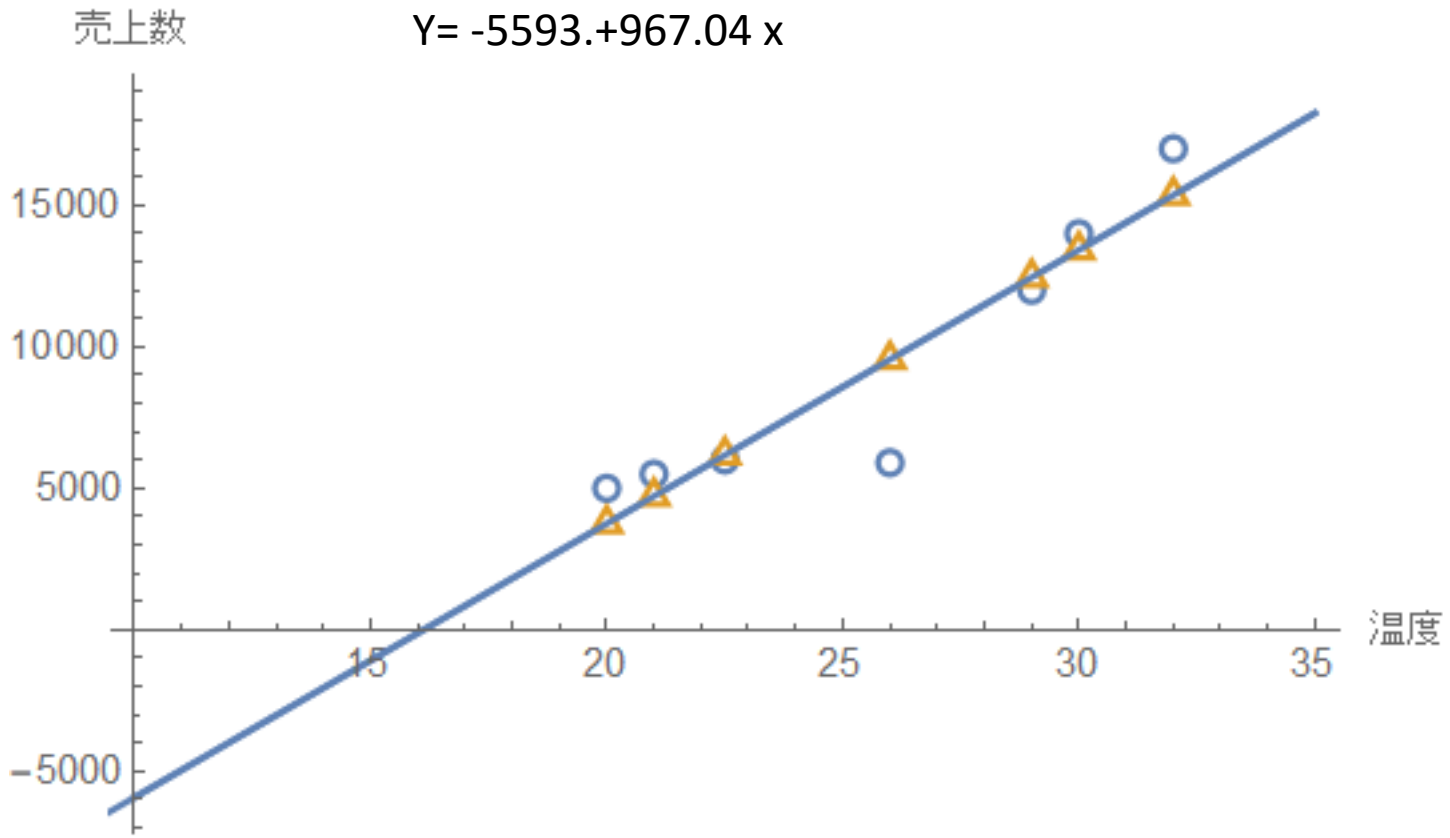
- クラスタリング
  - Kmeans
  - 階層型クラスタリング
  - 時系列データクラスタリング
    - Amplitude-based clustering
    - K-Shape
    - Dynamic Time Warp
- 次元圧縮
  - 主成分分析(PCA)
  - 特異値分解(SVD)
  - t-SNE
  - UMAP

大枠と代表的なモデル(手法）

- **回帰**(regression)　**教師あり**データ分析，数値と画像
  ニューラルネット，
  XGBoost モデル，ランダムフォレストモデル
- **分類**(classification) **教師あり**，数値と画像
  決定木，XGBoost モデル，ランダムフォレストモデル
- **クラスタリング**(clustering)　**教師無し**，数値と画像
  Kmeans法，Dynamic Time Warp法，
  階層化クラスタリング，ニューラルネット
- **次元圧縮**(dimensionality reduction)　数値
  PCA（主成分分析：Principal Component Analysis)

線形回帰：回帰モデルは直線
野球場でのビールの売上数の予測

- 観測値
- 線形回帰による予測値



線形回帰　　残差：観測値ー予測値

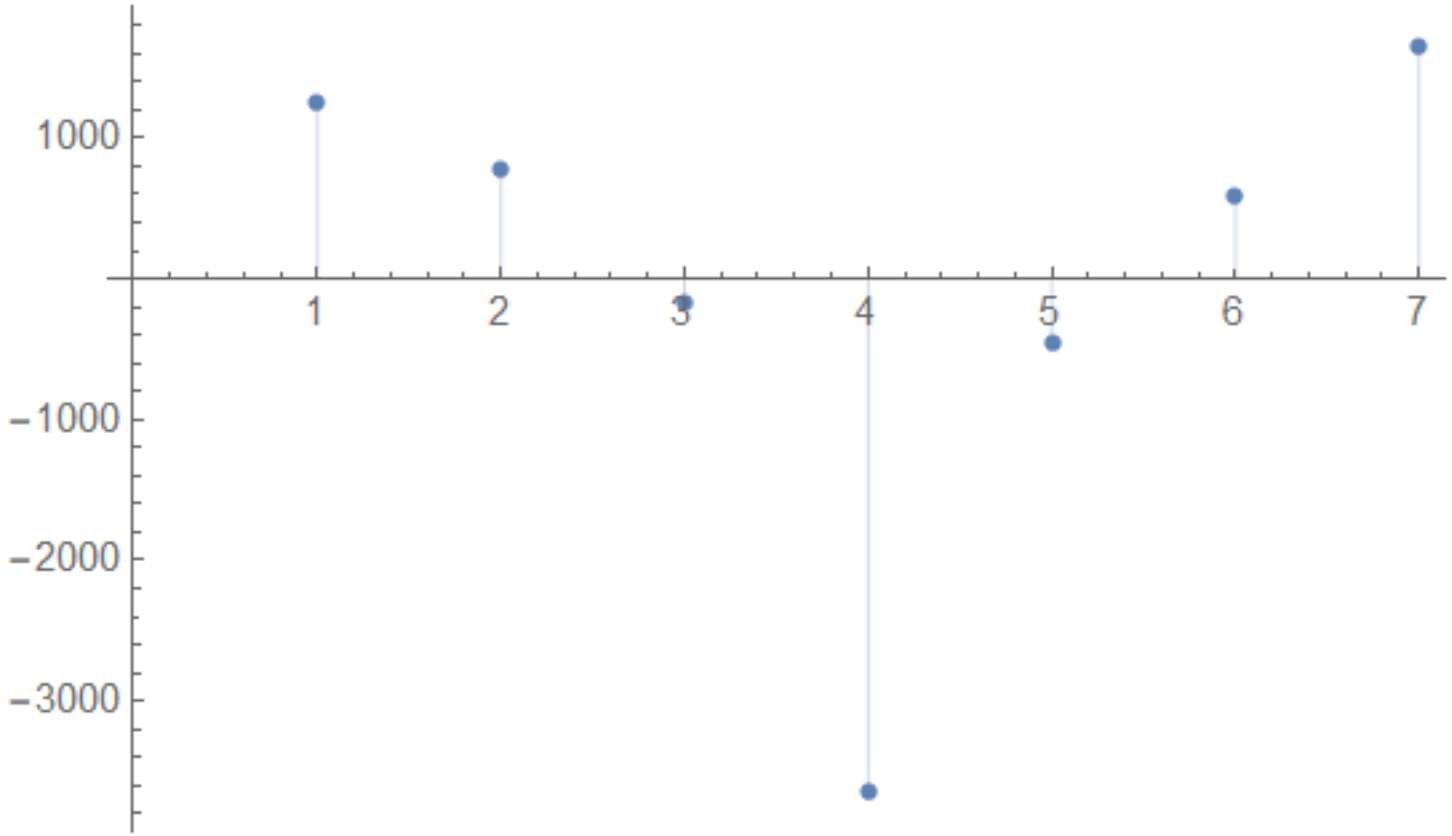
$$\mathrm{y_i} - (a + bx_i)$$

- 観測値 y_i
- 推定された回帰式

$$Y_i = a + bx_i$$

- 残差　$\mathrm{y_i} - (a + bx_i)$

$$\begin{pmatrix} 5000 - a - 20\,b \\ 5500 - a - 21\,b \\ 6000 - a - 22.5\,b \\ 5900 - a - 26\,b \\ 12\,000 - a - 29\,b \\ 14\,000 - a - 30\,b \\ 17\,000 - a - 32\,b \end{pmatrix}$$



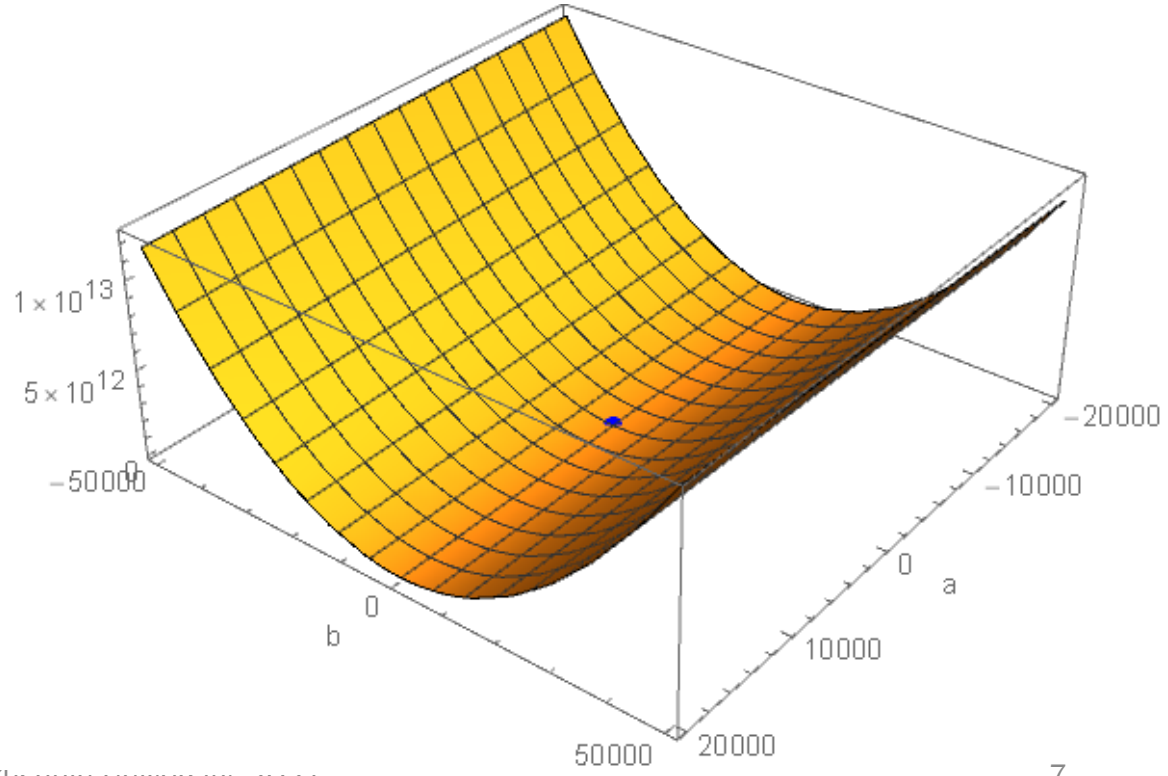
線形回帰　　最小二乗法
残差の平方和を最小にす a, bを求める

f(a,b)=
$(17000-a-32\,b)^2+(14000-a-30\,b)^2+(12000-a-29\,b)^2+(5900-a-26\,b)^2+(6000-a-22.5\,b)^2+(5500-a-21\,b)^2+(5000-a-20\,b)^2$

