

私が愛用しているツールたち

- python, jupyter-lab, nbdev, poetry, fastai, pycaret, prophet, streamlit, gurobi, mypulp, miro, etc. -

久保幹雄（東京海洋大学）

Amazonで検索

久保 幹雄



✓ フォロー中

フォローすると、最新刊やおすすめ作品の情報を入手できます。

久保 幹雄 Mikio Kubo
専門は、サプライ・チェーンならびに組合せ最適化、
早稲田大学理工学研究科卒、博士(工学)、早稲田大学助手、東京商船大学助教授、ポルト大学招聘教授などを歴任、現在 東京海洋大学教授。代表的な著書として、「離散構造とアルゴリズムIV」近代科学社、「巡回セールスマン問題への招待」朝倉書店、「組合せ最適化とアルゴリズム」共立出版、「ロジスティクス工学」朝倉書店、「実務家のためのサプライ・チェーン最適化入門」朝倉書店、「ロジスティクスの数理」共立出版、「メタヒューリスティクスの数理」共立出版、「サプライ・チェーン最適化ハンドブック」朝倉書店、「サプライ・チェーン最適化の新潮流-統一モデルからリスク管理・人進支援まで-」朝倉書店などがある。

<http://www.logopt.com/mikiokubo/>
もっと少なく読む

他のお客様と一緒に購入した商品:

Kindle版 ¥3,168 32pt (1%)	Kindle版 ¥6,336 63pt (1%)	Kindle版 ¥2,613 48pt (2%)	単行本 ¥2,987	単行本 ¥2,970 30pt (1%)	単行本 ¥3,630 36pt (1%)	単行本 ¥5,195	単行本 (ソフトカバー) ¥3,520 35pt (1%)	単行本 ¥4,514	単行本 ¥500	大型本 ¥28,600 286pt (1%)	単行本 ¥18,700 187pt (1%)

久保 幹雄の作品

すべてのフォーマット: Kindle版 ペーパーバック 単行本 (ソフトカバー) [続きを見る](#)

言語:

並べ替え:

あたらしい数理最適化: Python言語とGurobiで解く 2016/09/06
久保 幹雄, ベドロロ ジョア・ベドロ, 村松 正和, レイス アブドル
Kindle版
¥3,168 ~~¥3,520~~
ポイント: 32pt (1%)

単行本
¥3,520
ポイント: 35pt (1%)
残り1点 [ご注文はお早めに](#)

Python言語によるビジネスアナリティクス: 実務家のための最適化・統計解析・機械学習 2016/09/01
久保 幹雄, 小林 和博, 齊藤 努, 並木 誠, 橋本 英樹
Kindle版
¥6,336 ~~¥7,040~~
ポイント: 63pt (1%)

単行本
¥7,040
ポイント: 70pt (1%)
残り2点 (入荷予定あり)

★★★★☆ (3)

★★★★☆ (4)

メニュー

開発環境

[python](#), [jupyter-lab](#), [nbdev](#), [poetry](#)

機械学習

[fastai](#), [pycaret](#), [prophet](#)

Webアプリ作成とデプロイ

[streamlit](#)

数理最適化

[gurobi](#), [mypulp](#)

おまけ (コロナ禍でのSPRINT, 研究メモ, 翻訳, 研究連絡)

[miro](#), [notion](#), [deepl](#), [slack](#)