- $\frac{\partial f}{\partial a}=0, \quad \frac{\partial f}{\partial b}=0$

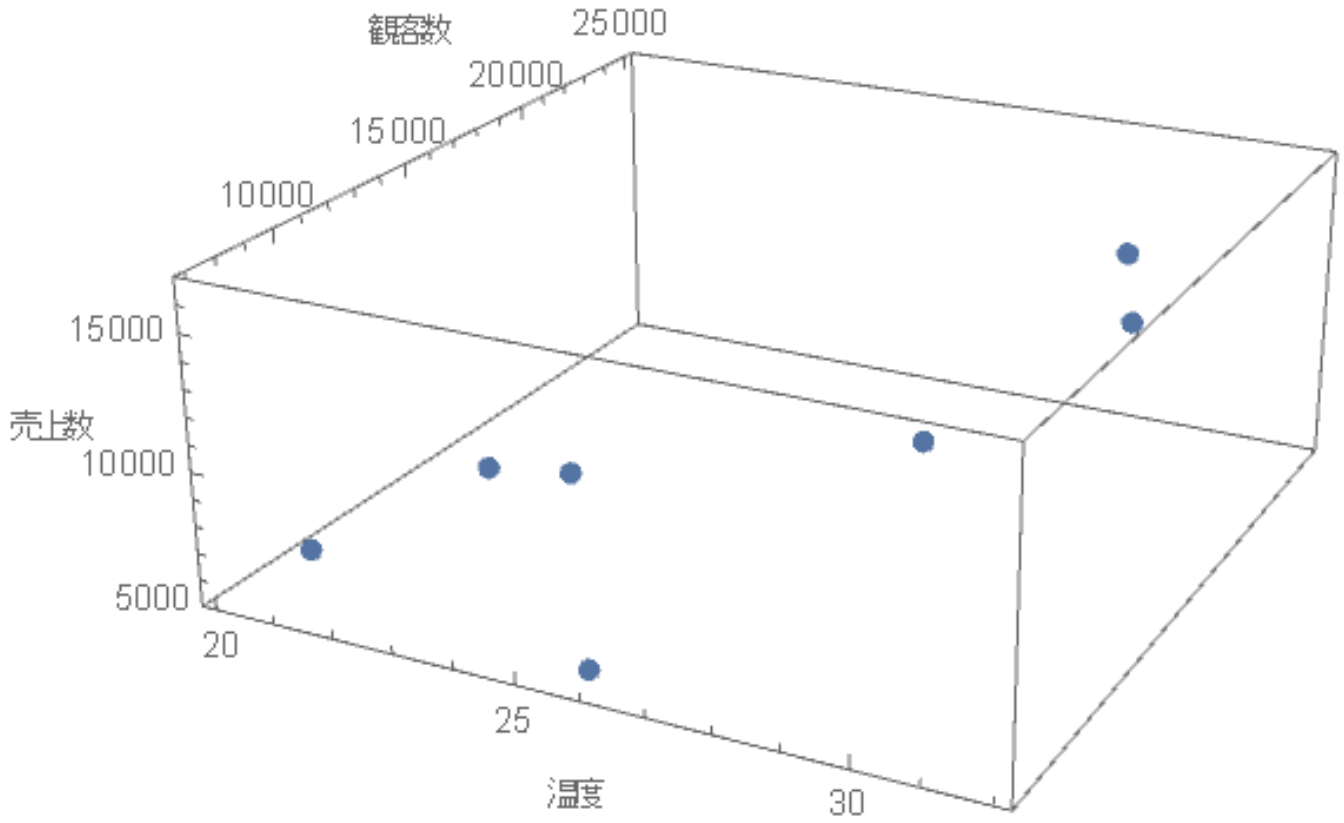
偏微分して
この2元連立方程式を解く

- {a= -15593., b= 967.04}
- **回帰式**$Y_i=-15593+967.04x_i$



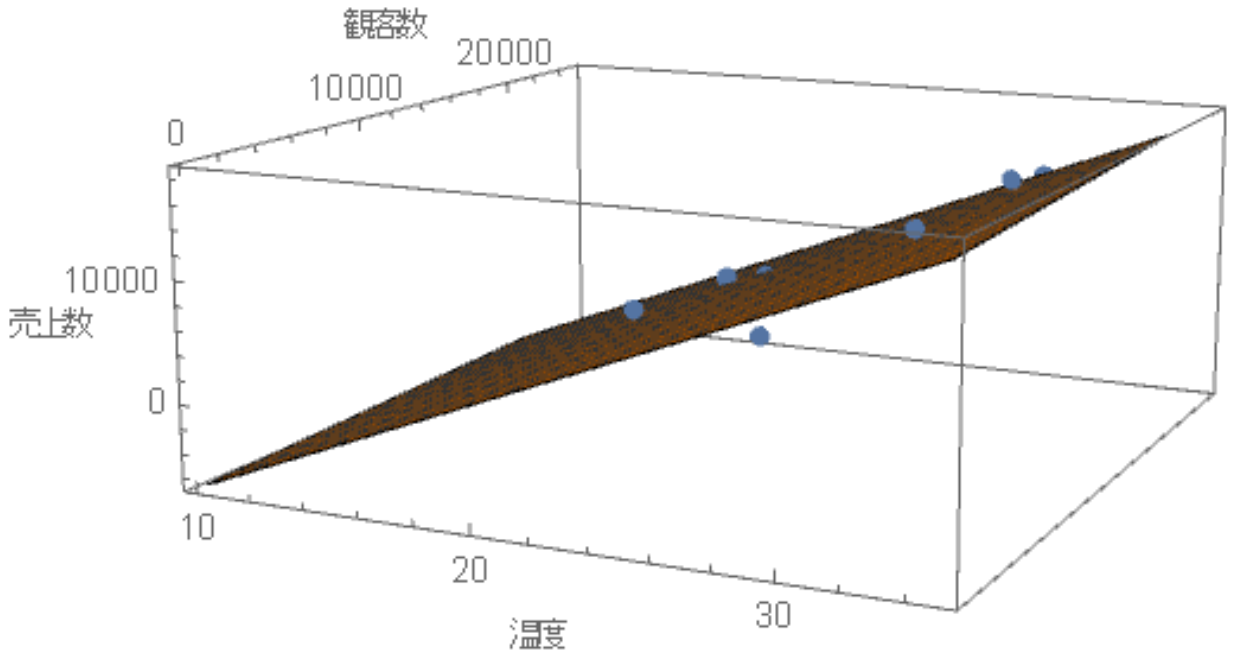
重回帰分析　２説明変数の場合

- 温度、観客数

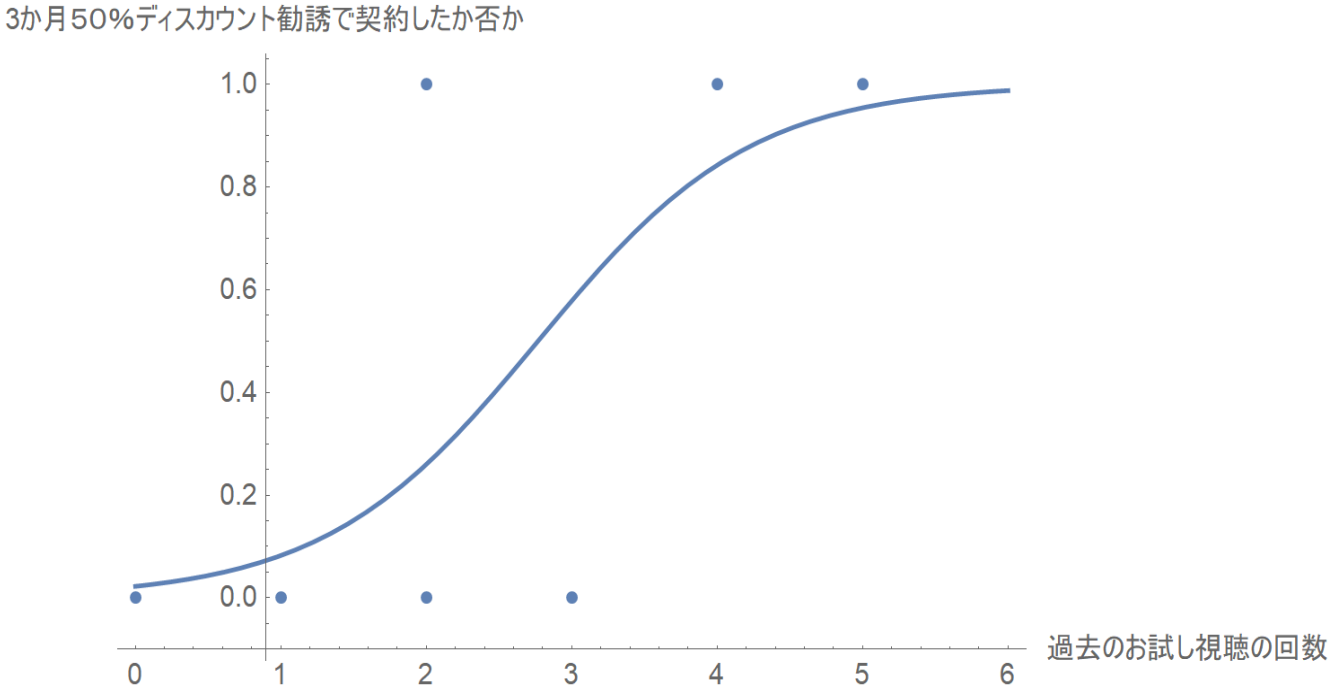


線形回帰：回帰モデルは平面
重回帰分析　２説明変数の場合

- 温度 x
- 観客数 y
- f(x,y)=-16119.9+955.762 x+0.0603841 y



分類：ロジスティック回帰
過去のお試し視聴の回数から契約するか否か予測



- 1変数によるモデル

$$\frac{1}{1+e^{-(a+bx)}}$$

$$\frac{1}{1+e^{-(-1.006+1.172x)}}$$

ロジスティック回帰は数学ツールにやってもらう。自分でa, bを解かなくてよい。

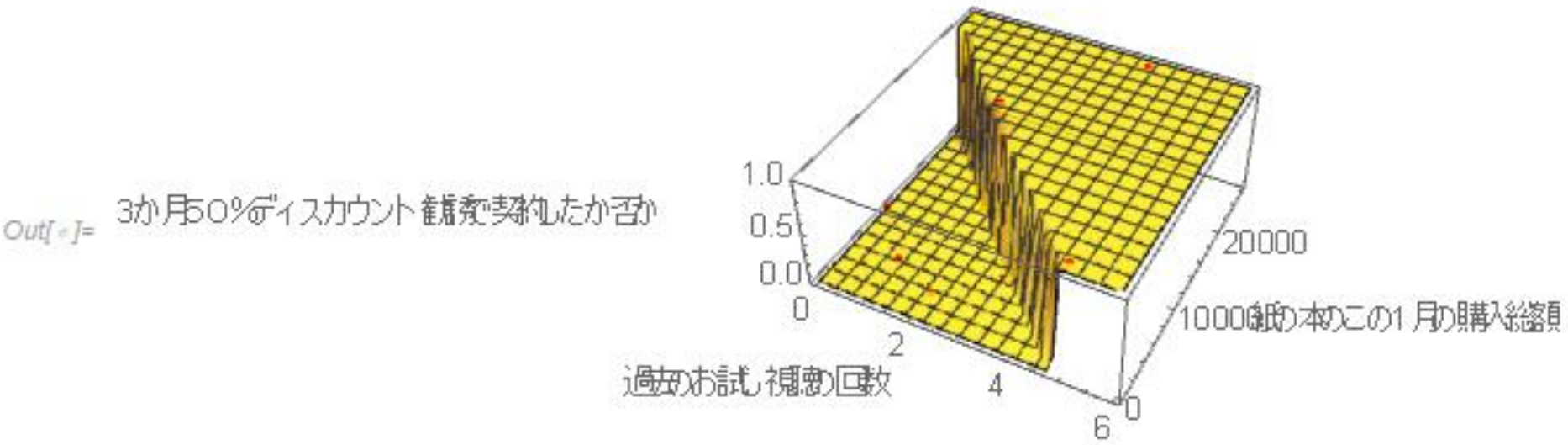
2個の説明変数でのロジスティック回帰

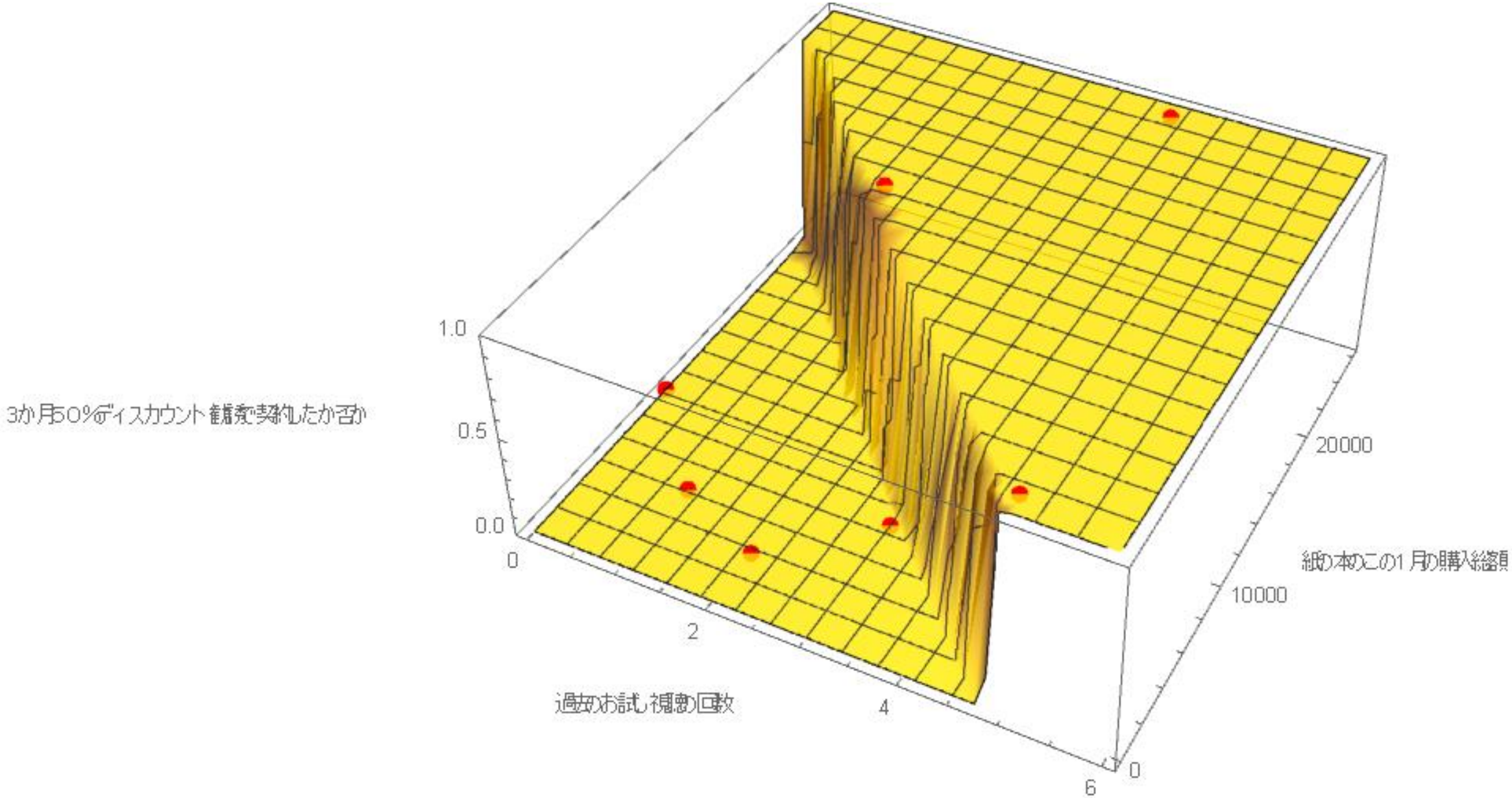
- 2変数によるモデル

$$\frac{1}{1+e^{-(a+bx+cy)}}$$

- $$\frac{1}{1+e^{271.424-55.957\,x-0.0119927\,y}}$$

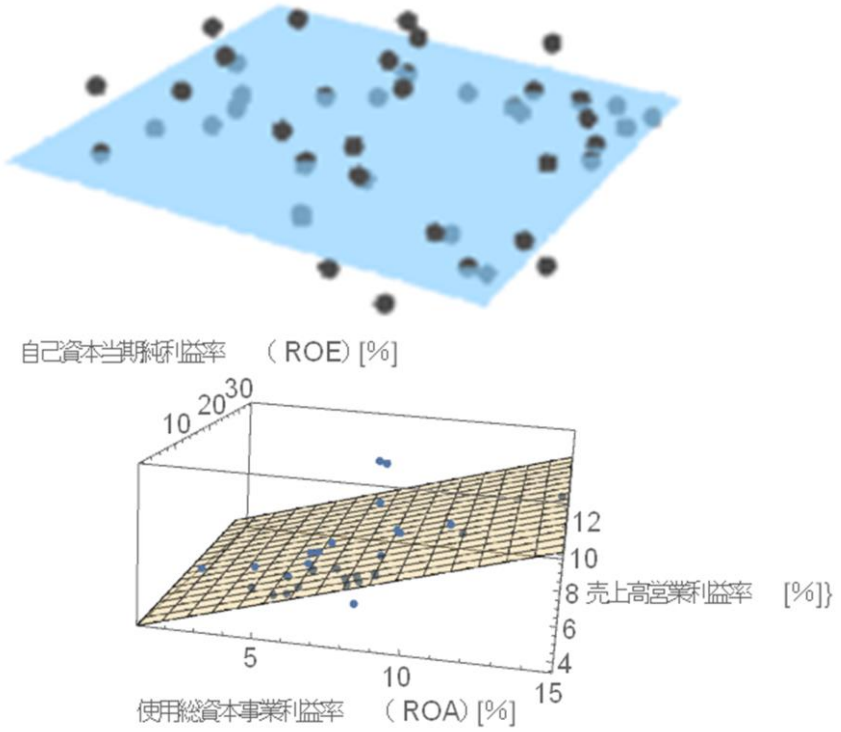
Out[ ]=

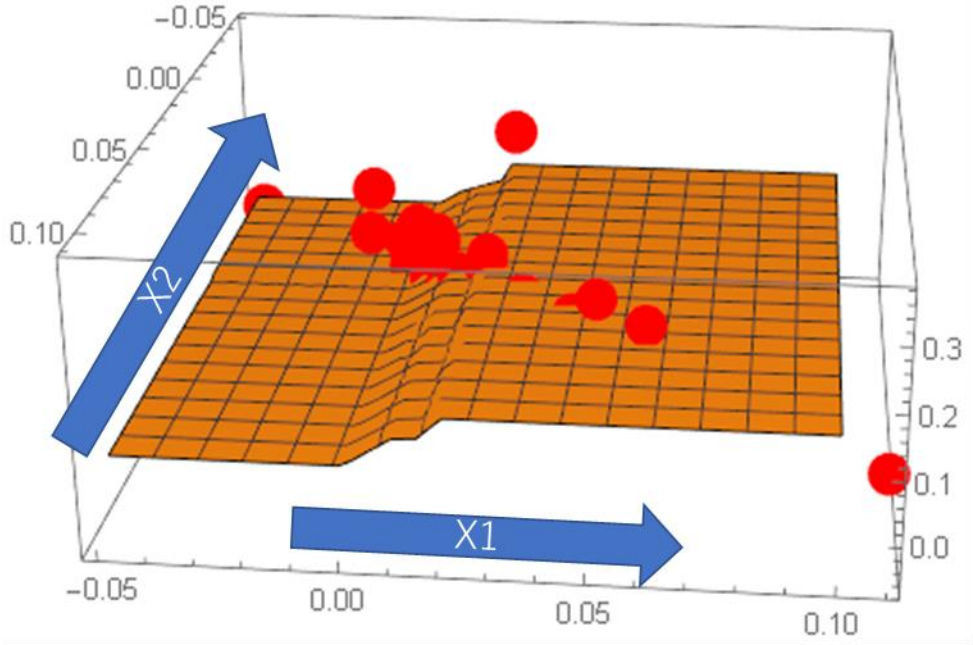




機械学習による回帰分析

- 線形モデルを仮定しない。
  複雑な形状のモデルとなる。





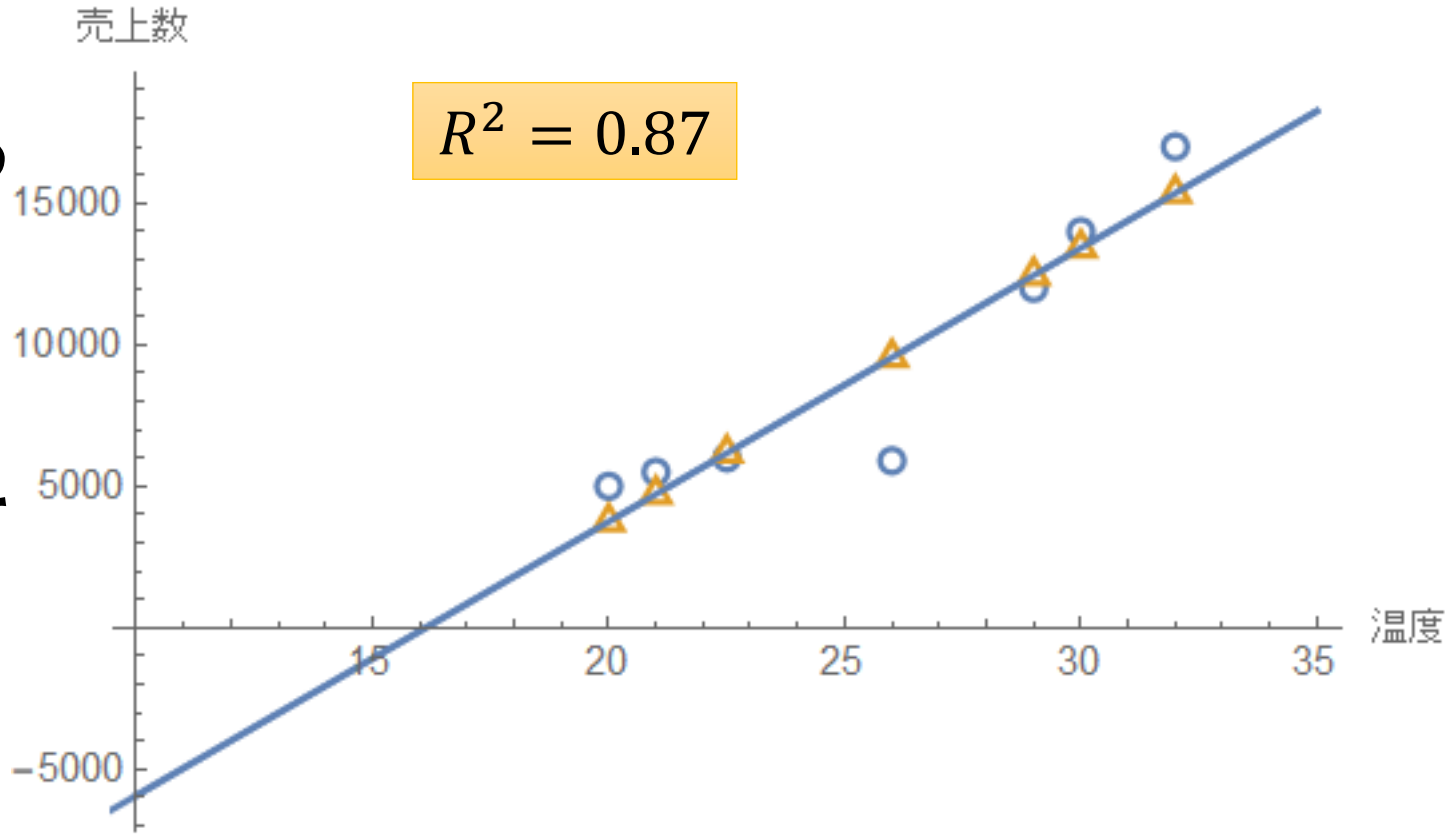
線形回帰に適している例

線形回帰に適さない例
傾向の異なる２つのドメイン

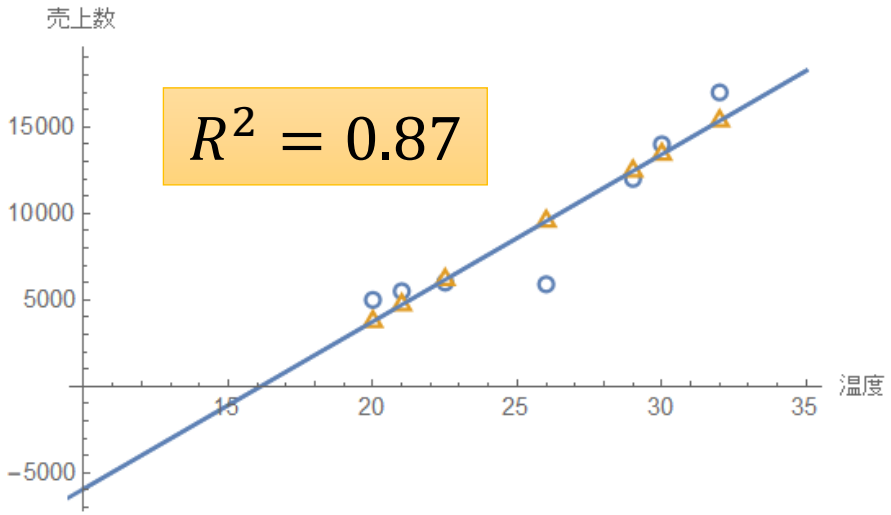
線形回帰に適さない例
傾向の異なる５つのアイランド領域

決定係数
フィッティングの良さ

- １に近いほど良い
- ０から１の値をとる
- 回帰分析では
  0.7以上はほしい
- 例えば、0.3では
  回帰が意味をなして
  いない



決定係数
フィッティングの良さ



- $R^2 = \dfrac{(\text{予測値で説明された変動})}{(\text{偏差の平方和})}$
- 恒等式
  $\sum_{i=1}^{n}(y_i - \bar{y})^2 = \sum_{i=1}^{n}(Y_i - \bar{y})^2 + \sum_{i=1}^{n}(y_i - Y_i)^2$
  偏差の平方和＝予測値で説明された変動＋残差の平方和

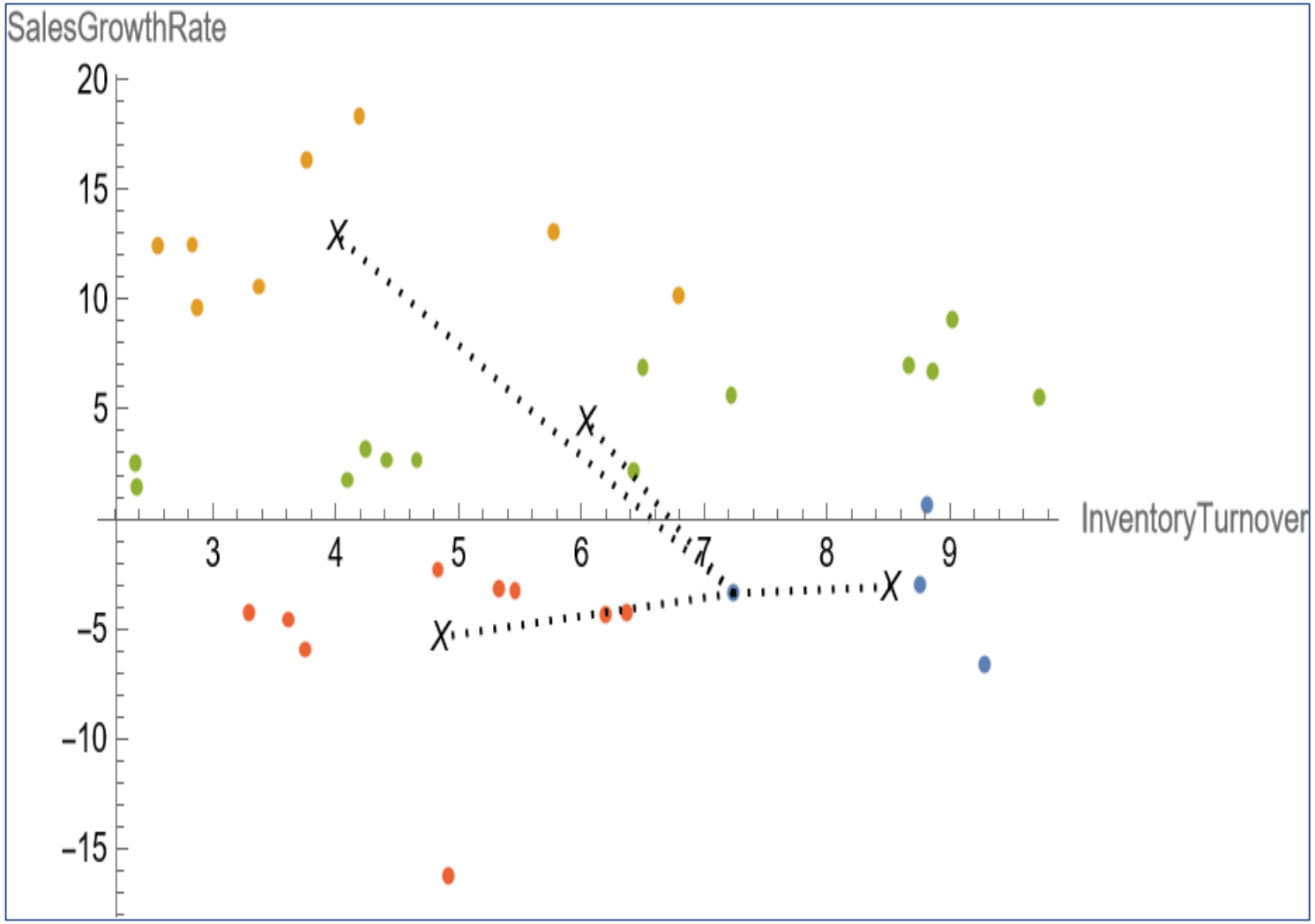
偏差とは平均$\bar{y}$からのずれ

観測値の平均と、予測値の平均はいつでも一致する

クラスタリング

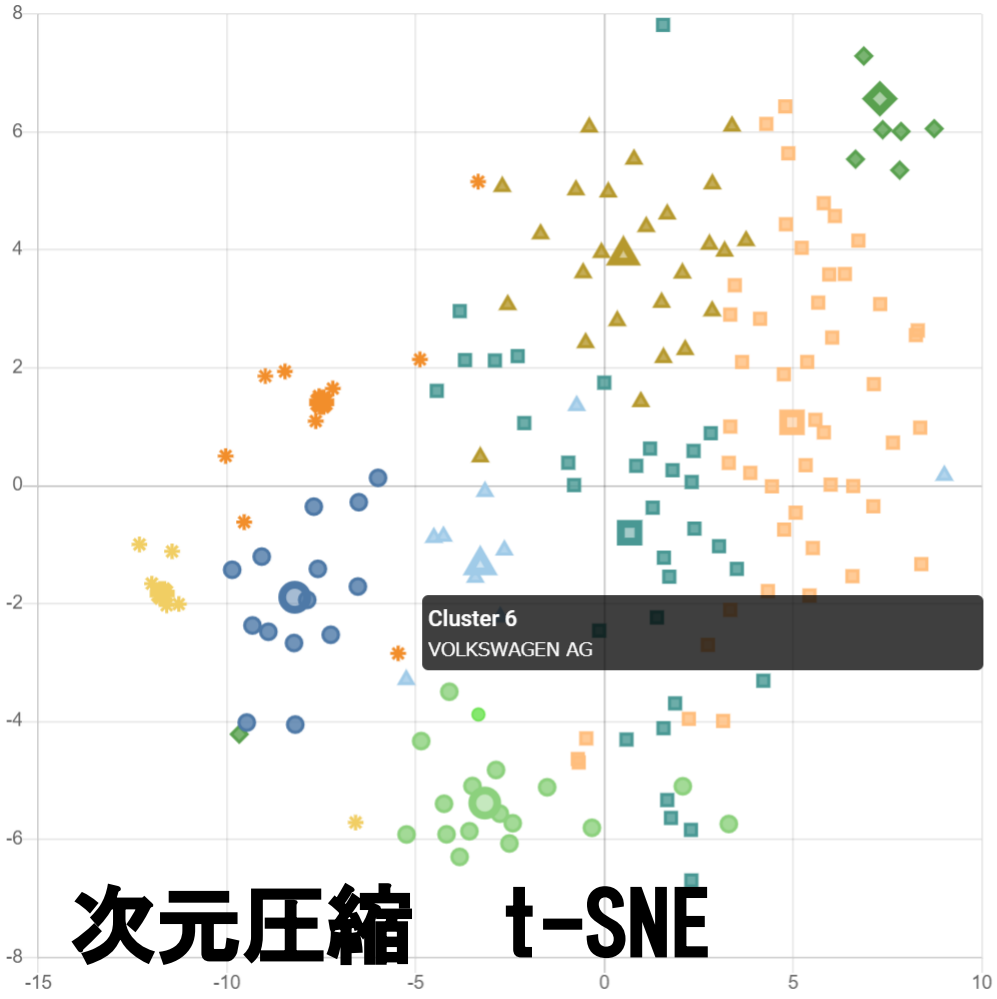
- K-means法
- **階層型クラスタリング**

K-means法



K-Means with Euclidean Distance
k=9

クラスタリング→　次元圧縮　という流れが一般的



次元圧縮　t-SNE

● Cluster1　▲ Cluster2　✶ Cluster3　■ Cluster4　◆ Cluster5　● Cluster6
▲ Cluster7　✶ Cluster8　■ Cluster9

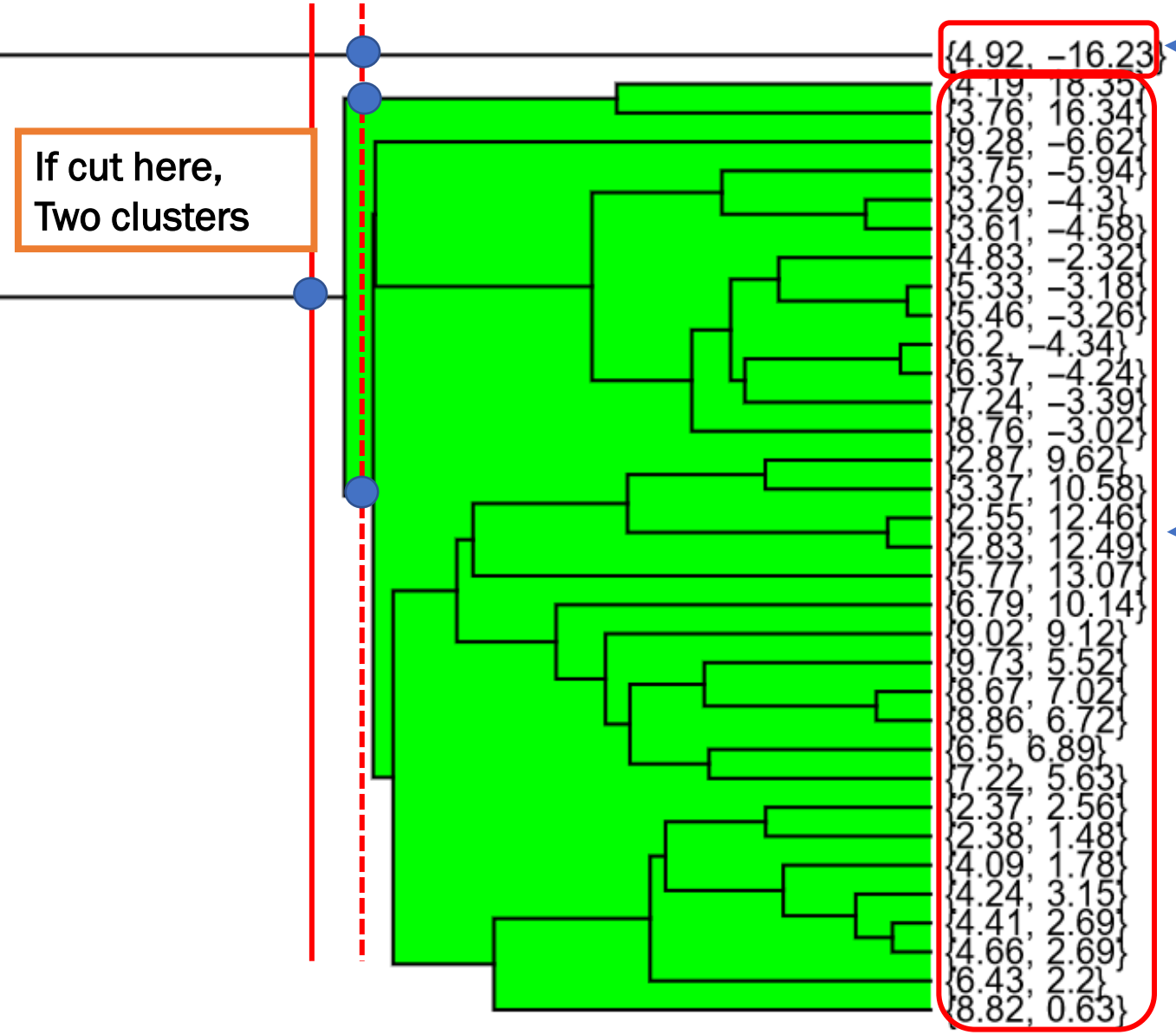
original

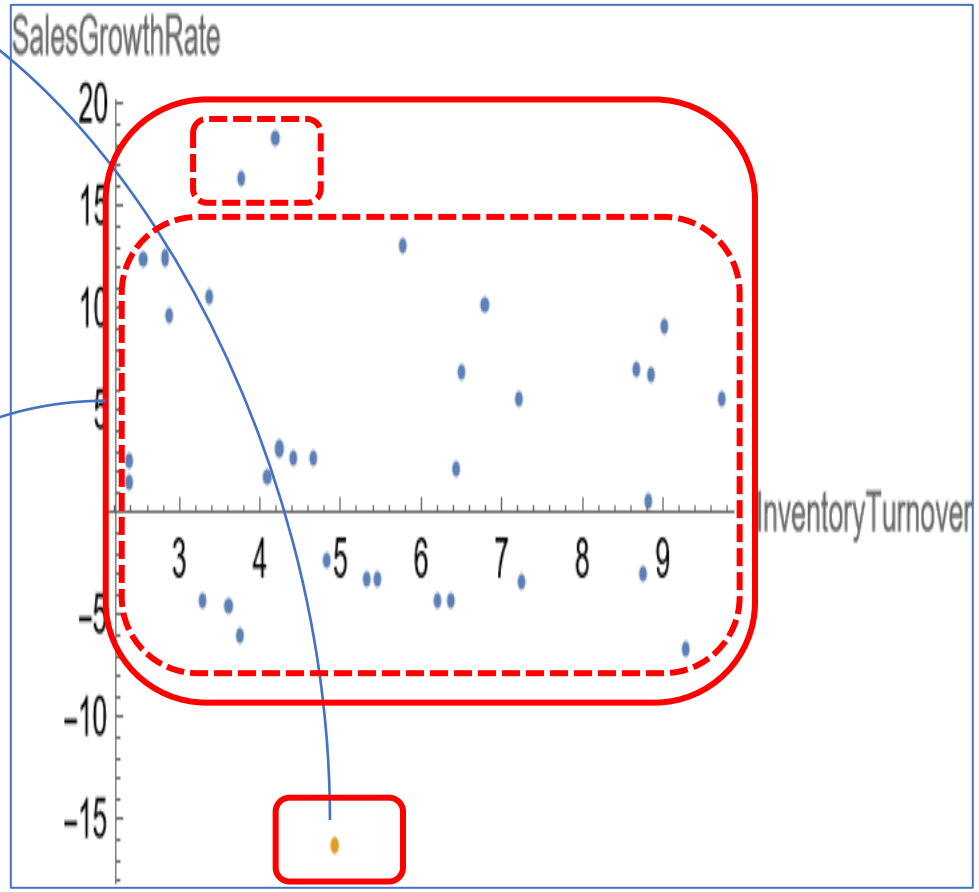
● VOLKSWAGEN AG　■ TOYOTA MOTOR CORPORATION　■ DAIMLER AG　■ FORD MOTOR CO　▲ GENERAL MOTORS COMPANY　■ SAIC MOTOR CORPORATION LIMITED　■ FIAT CHRYSLER AUTOMOBILES N.V.　■ BAYERISCHE MOTOREN WERKE AKTIENGESELLSCHAFT　■ NISSAN MOTOR CO LTD　■ HYUNDAI MOTOR CO.LTD.　✶ PEUGEOT S.A.　✶ AUDI AG　■ RENAULT　● KIA MOTORS CORPORATION　◆ DENSO CORPORATION　▲ AB VOLVO　■ TATA MOTORS LIMITED　■ MAGNA INTERNATIONAL INC　✶ HONEYWELL INTERNATIONAL INC　■ AISIN SEIKI CO LTD　▲ SUZUKI MOTOR CORPORATION　■ HYUNDAI MOBIS CO.LTD.　■ MAZDA MOTOR CORPORATION　▲ SUBARU CORPORATION　● PACCAR INC　✶ TESLA INC.　▲ MITSUBISHI MOTORS CORPORATION　■ VALEO SA　■ FAURECIA　▲ TOYOTA INDUSTRIES CORPORATION　■ LEAR CORP　◆ ISUZU MOTORS LIMITED　◆ HINO MOTORS LTD　■ TENNECO INC.　✶ PT ASTRA INTERNATIONAL TBK　■ SCHAEFFLER AG　◆ MITSUBISHI MATERIALS CORPORATION　■ APTIV PLC　▲ MAHINDRA & MAHINDRA LIMITED　■ TOYOTA BOSHOKU CORPORATION　▲ MARUTI SUZUKI INDIA LIMITED　■ NAVISTAR INTERNATIONAL CORP　▲ GESTAMP AUTOMOCION S.A.　■ BORGWARNER INC　▲ CHONGQING CHANGAN AUTOMOBILE COMPANY LIMITED　■ DANA INCORPORATED　■ AUTOLIV INC.　● OSHKOSH CORPORATION　▲ THOR INDUSTRIES INC　■ KOITO MANUFACTURING CO LTD　■ ANHUI JIANGHUAI AUTOMOBILE GROUP CORP. LTD.　■ RHEINMETALL AG　■ HYUNDAI WIA CORPORATION　▲ FORD OTOMOTIV SANAYI A.S.　✶ AMERICAN AXLE & MANUFACTURING HOLDINGS INC.　■ NOK CORPORATION　● HOTAI MOTOR CO. LTD　■ LINAMAR CORPORATION　● NISSAN SHATAI CO LTD　✶ HANON SYSTEMS CORP.　■ MANDO CORPORATION　■ ZHENGZHOU YUTONG BUS CO. LTD.　▲ TOKAI RIKA CO. LTD.　■ ASHOK LEYLAND LIMITED　● MERITOR INC.　■ DELPHI TECHNOLOGIES PLC　■ TEREX CORP　■ FUTABA INDUSTRIAL CO LTD　■ NEMAK S.A.B. DE C.V.　■ JIANGLING MOTORS CORPORATION LIMITED　✶ MITSUBISHI LOGISNEXT CO. LTD.　● FERRARI N.V.　■ TRELLEBORG AB　▲ NGK SPARK PLUG CO LTD　■ TS TECH CO. LTD.　● FAW CAR CO. LTD.　● WABCO HOLDINGS INC.　● SSANGYONG MOTOR CO.LTD.　● HYSTER-YALE MATERIALS HANDLING INC.　▲ KEIHIN

Nearest　Centerを探す

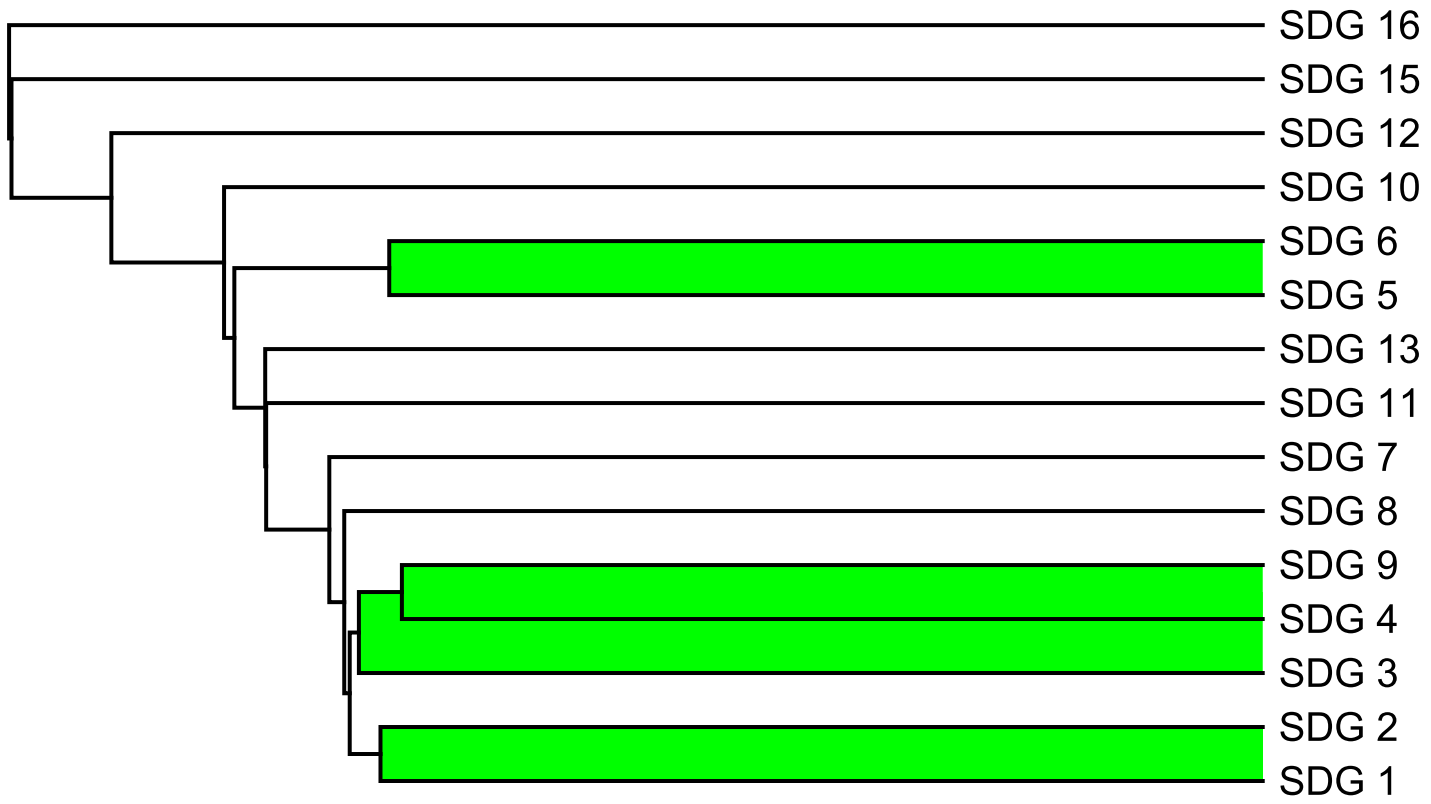
Name	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	Cluster 6	Cluster 7	Cluster 8	Cluster 9
NINGBO HUAXIANG ELECTRONIC CO. LTD.									
FAW CAR CO. LTD.									
DONGFENG AUTOMOBILE CO. LTD.									
WANXIANG QIANCHAO CO. LTD.									
LINGYUN INDUSTRIAL CORPORATION LIMITED.									
CHANGCHUN FAWAY AUTOMOBILE COMPONENTS COMPANY LIMITED									
SHANGHAI JIAO YUN GROUP CO. LTD.									
DIMA HOLDINGS CO. LTD.									
HANGCHA GROUP COMPANY LIMITED									
WUHU SHUNRONG SANQI INTERACTIVE ENTERTAINMENT NETWORK TECHNOLOGY CO. LTD.									
RHEINMETALL AG									
LIFAN INDUSTRY (GROUP) COMPANY LIMITED									
ANHUI ZHONGDING SEALING PARTS CO. LTD.									
VOLKSWAGEN AG									
SUNGWOO HITECH CO.LTD.									
ANHUI JIANGHUAI AUTOMOBILE GROUP CORP LTD									

階層型クラスタリング


If cut here,
Two clusters
{4.92, −16.23}
{4.19, 18.35}
{3.76, 16.34}
{9.28, −6.62}
{3.75, −5.94}
{3.29, −4.3}
{3.61, −4.58}
{4.83, −2.32}
{5.33, −3.18}
{5.46, −3.26}
{6.2, −4.34}
{6.37, −4.24}
{7.24, −3.39}
{8.76, −3.02}
{2.87, 9.62}
{3.37, 10.58}
{2.55, 12.46}
{2.83, 12.49}
{5.77, 13.07}
{6.79, 10.14}
{9.02, 9.12}
{9.73, 5.52}
{8.67, 7.02}
{8.86, 6.72}
{6.5, 6.89}
{7.22, 5.63}
{2.37, 2.56}
{2.38, 1.48}
{4.09, 1.78}
{4.24, 3.15}
{4.41, 2.69}
{4.66, 2.69}
{6.43, 2.2}
{8.82, 0.63}