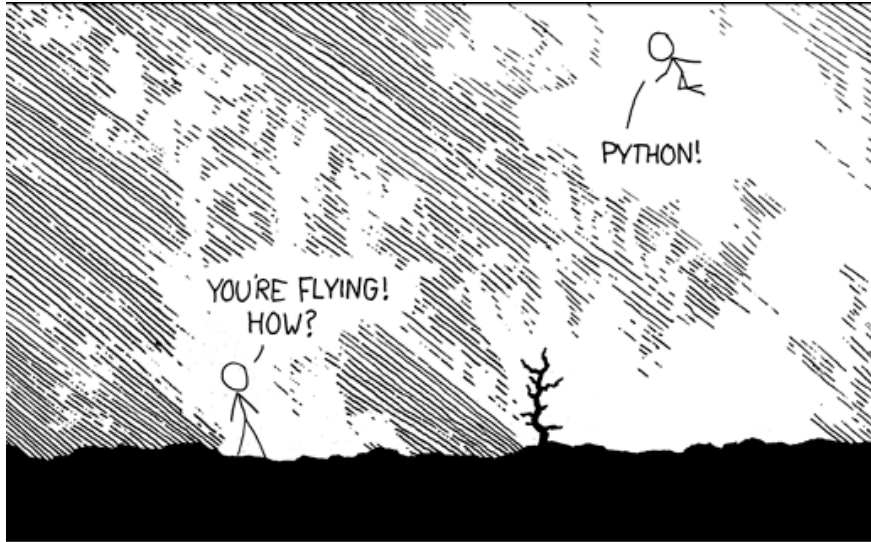
なぜ今Pythonか？

Pythonをお薦めする18の理由

久保 幹雄
東京海洋大学



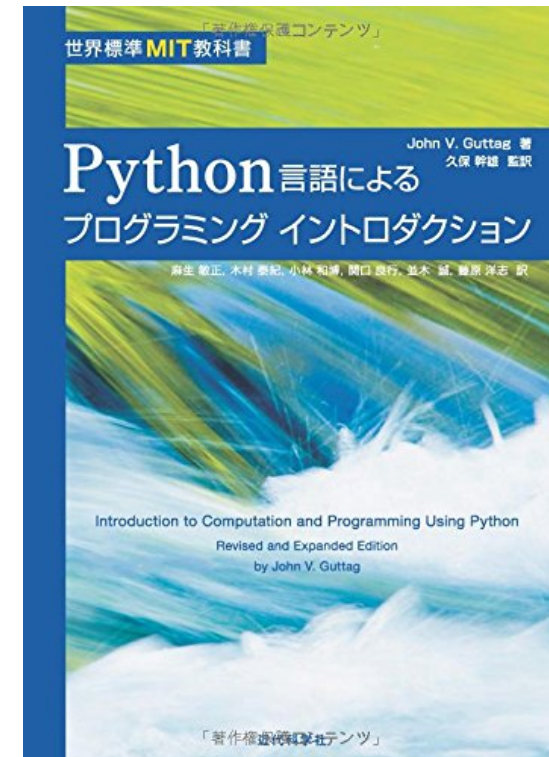
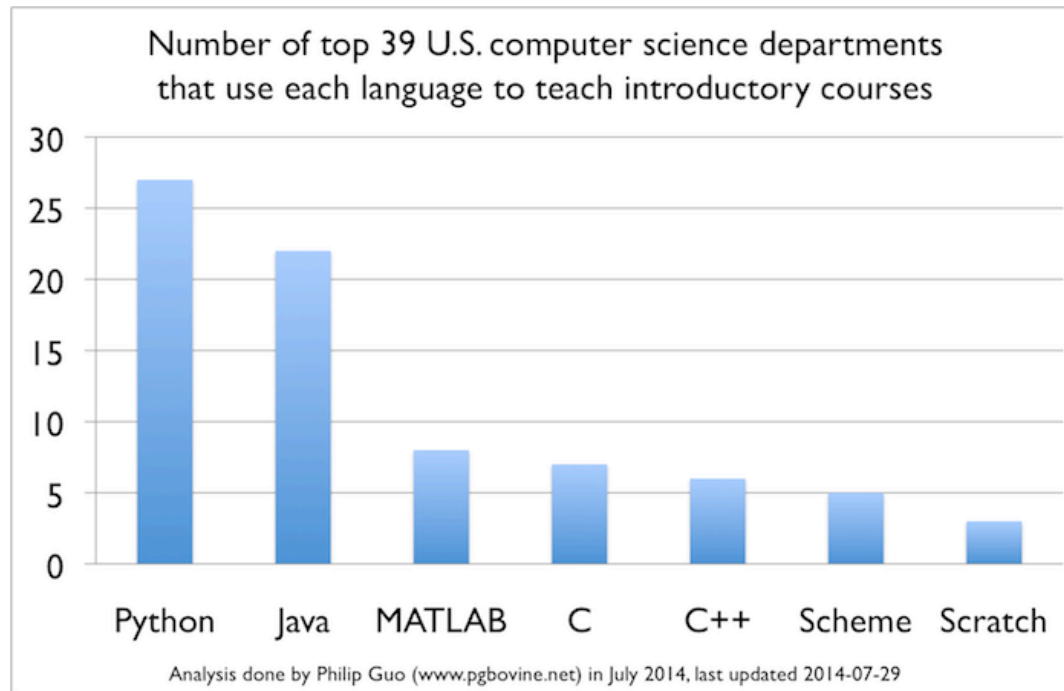
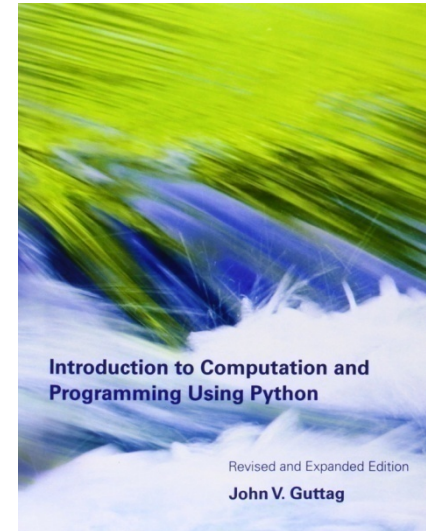
Why Python (1) ?



- モジュールをimportすれば何でもできる!
 - 最適化
 - データ解析
 - 統計
- 飛ぶこともできる !?
import antigravity
- Programming is fun again!

Why Python (2) ?