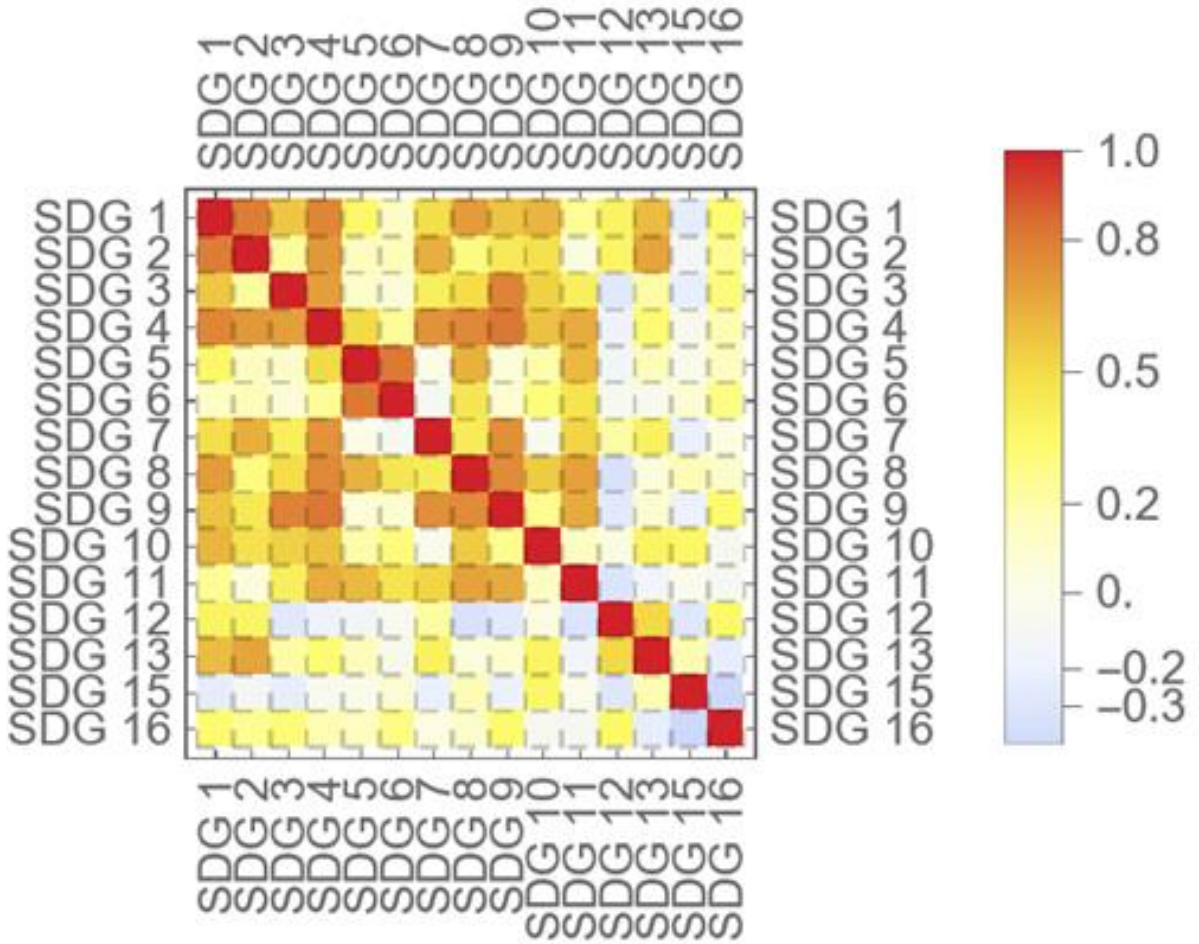

SalesGrowthRate
20
15
10
5
-5
-10
-15
3
4
5
6
7
8
9
InventoryTurnover

階層型クラスタリングの例
インド2020年のSDGスコアの階層型クラスタリングの結果



事例による検証
インドSDG州別スコア2020年

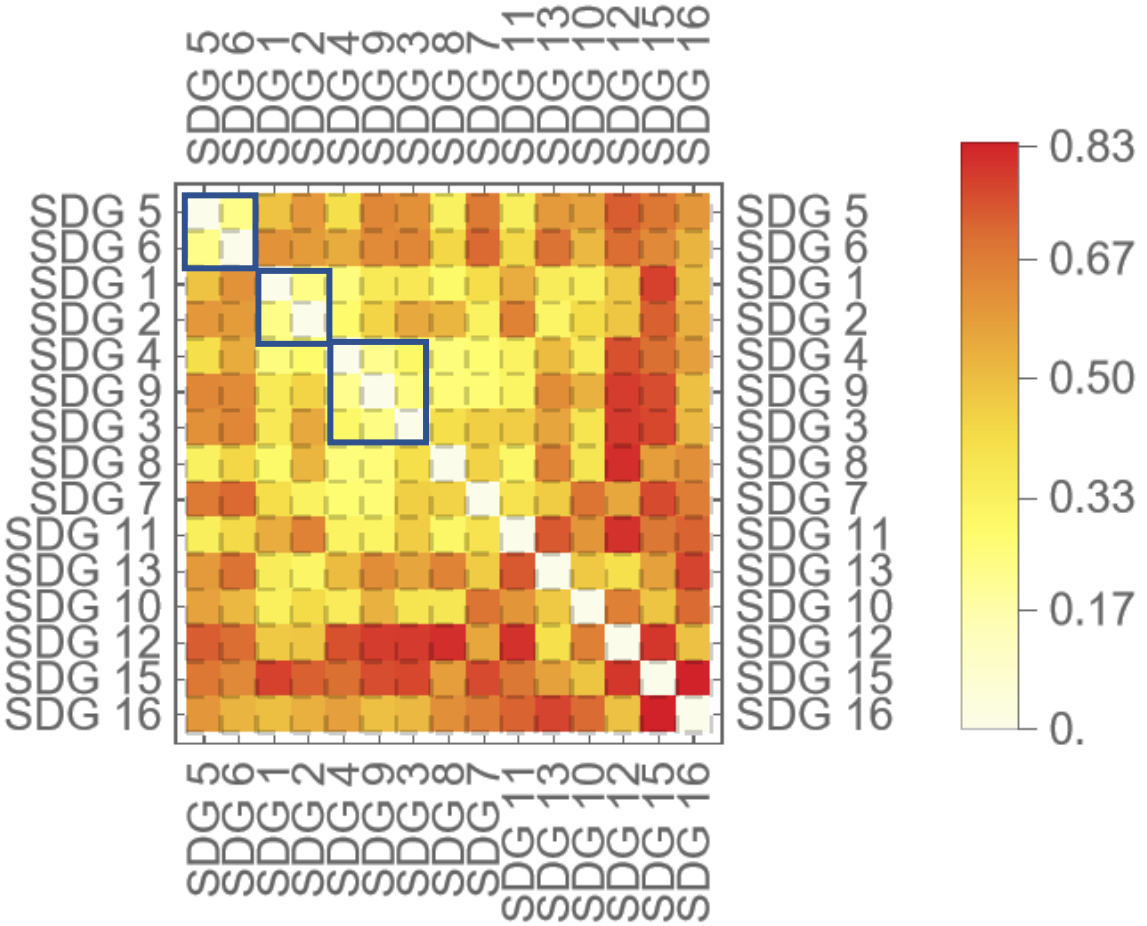
- 相関行列



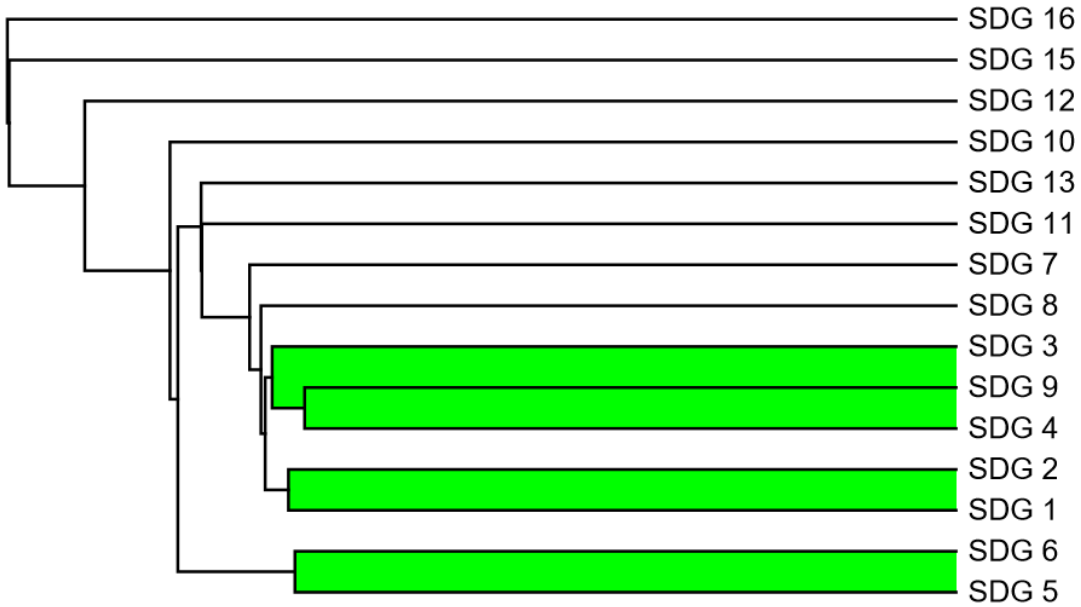
事例による検証
インドSDG州別スコア2020年

- クラスタリングと比較
- 階層型クラスタリング
- 距離 $d_{xy} = \sqrt{(\frac{1-r_{xy}}{2})}$

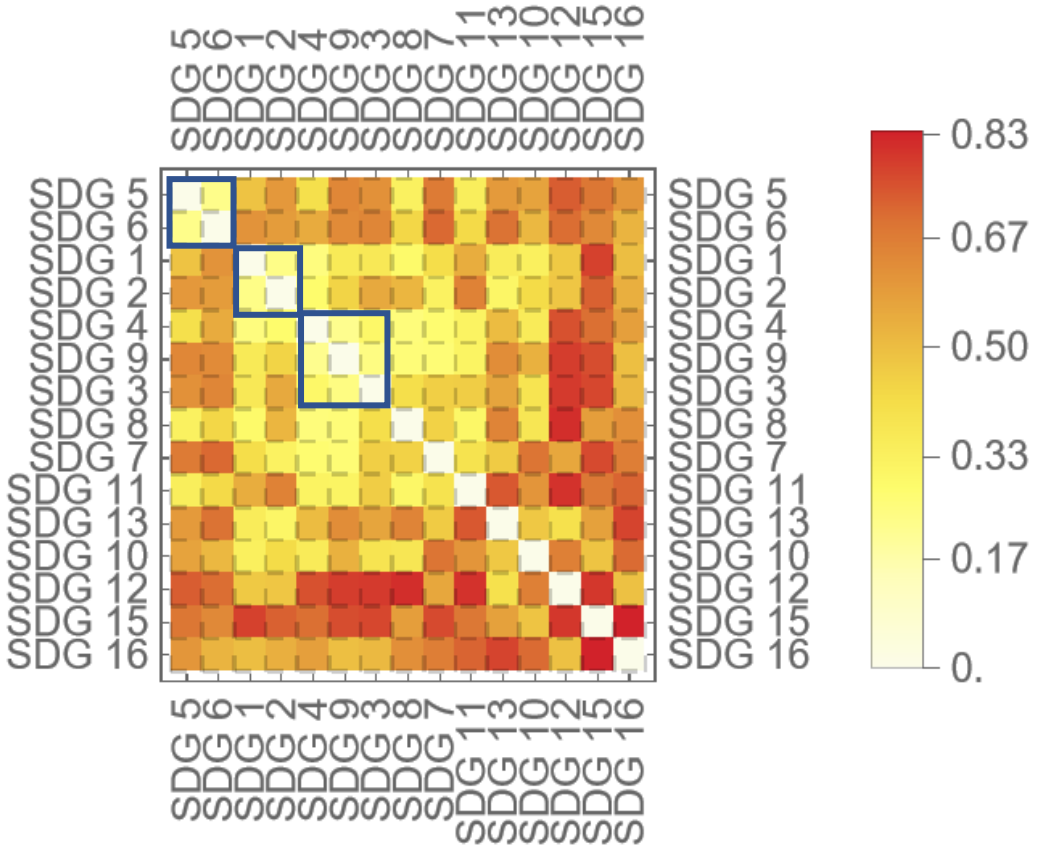
相関係数から距離を作成するよく使う手法



階層クラスタリングによるデンドログラムおよび距離行列(准対角化後）



- 対角線上に近いもの



AIによる回帰
ニューラルネット

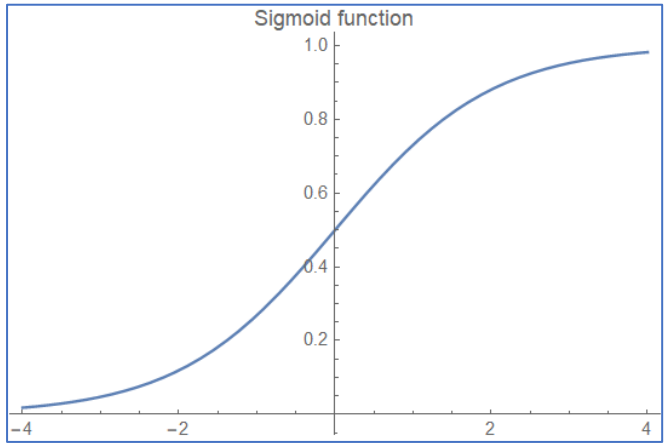
人間の脳の神経細胞（ニューロン）を模倣して作られた計算モデル
多層の人工ニューロンから構成される
ディープラーニングの基盤

ニューロン層に活性化関数をはさむ

- 典型的NNは
  線形変換⇒活性化関数⇒
  線形変換⇒活性化関数⇒
  線形変換
- 活性化関数
  ネットワークの非線形性を導入する役割を持ち、これにより複雑なデータパターンを学習することが可能

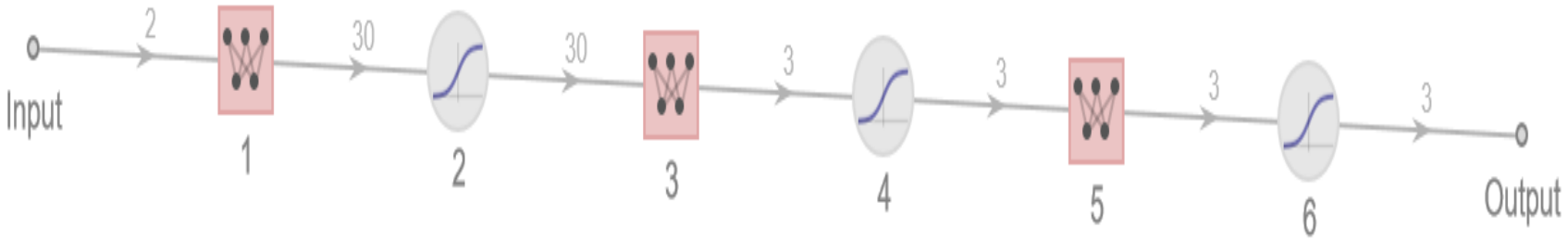
Out[7]= NetChain [uninitialized]

	Input	array
1	LinearLayer	vector (size: 10)
2	Ramp	vector (size: 10)
3	LinearLayer	vector (size: 20)
4	Ramp	vector (size: 20)
5	LinearLayer	vector (size: 10)
6	LogisticSigmoid	vector (size: 10)
7	LinearLayer	vector (size: 3)
	Output	vector (size: 3)



ニューラルネットの例

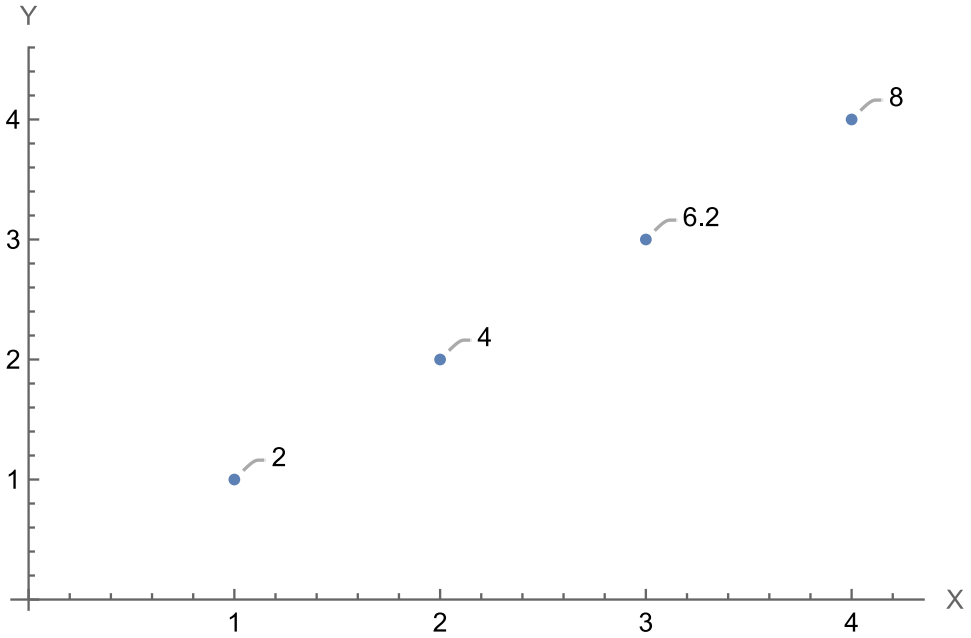
1. LinearLayer
2. Sigmoid
3. LinearLayer
4. Sigmoid
5. LinearLayer
6. Sigmoid



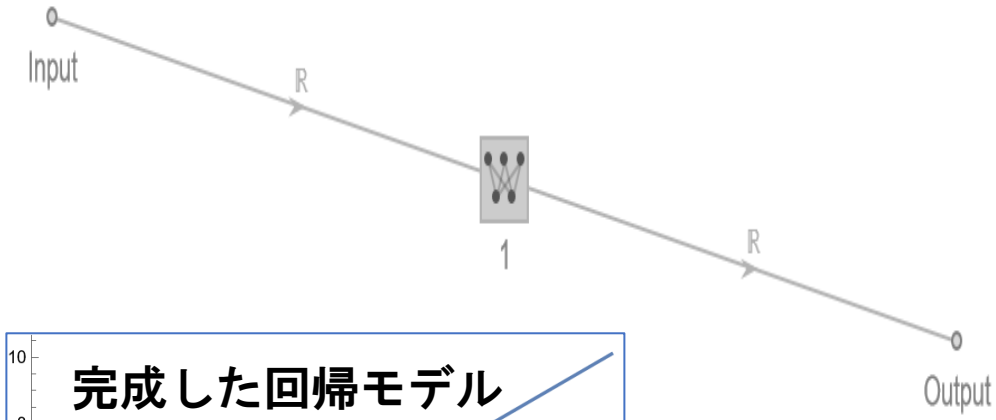
線形層で回帰：入力１→　出力１

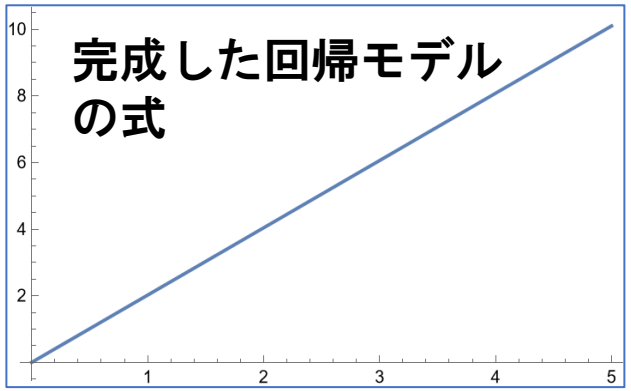
$$y_i = ax_i + b$$

- 以下のような４つの観測値



単一の線形層だけの非常にシンプルなNNの例





Linear Layer 線形層

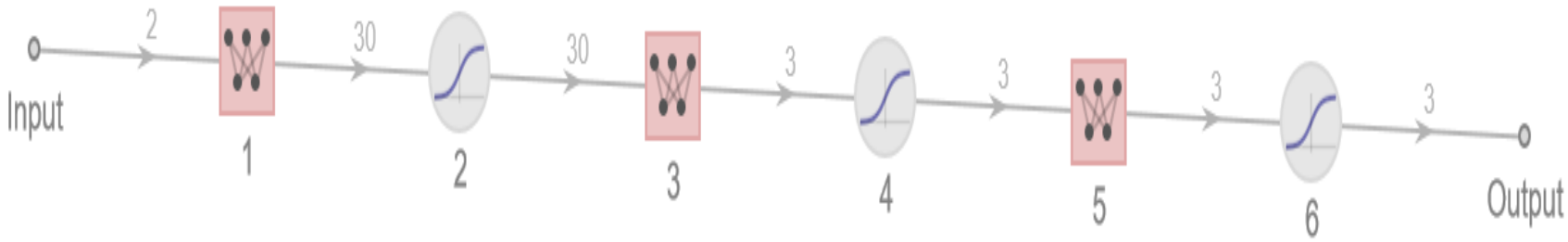
- 線形層の出力 y

$$y = x W + b$$

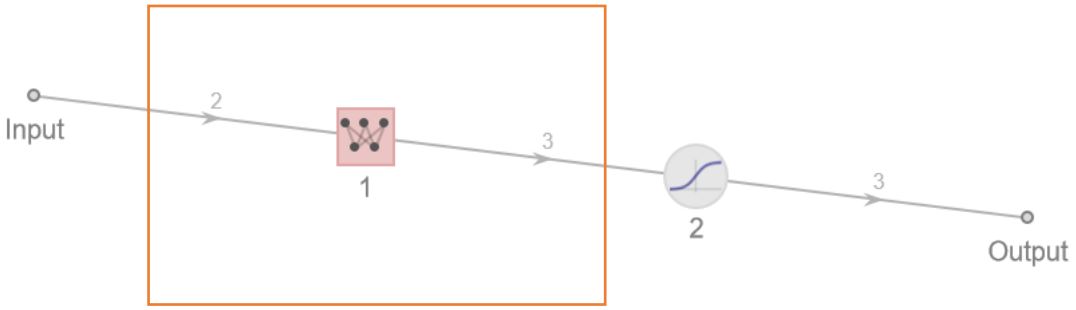
- x は入力ベクトル
- W は重み行列
- b はバイアスベクトル
- y は出力ベクトル

サイズの拡大縮小

- 入力が高次元の場合、次元削減して計算コストを削減
- 特徴を拡大して次の層で詳細に学習させる



ベクトルサイズ　2→3に拡大



- 2行3列の変換行列

$$\begin{pmatrix} w[1,1] & w[1,2] & w[1,3] \\ w[2,1] & w[2,2] & w[2,3] \end{pmatrix}$$

$$(x[1,1] \quad x[1,2]) \begin{pmatrix} w[1,1] & w[1,2] & w[1,3] \\ w[2,1] & w[2,2] & w[2,3] \end{pmatrix}$$

$$= (w[1,1]x[1,1] + w[2,1]x[1,2] \quad w[1,2]x[1,1] + w[2,2]x[1,2] \quad w[1,3]x[1,1] + w[2,3]x[1,2])$$

行列の掛け算

$$\begin{pmatrix} w[1,1] & w[1,2] & w[1,3] \\ w[2,1] & w[2,2] & w[2,3] \end{pmatrix} \cdot \begin{pmatrix} s[1,1] \\ s[2,1] \\ s[3,1] \end{pmatrix} = \begin{pmatrix} s[1,1]w[1,1] + s[2,1]w[1,2] + s[3,1]w[1,3] \\ s[1,1]w[2,1] + s[2,1]w[2,2] + s[3,1]w[2,3] \end{pmatrix}$$

行列の掛け算のサイズ

$$[x, y]\begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \end{bmatrix} + [b_1\ b_2\ b_3\ ]$$

＝

[b1+w11 x+w21 y　b2+w12 x+w22 y　b3+w13 x+w23 y　]

w11 x+w21 y ＋b1　このような計算が行列で表現されている

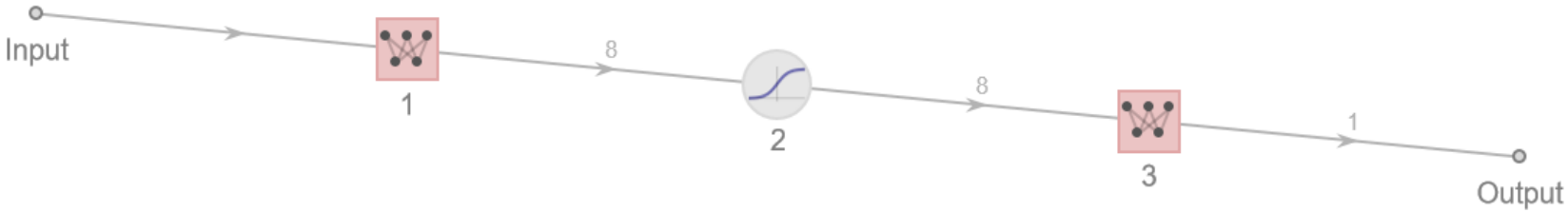
(1行×2列）×（2行×10列）＋（1行×10列）＝(１行×10列）

Report＃８　11-11
x, yを求めよ→小テスト

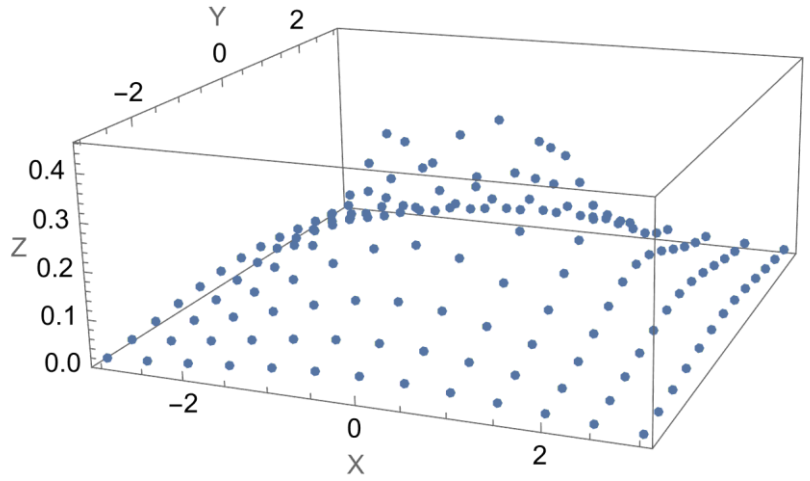
$$C = \begin{pmatrix} 7 & 4 & -1 \\ 3 & 0 & 5 \end{pmatrix}, D = \begin{pmatrix} 5 & 2 \\ 3 & 1 \\ 4 & -1 \end{pmatrix}, E = \begin{pmatrix} 8 & 4 & 2 \\ 1 & 3 & -6 \\ -7 & 0 & 5 \end{pmatrix}$$

$$\begin{bmatrix} x \\ y \end{bmatrix} = C \times E \times D \times \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

線形層で回帰：入力２→　出力１



- 与えられた観測値の集合



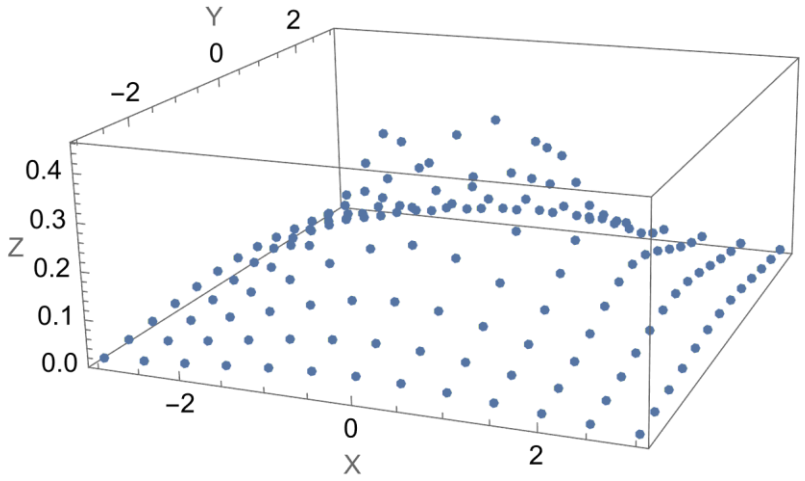
ニューラルネットモデルの例

- LinearLayer　2→8
- Sigmoid
- LinearLayer　8→1

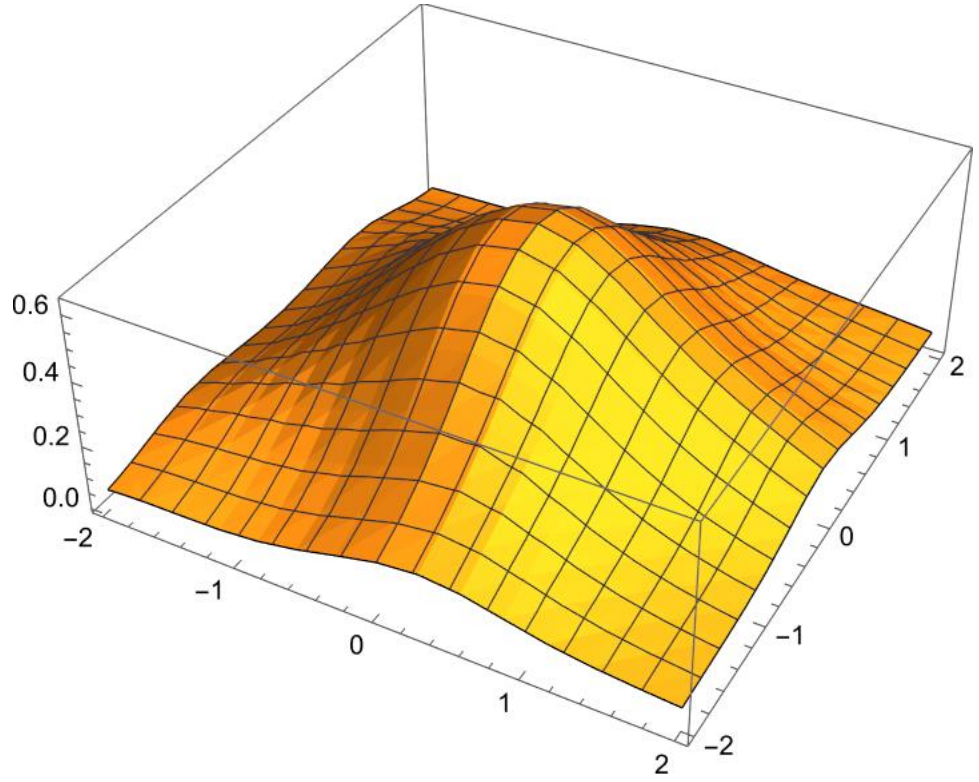
モデルは自由に作ってよい

線形層で回帰：入力２→　出力１

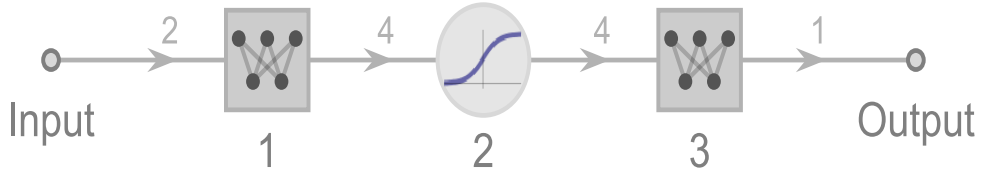
与えられた観測値の集合



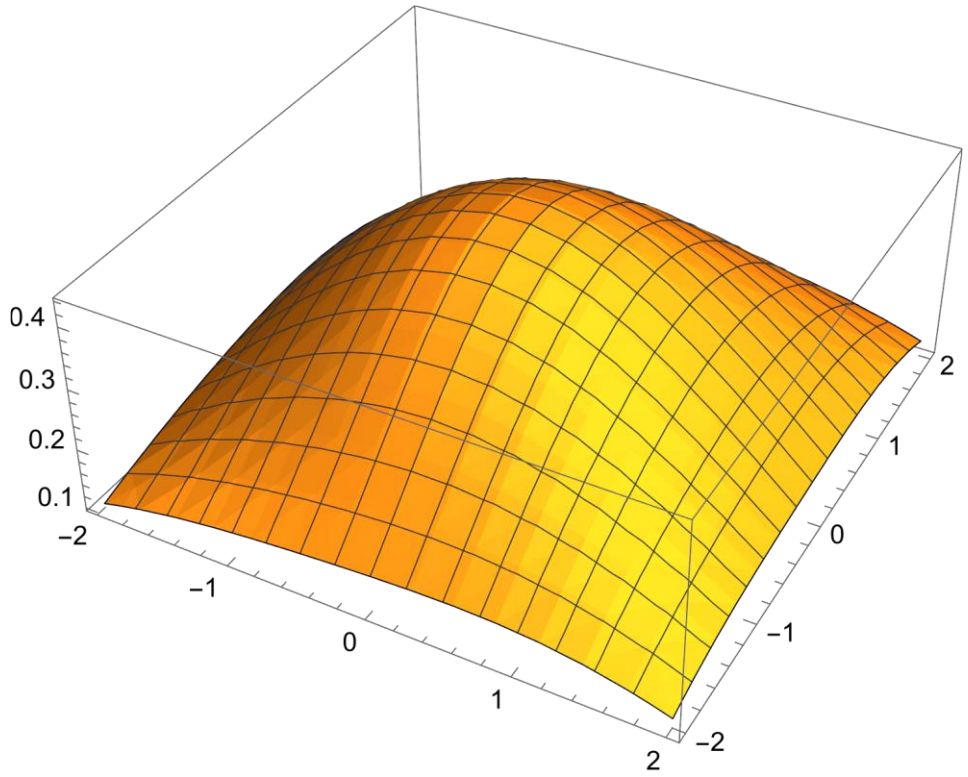
この回帰モデルで作成された予測値のようす



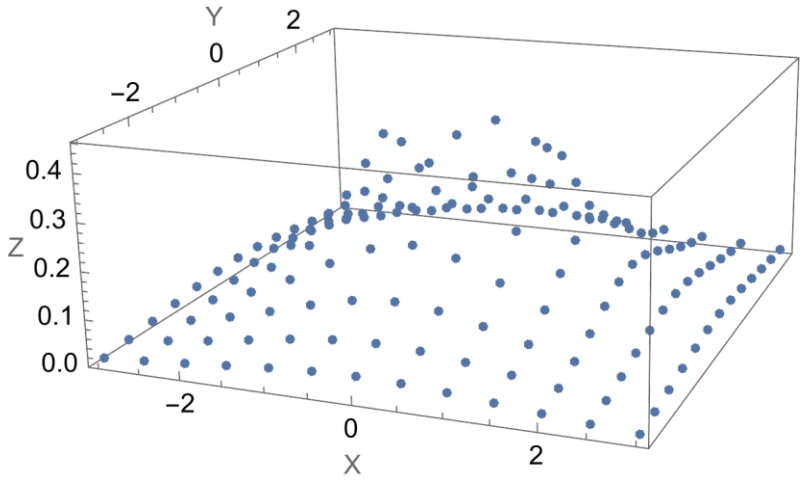
線形層で回帰：入力２→　出力１
異なるNNで回帰してみる．先ほどよりも簡単



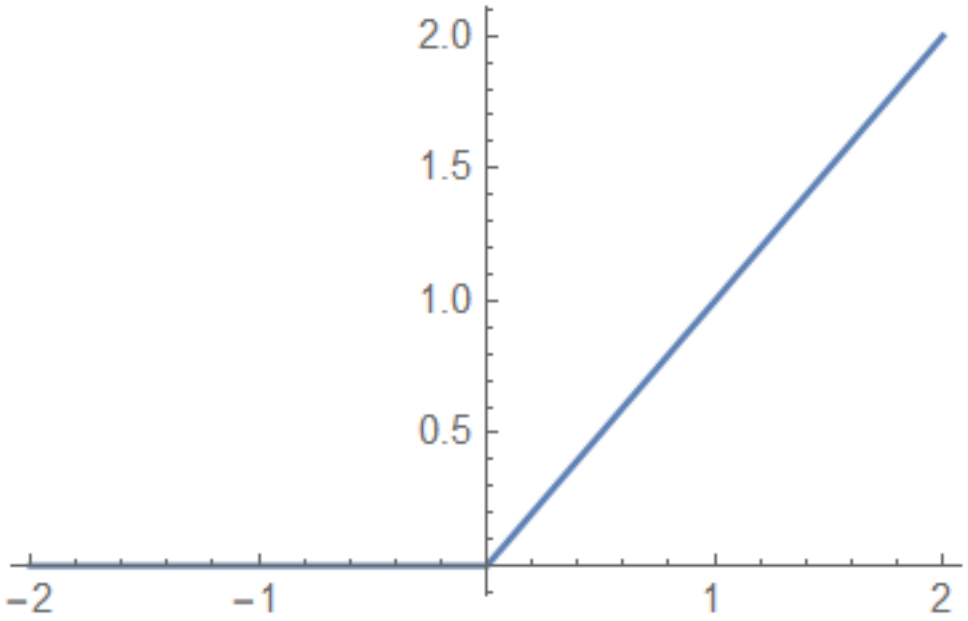
この回帰モデルで作成された予測値のようす
先ほどと異なる



与えられた観測値の集合



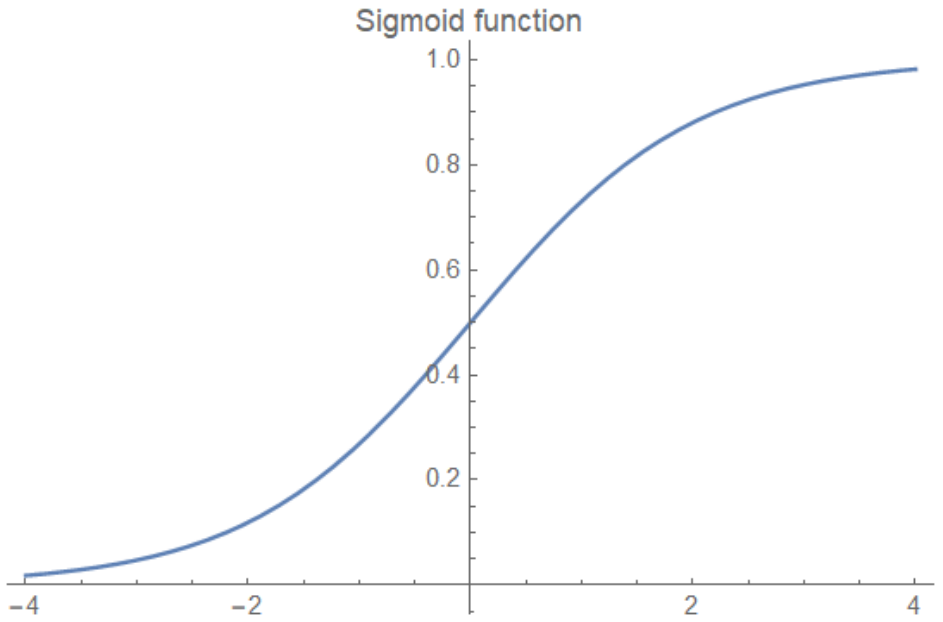
代表的　活性化関数
ReLU（Ramp）関数

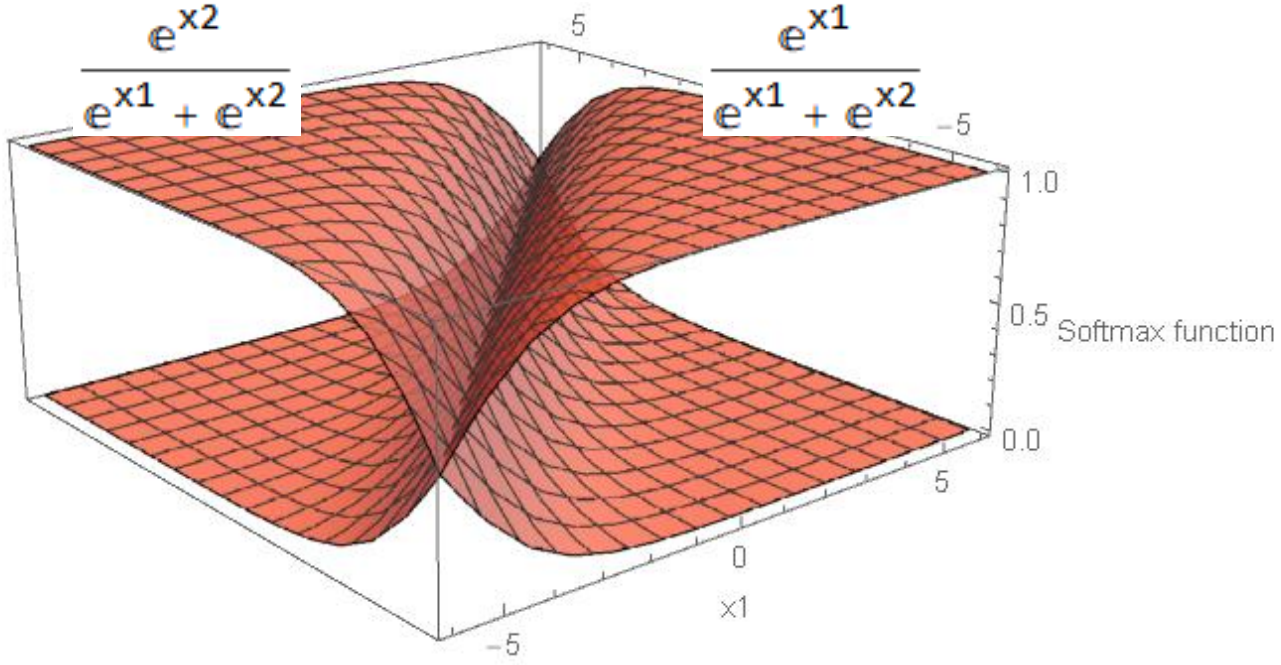


- x>=0のときは x を，
  それ以外のときは0

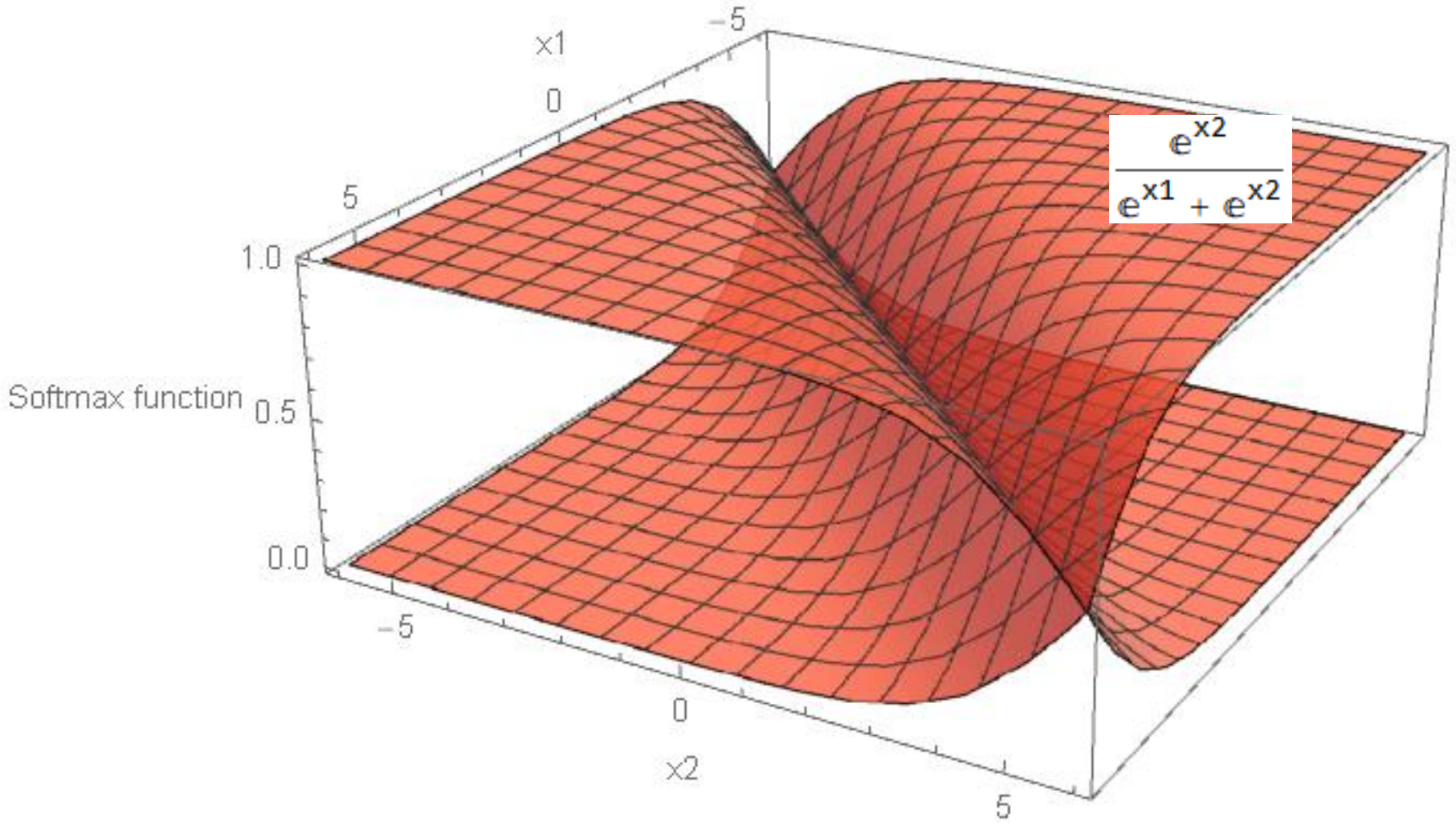
代表的活性化関数
シグモイド関数とソフトマックス関数

- $y = \frac{1}{1+e^{-x}}$

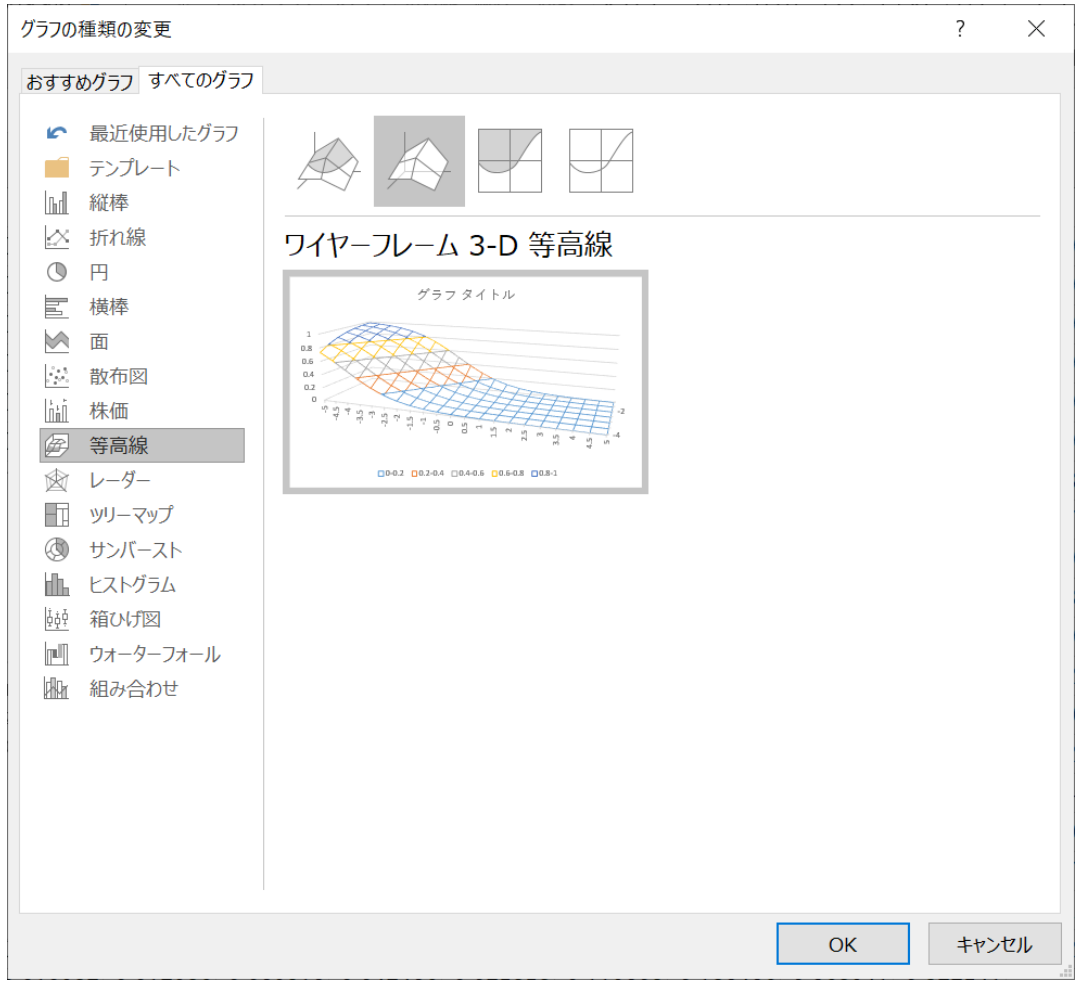


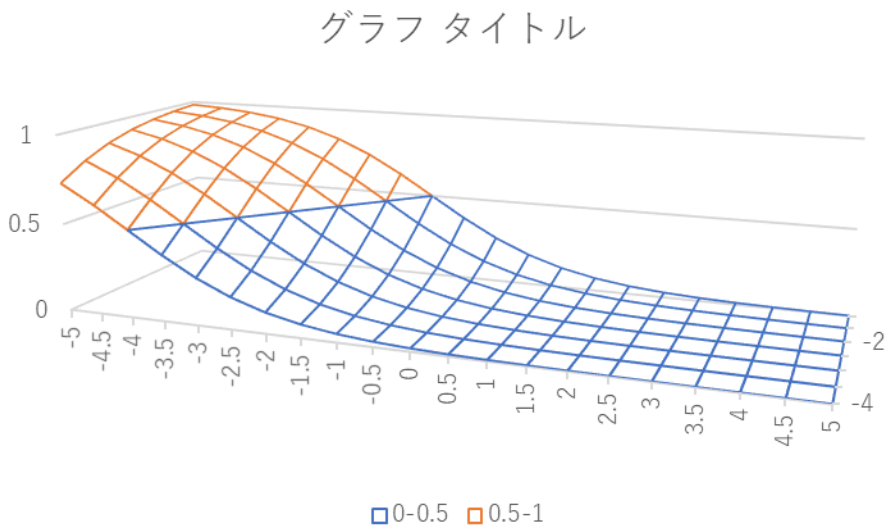


代表的活性化関数
2変数のソフトマックス関数



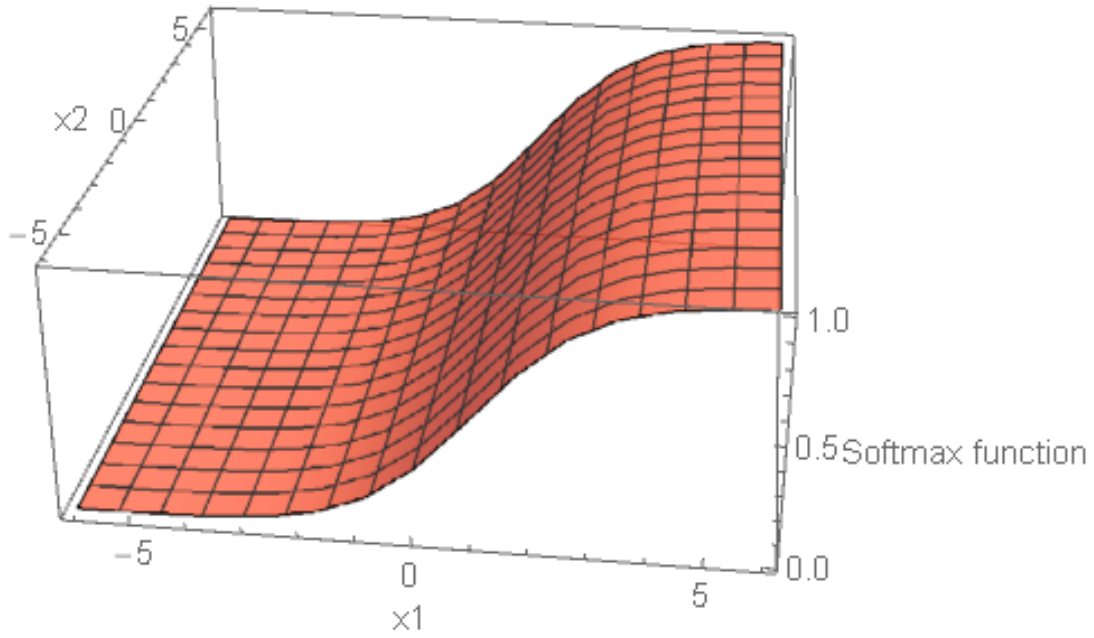
レポート＃７