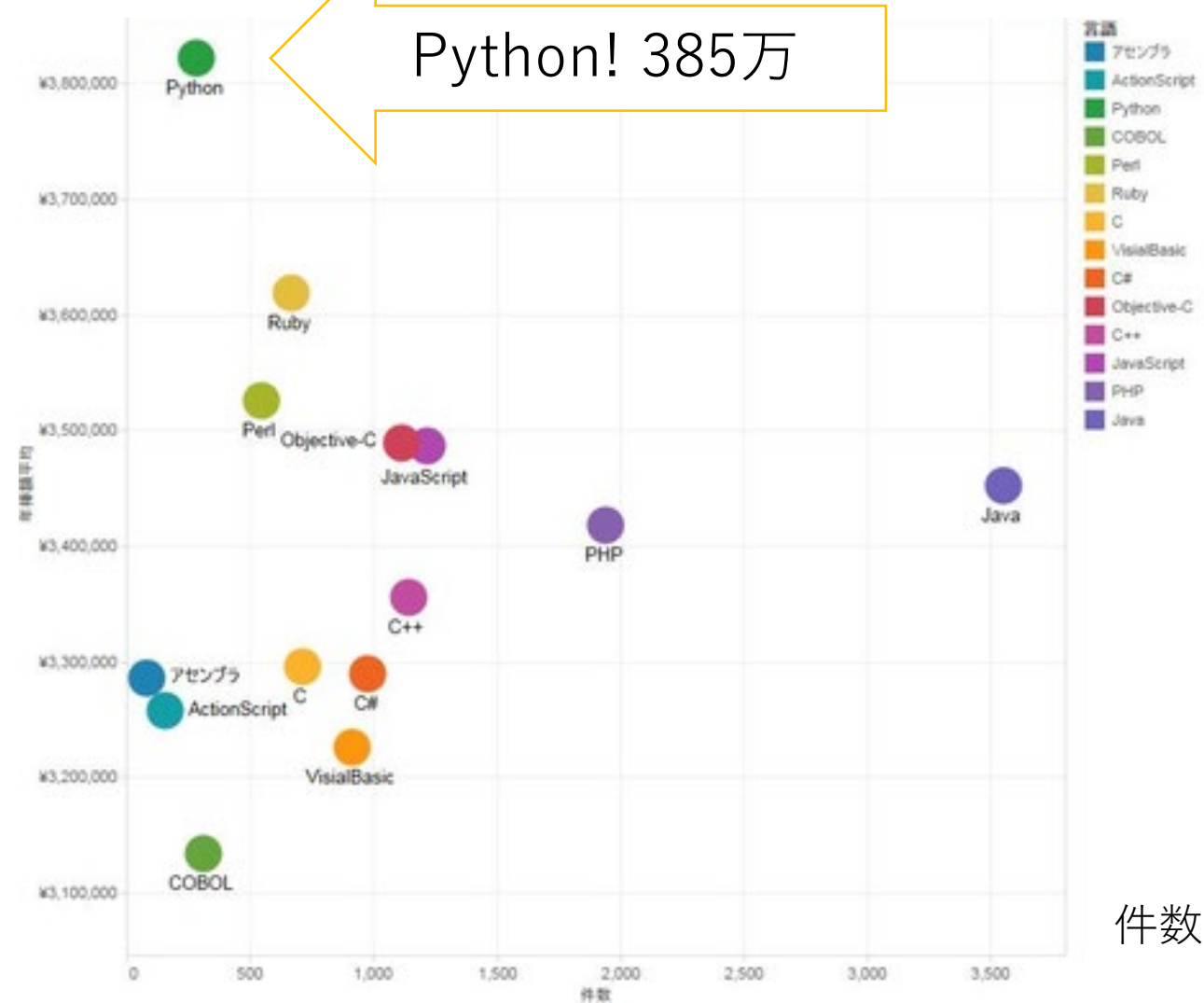
- 米国計算機科学科で採用数No.1
- もちろんMITも！ (Introduction to Computer Science and Programming Using Python by John Guttag)



Why Python (3) ?

- お金を稼ぐため!

プログラマの平均
初任給
(日本)



出典：マイナビ

件数

Why Python (4)?

- もっとお金を稼ぐため!

Top 10 Programming Languages to Learn in 2014

No 1. Python

平均収入: \$93,000

求人数: 24,533

Top Employers: Amazon, Dell, Google, eBay & Yahoo, Instagram, NASA, Yahoo

Example Sites: Google.com, Yahoo Maps, Reddit.com, Dropbox.com, Disqus.com

No 2. Java, No 3. Ruby, No. 4 C++, No5. JavaScript

No 6. C#, No. 7 PHP, No8. Perl ...

Why Python (5)?

- もっと、もっと、お金を稼ぐため!

Pythonはデータサイエンティストの必需品

*Help Wanted: **Black Belts in Data***

by Rodrigo Orihuela and Dina Bass

Bloomberg Businessweek



*Starting salaries for data scientists
have gone north of **\$200,000***

*McKinsey projects that by 2018 demand for data scientists
may be as much as **60 percent greater than the supply***

Why Python (6)?

- キーワード（覚えるべき予約語）が33個（Python 3.4）と圧倒的に少ない（Python3.9だと36個）。

False	None	True	and	as
assert	break	class	continue	def
del	elif	else	except	finally
for	from	global	if	import
in	is	lambda	nonlocal	not
or	pass	raise	return	try
while	with	yield		


Why Python (7)?

字下げの強要で、誰でも読みやすいプログラム

```
if (x > 1) { y=x+1;           (行儀の悪い) C++  
  z=x+y; } else { y=0; z=0; }
```

```
if x > 1:  
    y=x+1  
    z=x+y  
else:  
    y=z=0
```

Pythonだと誰でも

 この高さ（インデント）がそろっていないとエラーする！

Why Python (8)?

- 短時間で開発可能
(行数が短く, モジュール豊富)

“Hello, world!”と出力するプログラム Python 3 版

C++版 (覚える必要なし!)

```
#include <iostream>
int main() {
    std::cout << "Hello, world!" <<
std::endl;
    return 0;
}
```

```
print (“Hello, world!”)
```

python 2版

```
print “Hello, world”
```

Why Python (9)?

- 変数の宣言必要なし

```
Option Explicit

Sub Sample()
  Dim 基準値, N, Message
  基準値 = 100
  N = 基準値 * 2
  Message = "答は" & N & "です"
  MsgBox MESAAGE
End Sub
```



The image shows a screenshot of a Microsoft Visual Basic code editor window. The code defines a subroutine named 'Sample' with three variables: '基準値', 'N', and 'Message'. The code assigns values to these variables and uses them in a 'MsgBox' statement. However, the variable 'MESAAGE' is used in the 'MsgBox' statement, which is a typo for 'Message'. This causes a compilation error. An error dialog box is displayed over the code, with the title 'Microsoft Visual Basic' and a yellow warning icon. The message in the dialog box reads: 'コンパイルエラー: 変数が定義されていません。' (Compilation error: Variable is not defined). The dialog box has 'OK' and 'ヘルプ' (Help) buttons.

Why Python (10)?

- インタープリタ（コンパイルする必要なし）

巷の声

コンパイルを待つのは辛い

- 思考の中断
- 待ち時間にほかのことを始める

C#なんかで開発する場合だと、細かい修正をして動作確認を繰り返す場合、「F5押す→コンパイル待ち」となるのでそこらへんはムカついてくる。

コンパイルが遅すぎる

- 良いマシンを... (震え声)
- sbt の `~:compile` とか `~:test` でごまかす
- コンパイルが遅くなる言語機能を避ける...
- 何でもかんでもコンパイルしない (個人の意見です)

Why Python (11)?

- メモリ管理も必要なし
(自動ガーベッジ・コレクション)



Why Python (12)?

- 多くのプラットフォームで動作
(Windows, Mac, Linux, iphone, ...)



as seen by...

Mac
Fanboys

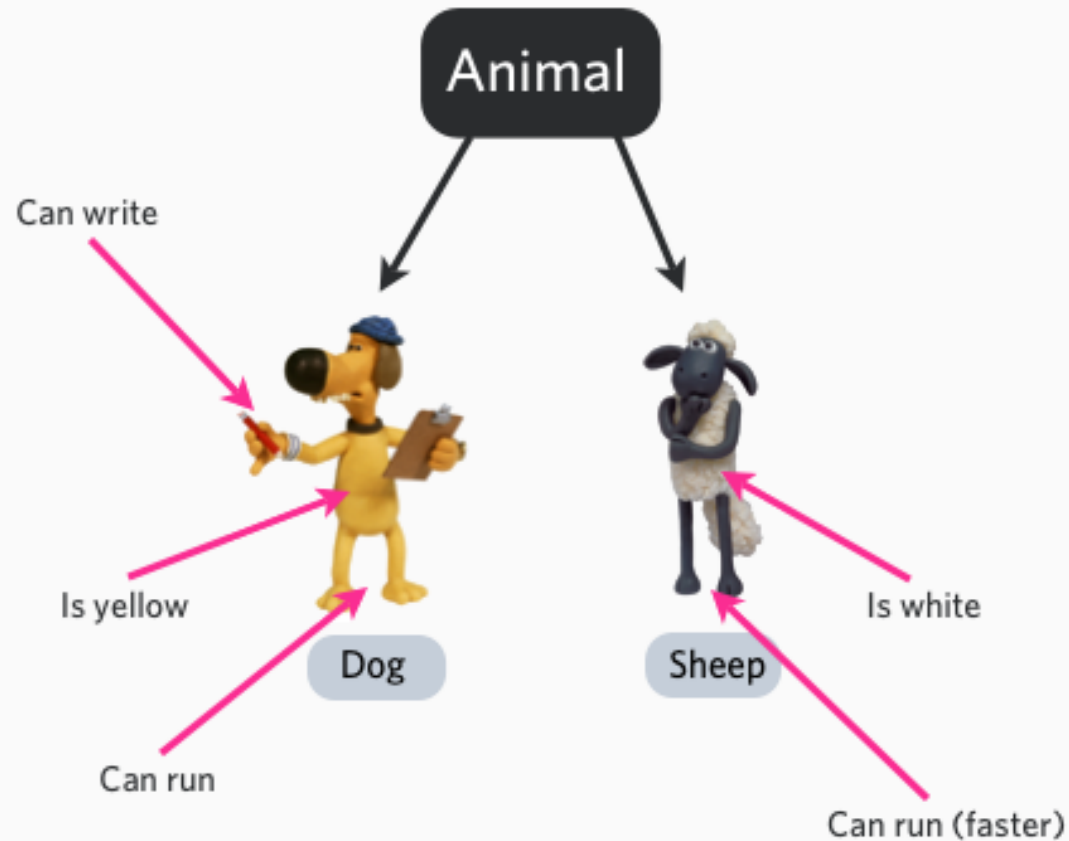
Windows
Fanboys

Linux
Fanboys



Why Python (13)?

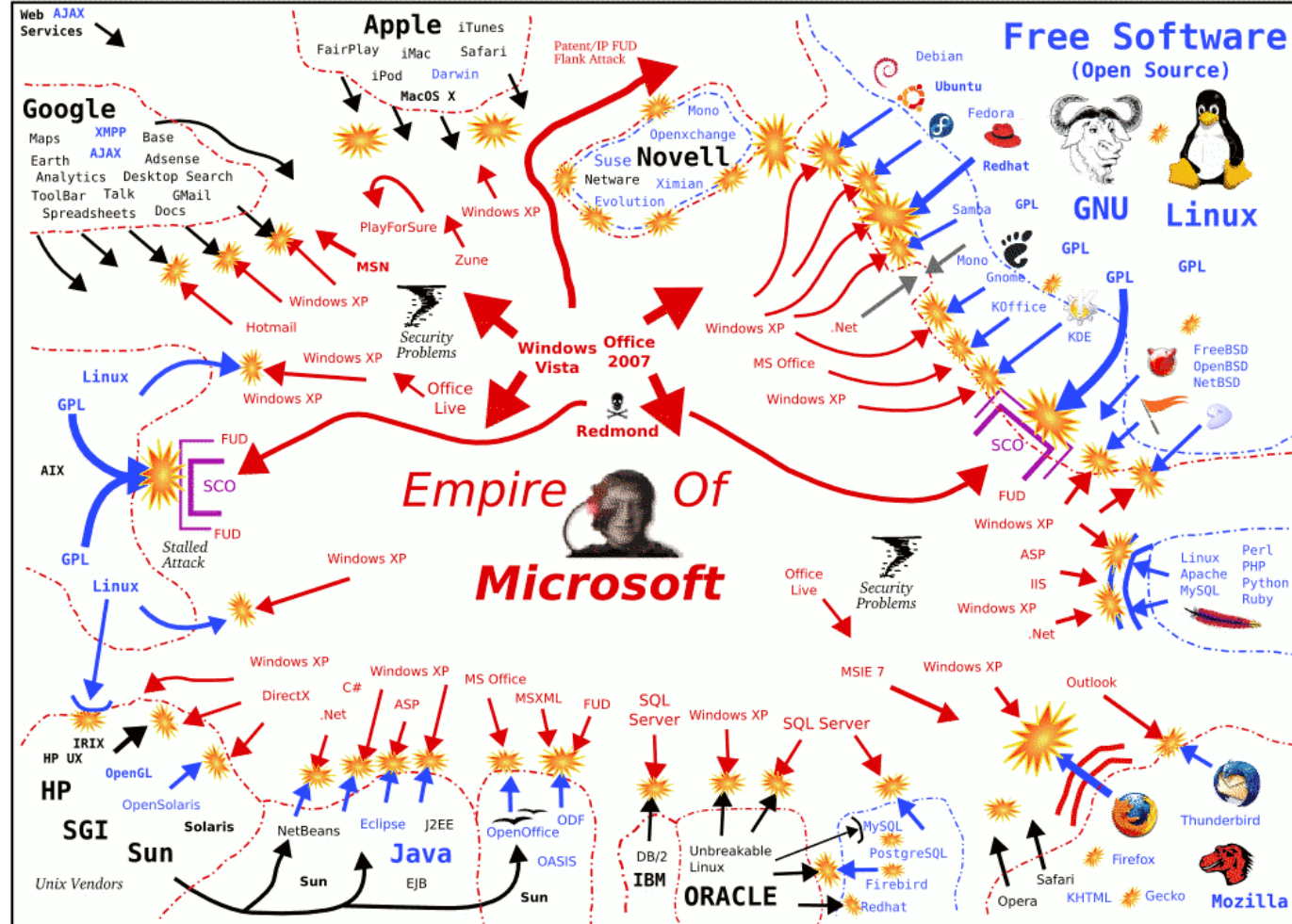
- オブジェクト指向 (すべてがオブジェクト)