EXCELで3次元の図を描く



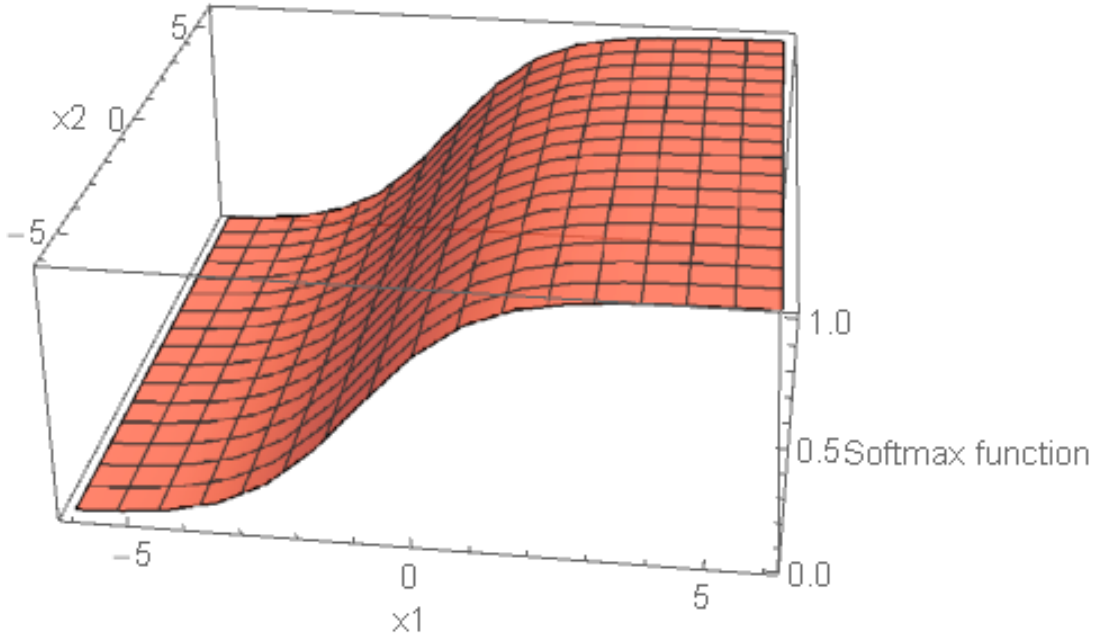


代表的活性化関数
2変数のソフトマックス関数

変数x2=1 の時のソフトマックス



変数x2=−1 の時のソフトマックス

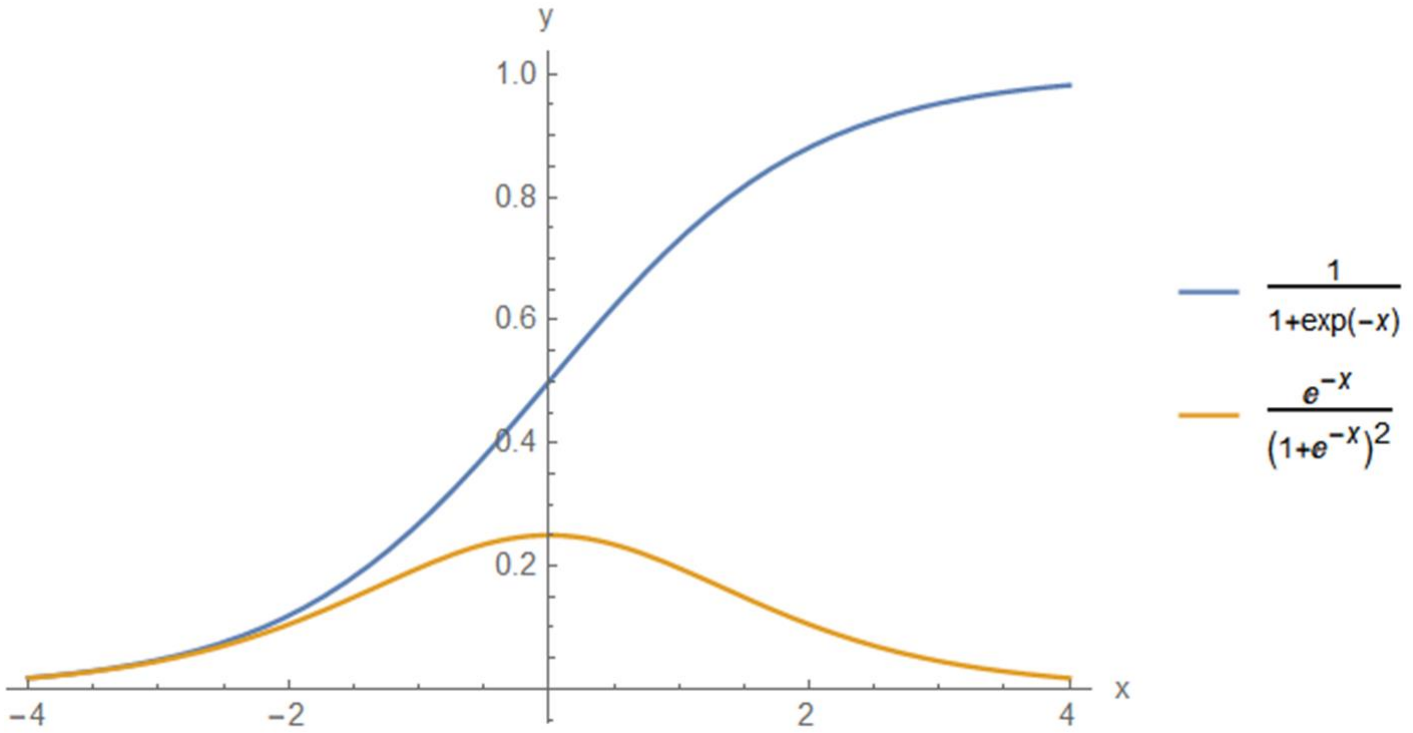


他の変数の値を固定すると、シグモイド関数のように増加する関数

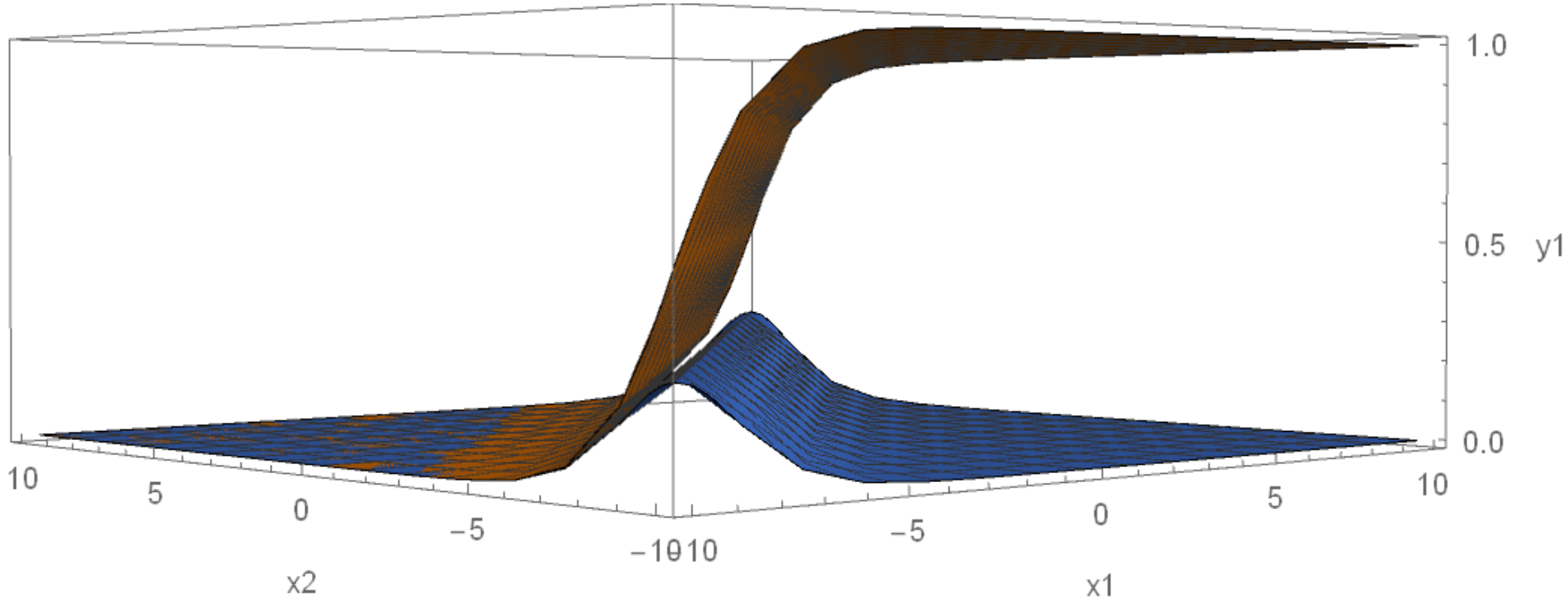
シグモイド関数を微分
最大微分係数はx=0において0.25(しかない）

- 課題：

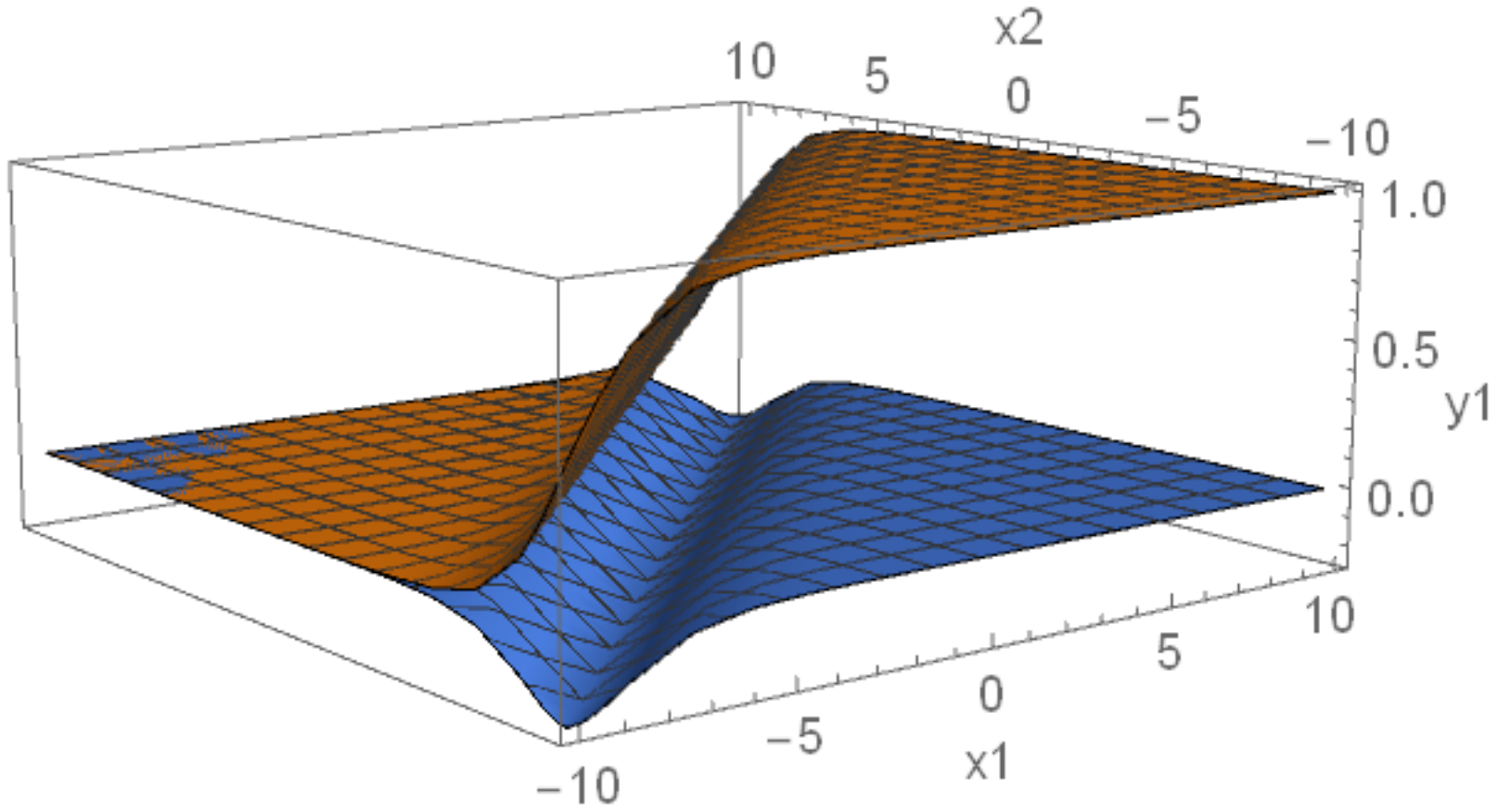
シグモイド関数を微分して、ｘ＝０において微分係数が0.25になることを確認せよ。



ソフトマックス関数y1をx1で微分
最大微分係数はx=0において0.25(しかない）



ソフトマックス関数y1をx2で微分
最小微分係数はx=0において−0.25(しかない）



ソフトマックス関数の一般系

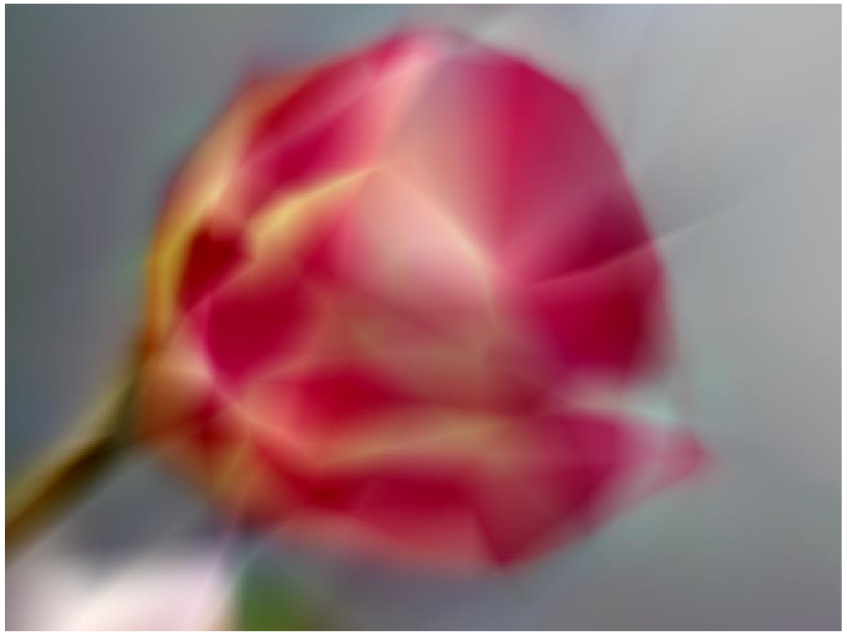
$$y_i = \frac{e^{x_i}}{e^{x_1} + e^{x_2} + e^{x_3} + \cdots + e^{x_n}}$$

$$= \frac{e^{x_i}}{\sum_{k=1}^{n} e^{x_k}}$$

NNを使ってどのように回帰を行うのか

NNの例：お花の絵を関数として覚えさせたい
写真の位置情報(x, y)を入力すると
その位置のRGBの色情報(r, g, b)を出力するNN
(引用：Wolfram Mathematica オンラインマニュアル NetTrain)

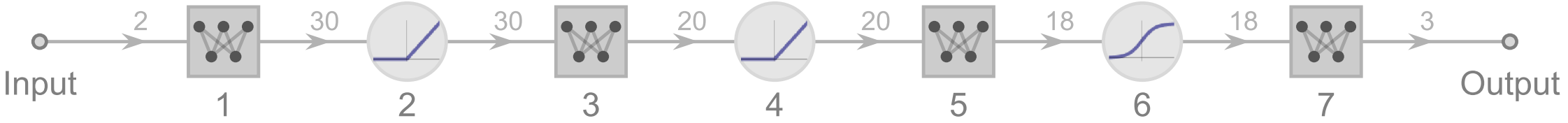
回帰



NNからもとの画像全体を予測する

7レイヤのNN

- このようなNNモデルにしてみた



- 座標（x,y）からその点の色RGBを予測する.