Why Python (14)?

- しかもフリーソフト

Software Wars http://mshiltonj.com/software_wars/ All trademarks are property of their respective owners. Inspired by Andy Tai <http://atai.org/>
2006-12-28 Copyright 2006 - Steven Hilton <mshiltonj@gmail.com> Permission to copy is granted if copyright notice is preserved.



Drawn by
Steven Hilton in
2006

Why Python (15)?

- インストールが簡単
Python IDLE付き (Windows, Mac, Linux)
Batteries Included!



Why Python (16)?

- 追加モジュールのインストールも簡単

Anaconda

<https://store.continuum.io/cshop/anaconda/>

無料版で十分

100以上の便利なモジュールを含む

Mac, Linux, Windowsをサポート

最近ではPoetryで仮想環境がおすすめ



Why Python (17)?

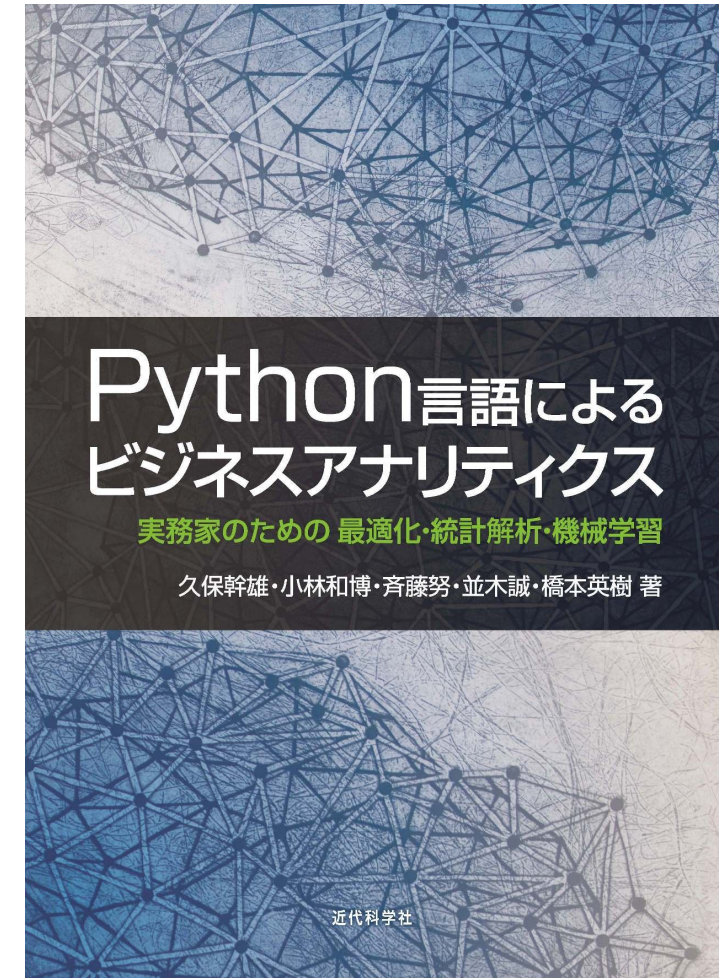
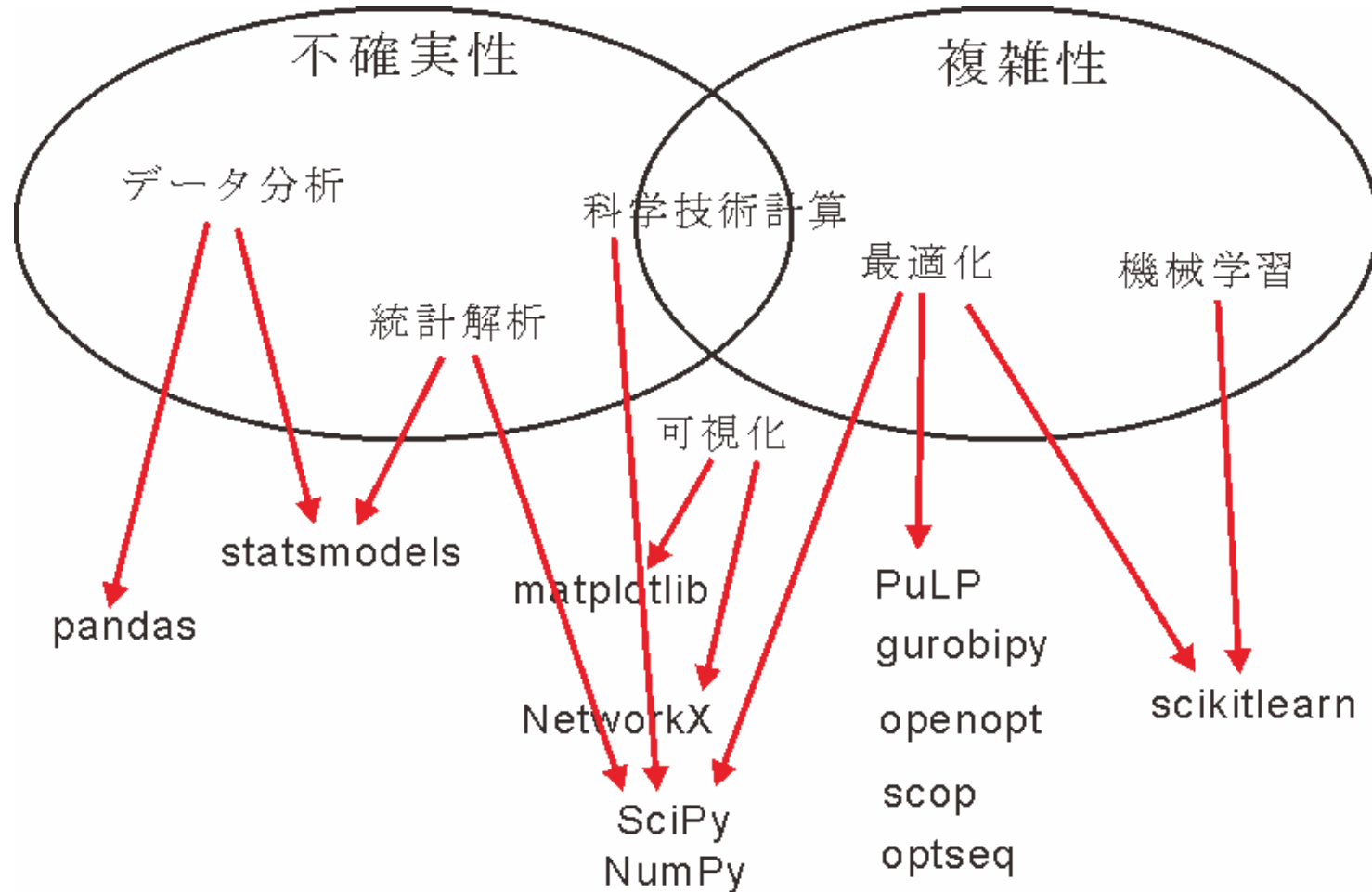
- CやFORTRANとの接続が容易
- 過去のプログラム遺産を再利用するための糊の役目
- 高速化が必要な部分を C,FORTRAN で、インターフェイスは Pythonで



Why Python (18)?

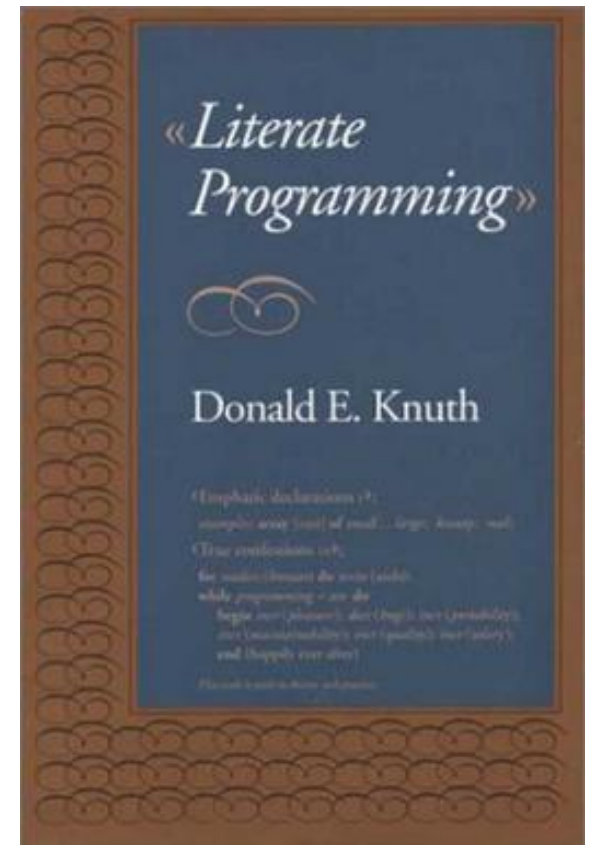
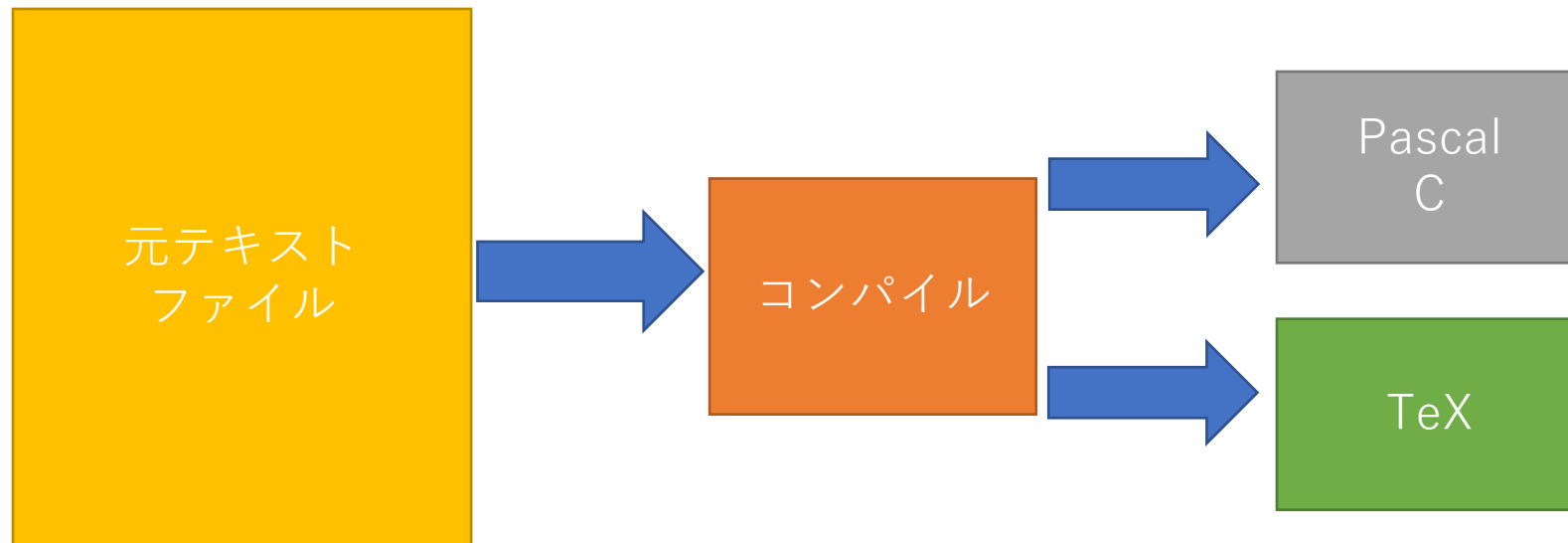