NetChain[uninitialized]

	Input	array
1	LinearLayer	vector (size: 30)
2	Ramp	vector (size: 30)
3	LinearLayer	vector (size: 20)
4	Ramp	vector (size: 20)
5	LinearLayer	vector (size: 18)
6	LogisticSigmoid	vector (size: 18)
7	LinearLayer	vector (size: 3)
	Output	vector (size: 3)

4個の線形層

- (x,y)を
  サイズ10に増やす線形変換
- Wとｂのパラメータ値を訓練で
  調整していく

$$h = xW + b$$

右のNNで、ｗのパラメータは
何個か？
2×10＋10×20＋20×10＋10×3

	Input	array
1	LinearLayer	vector (size: 10)
2	Ramp	vector (size: 10)
3	LinearLayer	vector (size: 20)
4	Ramp	vector (size: 20)
5	LinearLayer	vector (size: 10)
6	LogisticSigmoid	vector (size: 10)
7	LinearLayer	vector (size: 3)
	Output	vector (size: 3)

NNのモデルを変えると，予測値も変わる

NetChain[uninitialized

	Input	array
1	LinearLayer	vector (size: 20)
2	Ramp	vector (size: 20)
3	LinearLayer	vector (size: 20)
4	Ramp	vector (size: 20)
5	LinearLayer	vector (size: 15)
6	LogisticSigmoid	vector (size: 15)
7	LinearLayer	vector (size: 3)
	Output	vector (size: 3)

ChatGPTもNN

入力

- -ユーザーからのテキストデータ（質問、コマンドなど）
- テキストはトークン化され、各トークンは数値（トークンID）に変換される
- これらのトークンIDがニューラルネットの入力となる

出力

- 確率分布に基づき次に来るべきトークンを選択
- 文や文章を生成するプロセスを繰り返す

Mathematica 13.3でインプリされたGPT2

```
In[32]:= model = NetModel["GPT2 Transformer Trained on WebText Data", "Task" → "LanguageModeling"]
```
ネットワークモデル

Out[32]= NetChain

	Input	string
embedding	NetGraph (4 nodes) ↗	vector of *n* indices (range: 1..50 257)
decoder	NetChain (13 nodes)	matrix (size: *n*×768)
last	SequenceLastLayer	matrix (size: *n*×768)
classifier	LinearLayer ↗	vector (size: 768)
probabilities	SoftmaxLayer	vector (size: 50257)
	Output	vector (size: 50257)
		class

ノートブックにデータがありません．今すぐ保存する

```
In[33]:= Information[model, "SummaryGraphic"]
```
情報

Out[33]=


```
In[35]:= model["Gakushuin University offers higher education to", {"TopProbabilities", 5}]
Out[35]= { people → 0.0262668,  those → 0.0453833,  the → 0.0493315,  students → 0.0922493,  all → 0.0999936}

In[36]:= model["My cat is so", {"TopProbabilities", 5}]
Out[36]= { much → 0.0225146,  small → 0.0302152,  adorable → 0.0386848,  happy → 0.0532003,  cute → 0.129104}
```

Enbedding 埋込：トークンの意味を数字のベクトルで表わす

Embedding.nb - Wolfram Mathematica

ファイル (F)　編集 (E)　挿入 (I)　書式 (R)　セル (C)　グラフィックス (G)　評価 (V)　パレット (P)　ウィンドウ (W)　ヘルプ (H)

Output

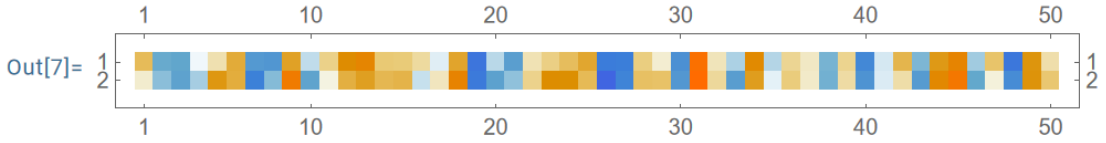
```
In[1]:= net = NetModel["GloVe 50-Dimensional Word Vectors Trained on Wikipedia and Gigaword-5 Data"]
        ネットワークモデル

Out[1]= EmbeddingLayer[ Input: string
                        Output: matrix (size: n×50) ]
        ノートブックにデータがありません．今すぐ保存する

In[2]:= NetExtract[net, "Input"]
        ネットワークの層を取り出す

Out[2]= NetEncoder[ Type: Tokens
                    Output: vector of n indices (range: 1..400 001) ]
        ノートブックにデータがありません．今すぐ保存する

In[7]:= net["cat dog"] // MatrixPlot
        行列プロット
```



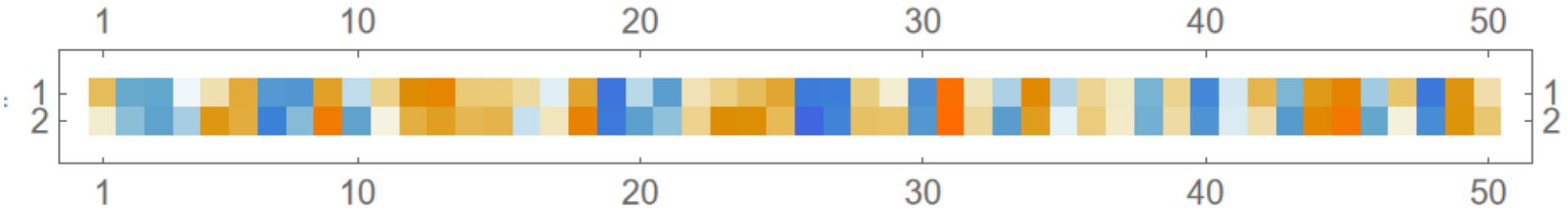

```
In[8]:= net["black cat"] // MatrixPlot
        行列プロット
```

意味が近ければ，似たような数字列になる

埋込
意味が近ければ，似たような数字列になる

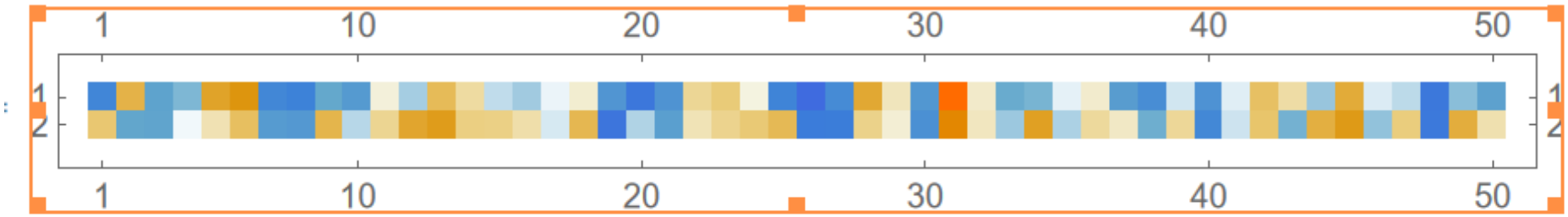
```
net["cat dog"] // MatrixPlot
```

行列プロット



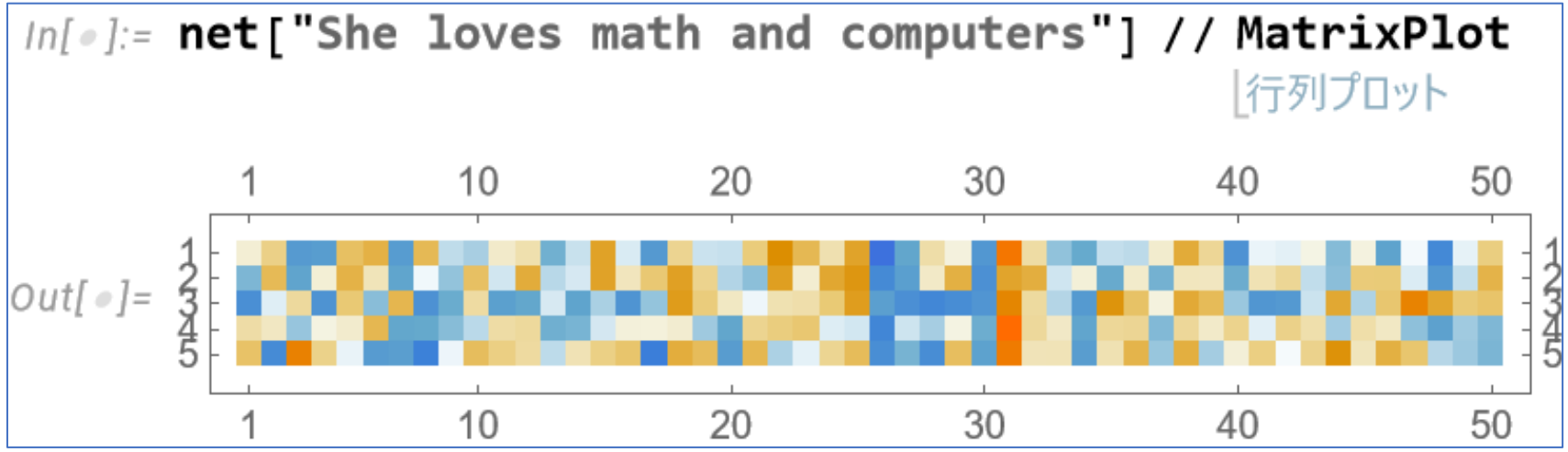

```
net["black cat"] // MatrixPlot
```

行列プロット



GPT-2 のバージョンとエンベディングの次元各種ある

- **小規模モデル (Small)**: 768次元
- **中規模モデル (Medium)**: 1024次元
- **大規模モデル (Large)**: 1280次元
- **最大規模モデル (XL)**: 1600次元
- GPT-4:大体768から12800までの範囲



損失関数

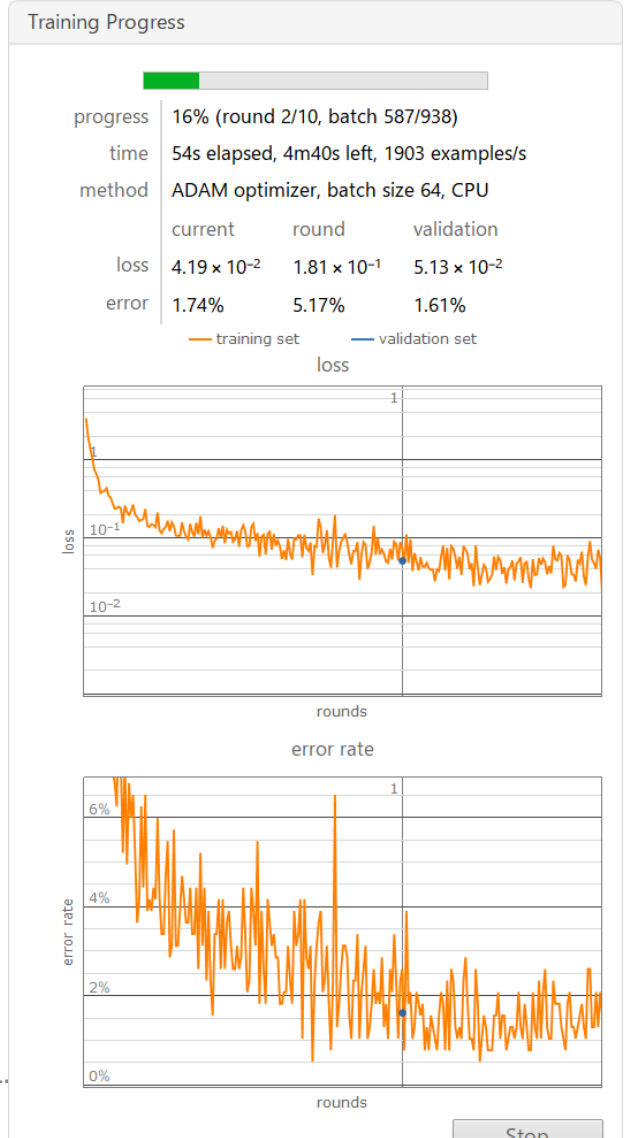
損失関数(Loss Function)

- 与えた教師データ t のように、パラメータを調整してほしい。
- ずれていたらその差を減じる方向に、パラメータを調整したい
- ニューラルネットからの実際の出力 y と，
  教師データ t のずれを示した関数が、損失関数
- 代表的な損失関数：**平均２乗誤差**(mean squared error)
- $L = \frac{1}{N}\sum_{i=1}^{N}(t_i - y_i)^2$

  残差の平方和

手書き数字の認識　MNISTデータ
(引用：Wolfram Mathematica オンラインマニュアル NetTrain)
tutorial/NeuralNetworksComputerVision#1784732780

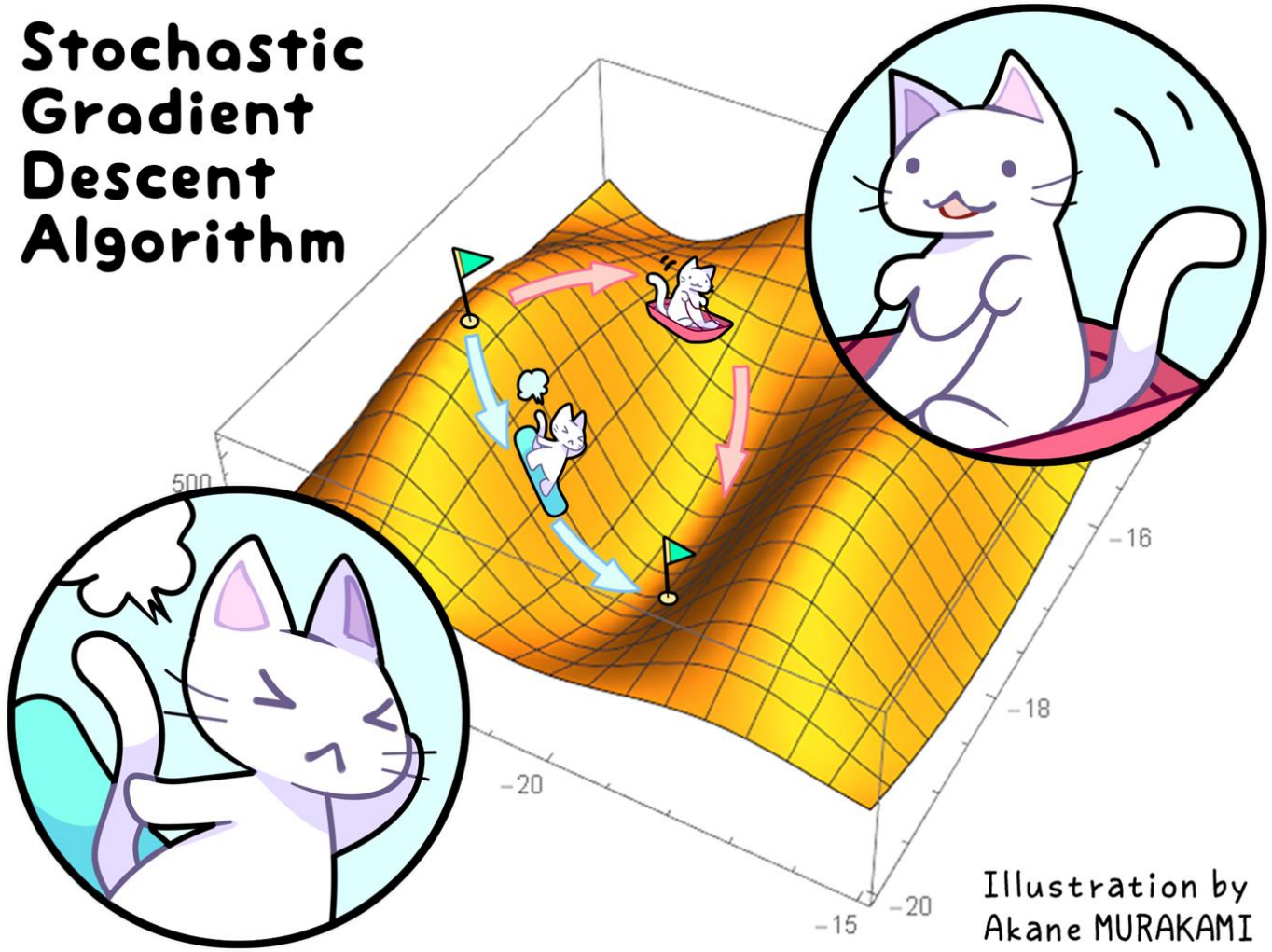
- 訓練している間、損失関数の値が減じるようす
- エラー率を見ると、時々かえってエラー率が大きく
  なることもあるが、全体としては減少している様子が見える



勾配降下法

どちらの方向に滑り落ちれば最速で極小値に行けるか、偏微分の係数で決める

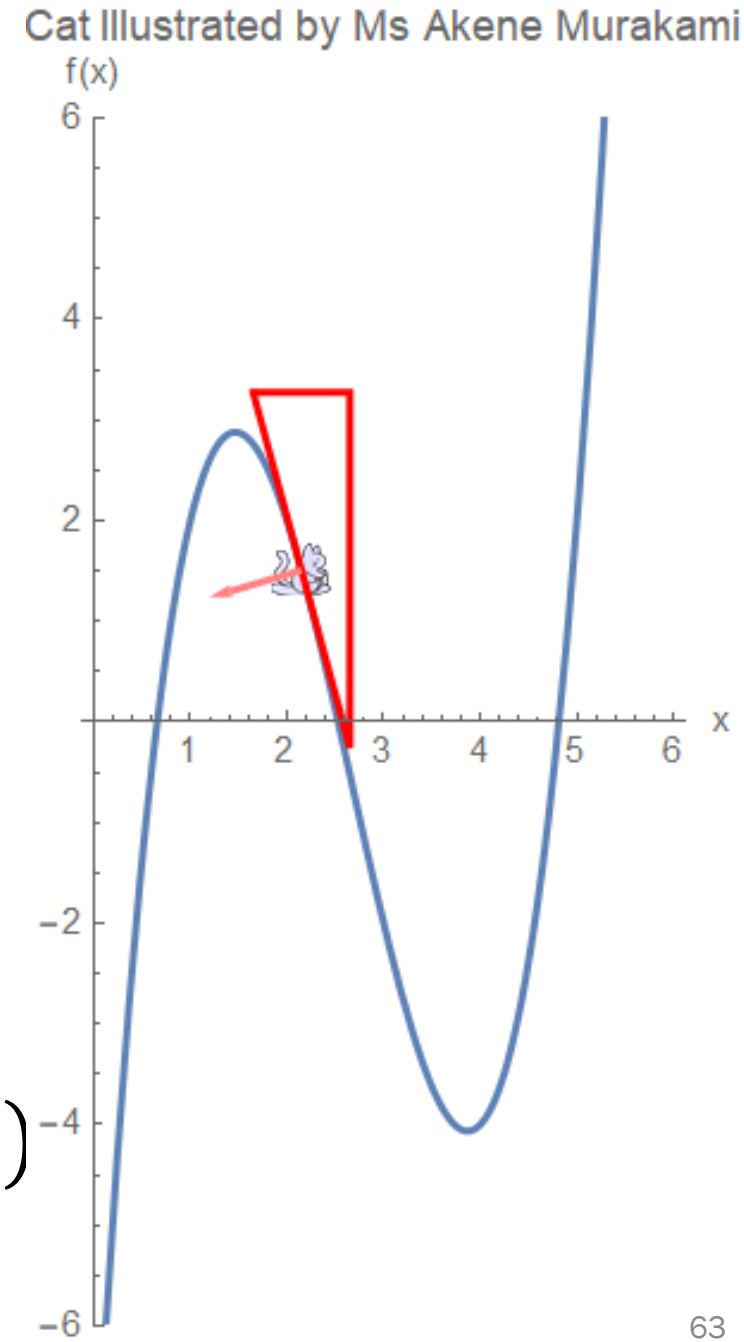
- 1000次元のような高次元空間の極小値を解析的に求めることは不可能なので、足元周辺で微分係数が負に大きいところを探して、1歩進む。また、その足元周辺で微分係数が負に大きいところを探して1歩進む。

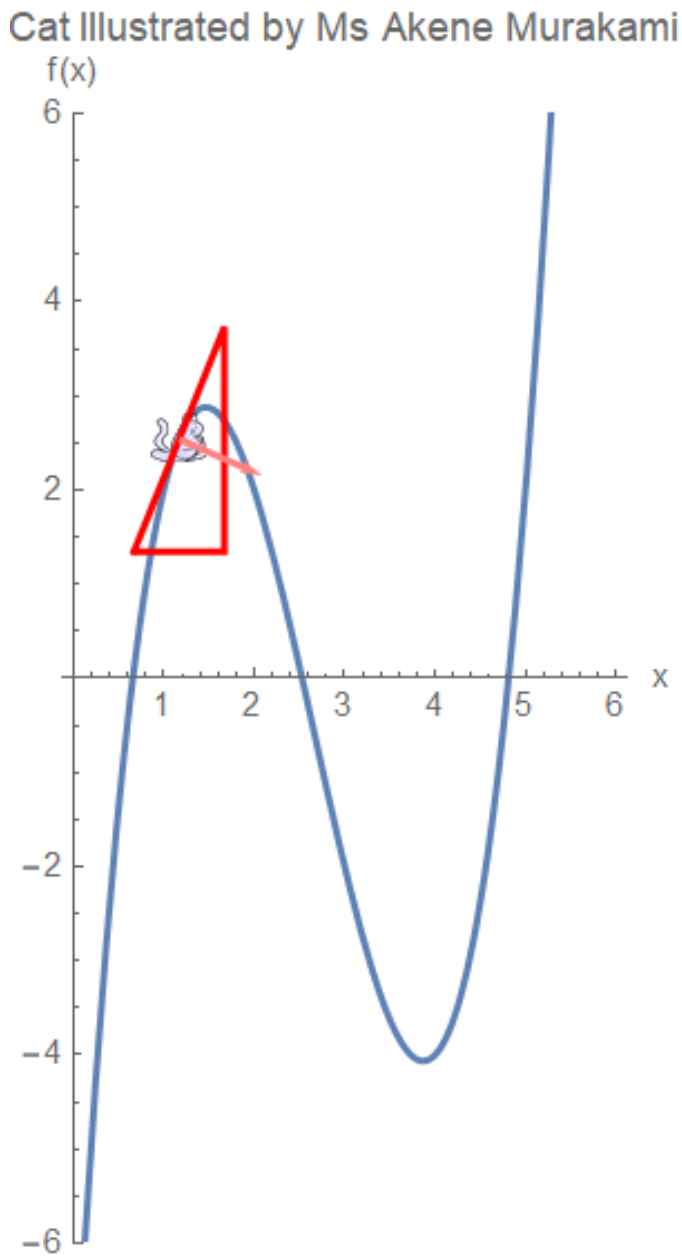


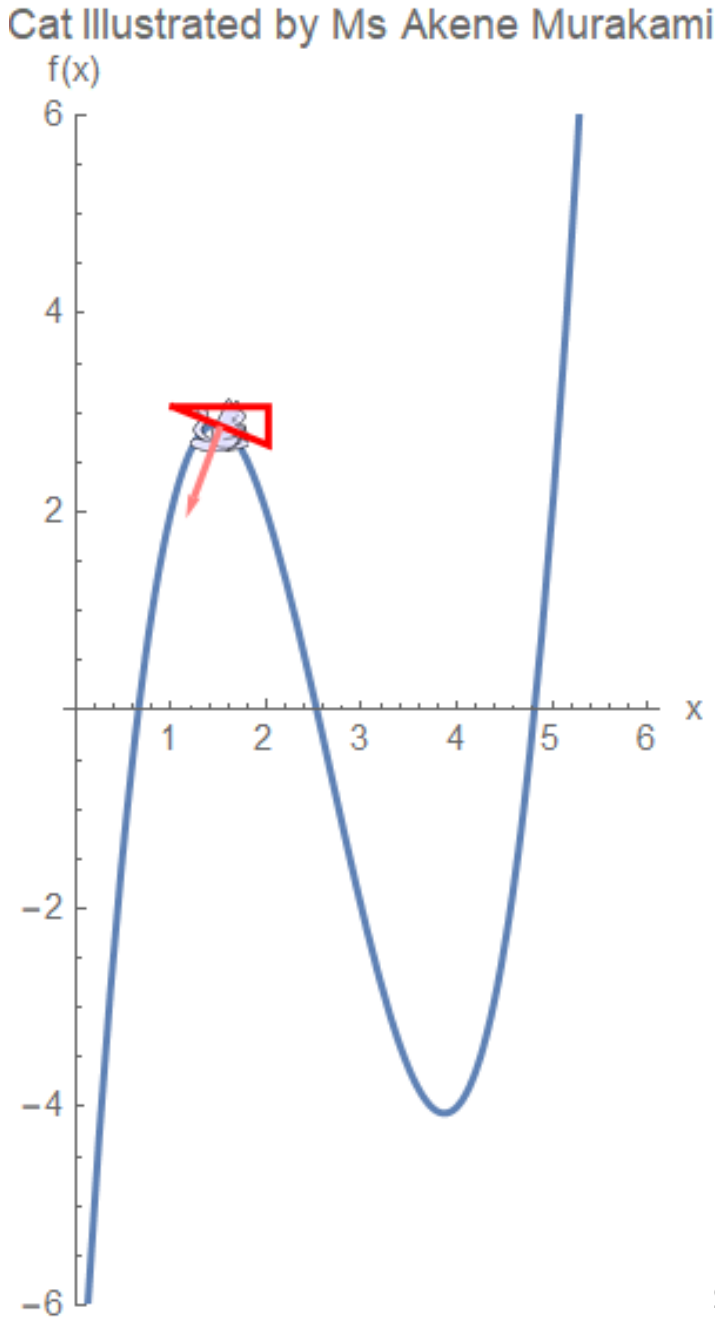
1変数関数は直線のサーフボード(接線の１部分）
2変数関数は平面のサーフボード（接平面の1部分）

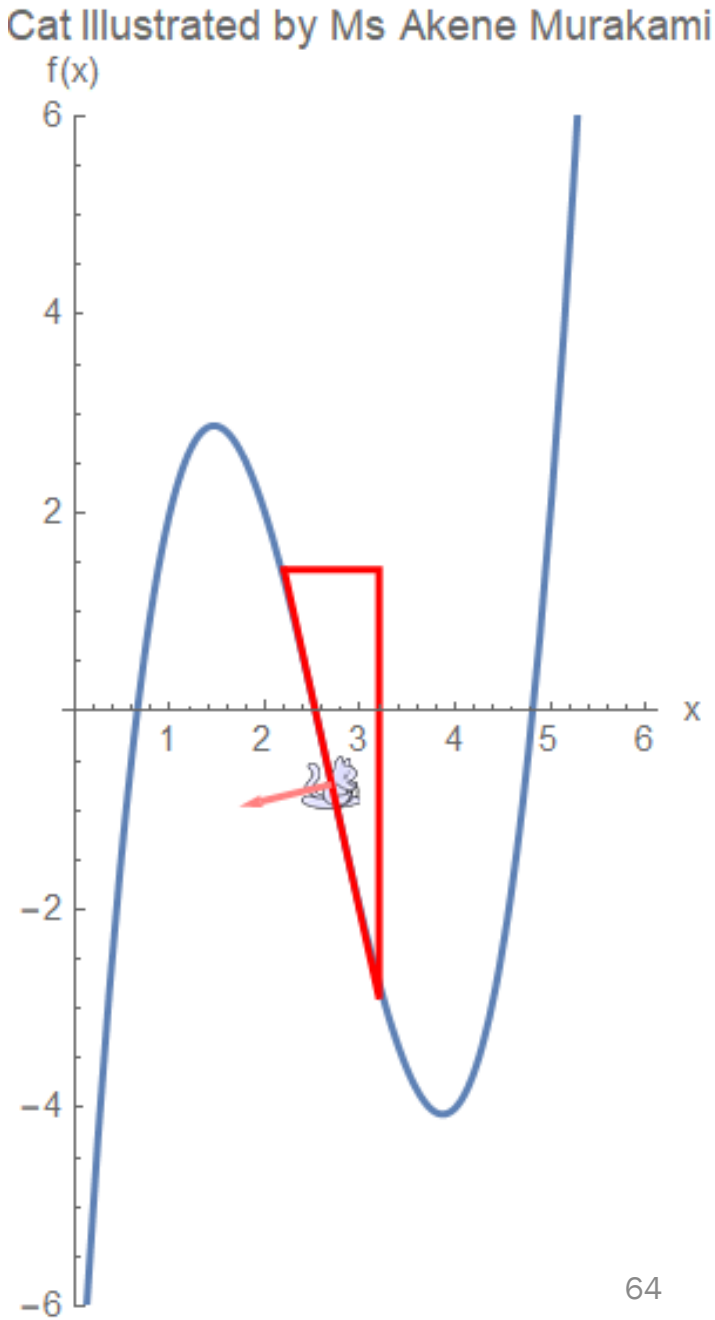
- 1変数関数の場合
- 進むべきは、極小値を目指すので
  法線ベクトルと逆のｘ方向に進む
- **法線：接点を通り接線に垂直な曲線**
- 点x=aにおける**法線ベクトル**
  $\{f'(a), -1\}$ 方向の単位ベクトル（長さ１）
- 証明

$$y - f(a) = f'(a)(x - a)$$
$$f'(a)(x - a) - 1 \times \big(y - f(a)\big)$$



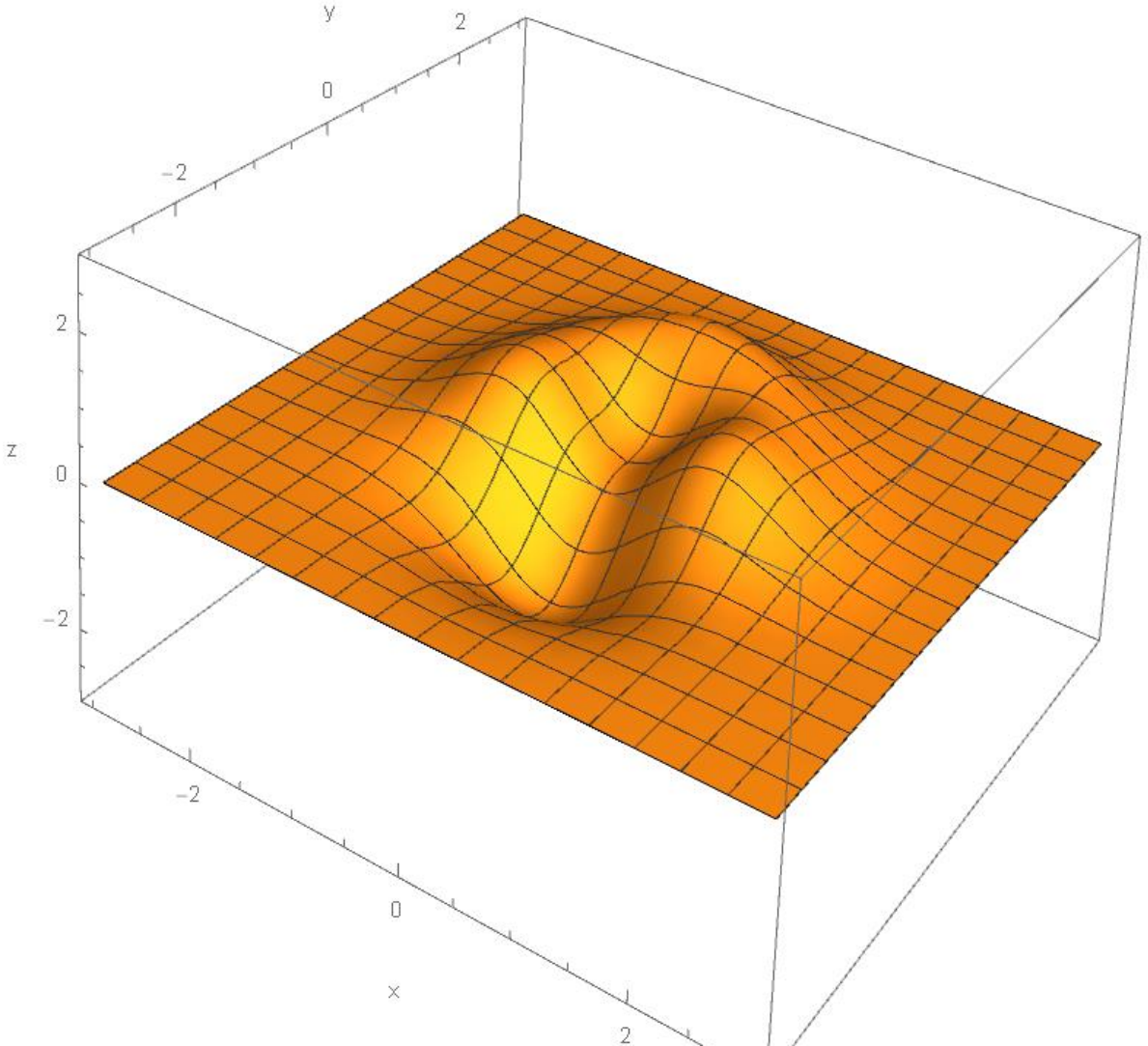

Cat Illustrated by Ms Akene Murakami
f(x)
x
6
4
2
−2
−4
−6
1
2
3
4
5
6

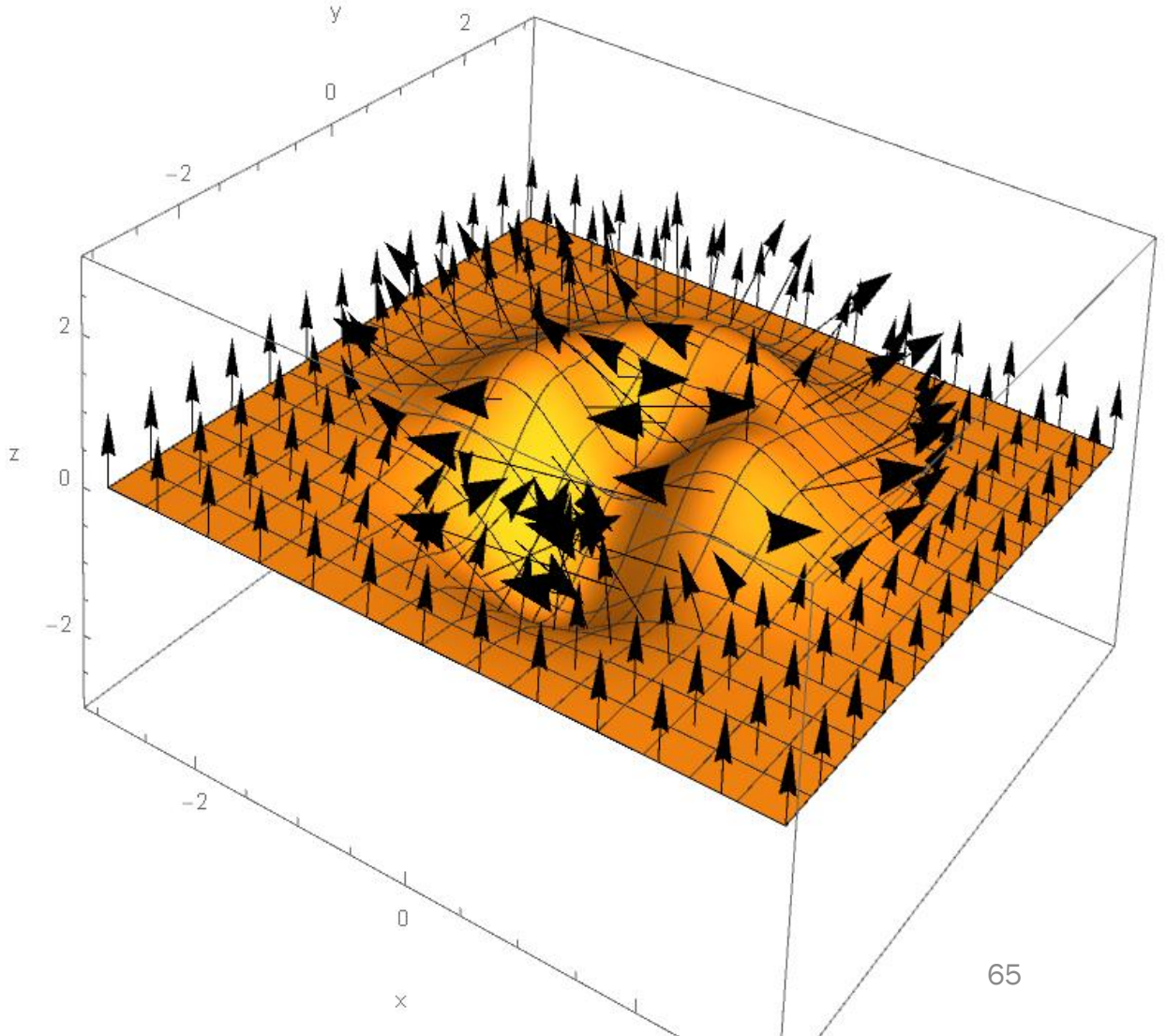

Cat Illustrated by Ms Akene Murakami
f(x)
x
6
4
2
−2
−4
−6
1
2
3
4
5
6


Cat Illustrated by Ms Akene Murakami
f(x)
x
6
4
2
−2
−4
−6
1
2
3
4
5
6

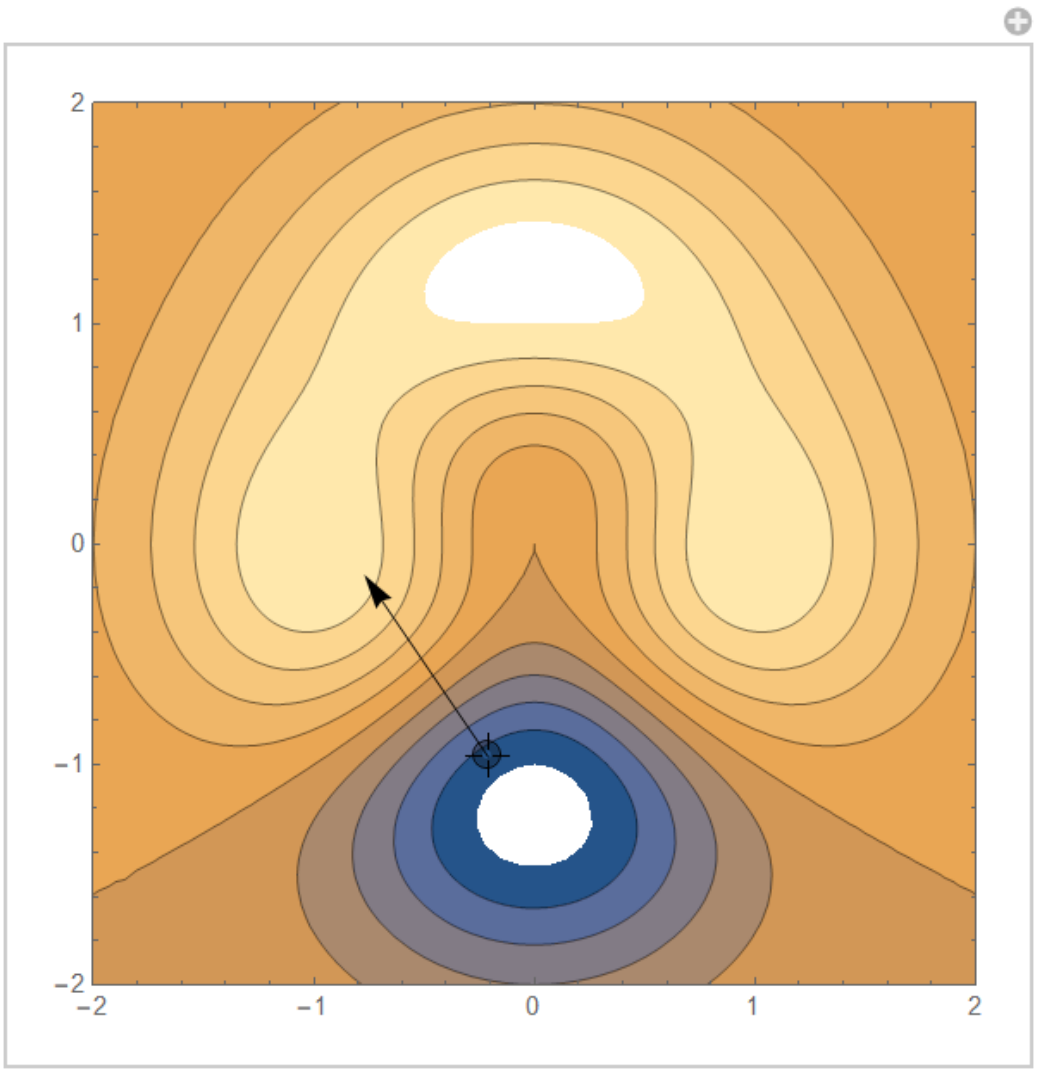
1.

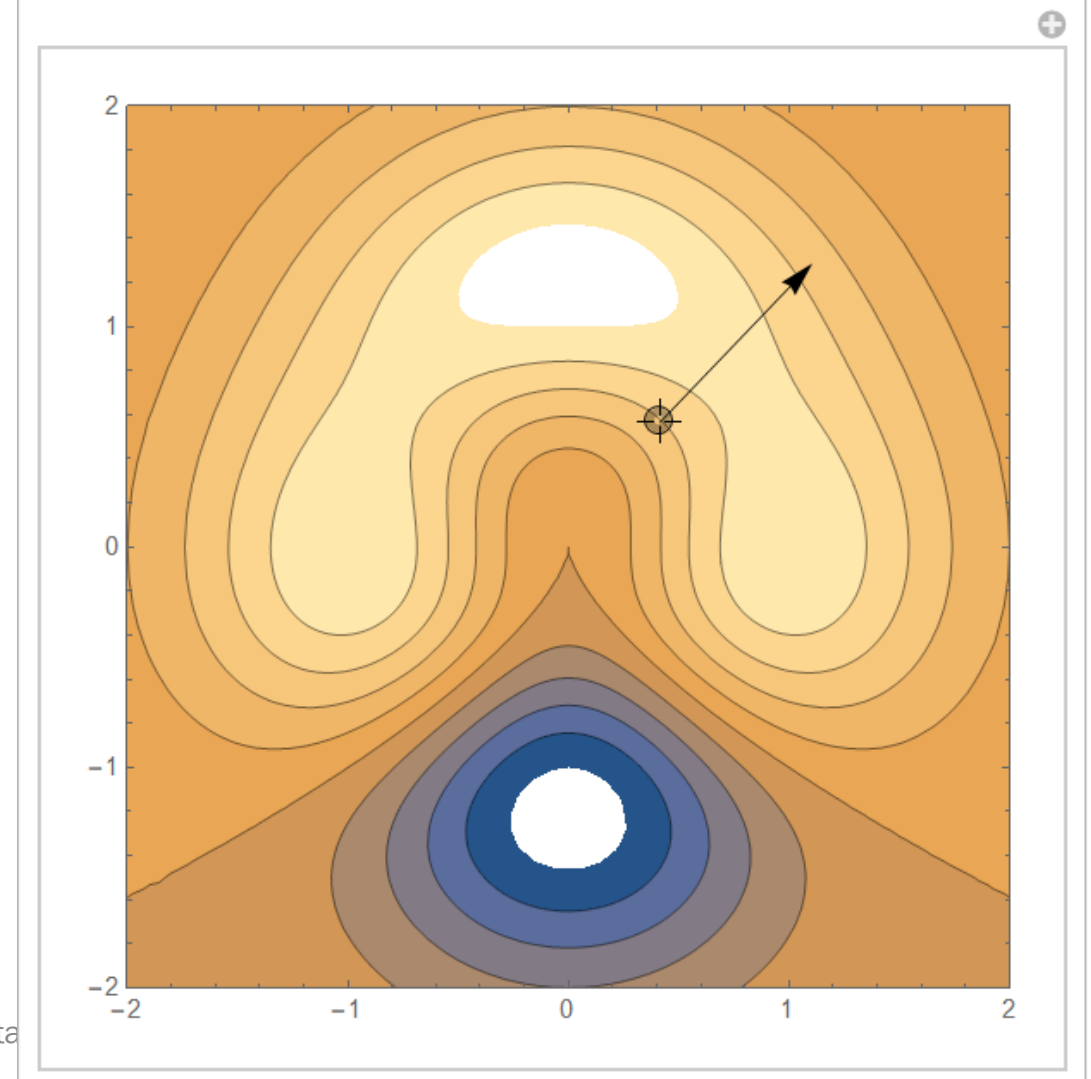
2変数関数は平面のサーフボード
接平面
法線方向を矢印で示した(長さは適当）





法線ベクトルを等高線図上で示す
進むべきは、極小値を目指すので法線ベクトルと逆の方向に進む





法線ベクトル

- 法線ベクトルは$\{f_x(a,b), f_y(a,b), -1\}$方向の単位ベクトル
- 証明
  接平面の式
  $z - f(a,b) = f_x(a,b)(x-a) + f_y(a,b)(y-b)$