- ビジネス解析のためのモジュールが豊富



Jupyter+nbdevで文芸的プログラミング

- Knuthが1984年に提案したプログラミングのパラダイム
- 文芸的プログラミングツール WEB (CWEB)



Jupyter Lab.

このままでも文芸的プログラミング => 隠したいコードやメモ, ライブラリ自動生成

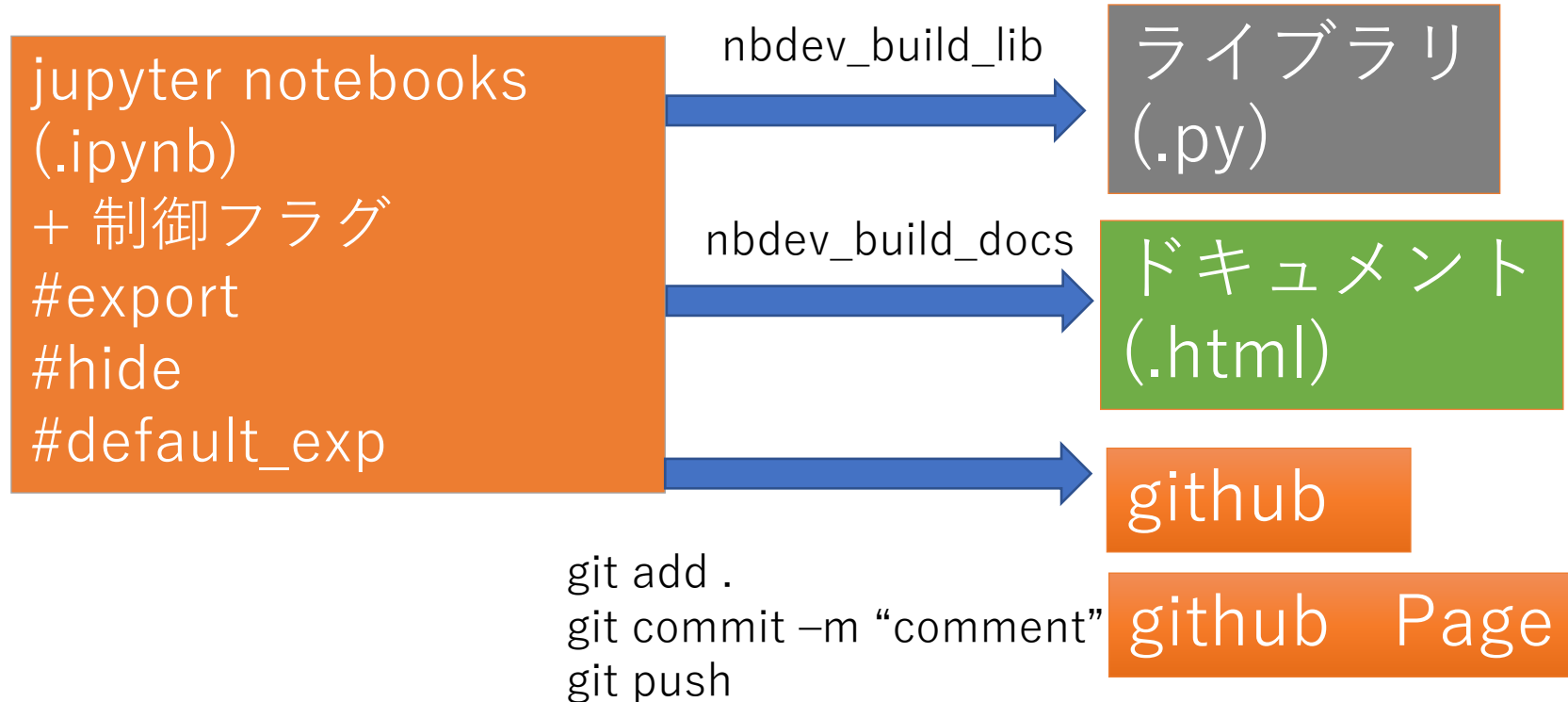
The screenshot shows the Jupyter Lab interface with a notebook titled "03inventory.ipynb". The notebook contains the following cells:

- Markdown Cell:** A title cell containing the text "在庫最適化モジュール `optinv` と安全在庫配置システム MESSA" and a subtitle "在庫最適化と安全在庫配置システム MESSA (MEta Safety Stock Allocation system)". A green box labeled "Markdownセル" points to this cell.
- Code Cell 1:** A cell with the code `#default_exp optinv`.
- Code Cell 2:** A cell with the code `#hide` followed by `%reload_ext autoreload`, `%autoreload 2`, and `%matplotlib inline`.
- Code Cell 3:** A cell with the code `#export` followed by a list of imports: `import warnings`, `import random`, `import string`, `import datetime`, `import math`, `import pickle`, `from collections import OrderedDict, defaultdict`, `import xml.etree.ElementTree as ET`, `import plotly.graph_objs as go`, `import plotly.express as px`, `import plotly`, `import chart_studio.plotly as py`, `from plotly.subplots import make_subplots`, `from scipy.stats import truncnorm`, `from scipy.stats import norm`, `import scipy.stats as st`, `import pandas as pd`, `import colorlover as cl`, `import numpy as np`, `import networkx as nx`, `from IPython.display import Image, YouTubeVideo`, `import sys`, and `sys.path.append('..')`. A grey box labeled "コードセル" points to this cell.

The interface includes a file browser on the left, a menu bar at the top, and a status bar at the bottom showing "Saving completed" and "Mode: Edit Ln 4, Col 14 03inventory.ipynb".

nbdev

- 深層学習パッケージ fastai の開発のためのツール
<https://nbdev.fast.ai/>



nbdev で公開中のプロジェクト (1)

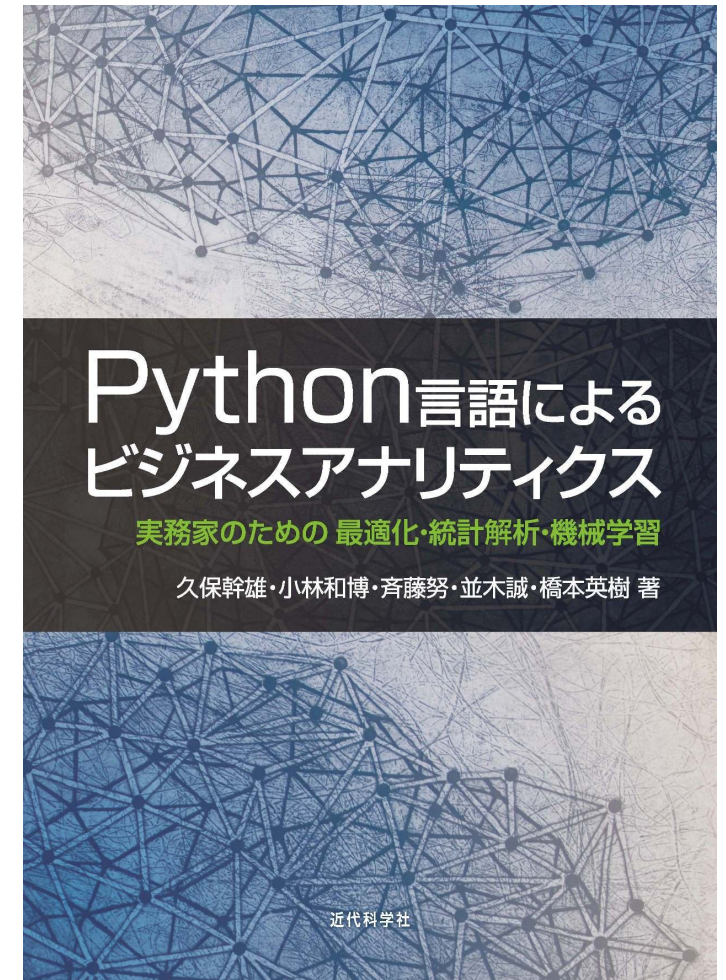
- Python言語による実務で使える100+の最適化問題
<https://mikiokubo.github.io/opt100/>



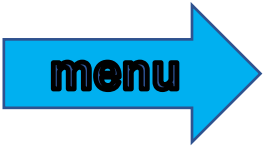
nbdev で公開中のプロジェクト (2)

- アナリティクス (データサイエンス) 練習問題集

<https://mikiokubo.github.io/analytics/>



nbdevの利点と弱点



- 利点
 - 本にもなる (らしい)
 - 並列自動テスト (私はオフにしている)
 - pypiとcondaに自動アップロード
- 弱点
 - 導入が面倒
 - バグが多い
 - 開発が粗い (開発者が気まぐれ)

Poetry

- <https://python-poetry.org/>
- 仮想環境 + パッケージ管理
- パッケージの依存関係の処理
- パッケージの公開
- (pipenvよりは) 高速

PYTHON PACKAGING AND DEPENDENCY MANAGEMENT MADE EASY

Poetry

Pyproject.tomlの例

```
[tool.poetry.dependencies]
python = "^3.8"
uvicorn = "^0.12.3"
fastapi = "^0.61.2"
pymongo = "^3.11.1"
mongoengine = "^0.21.0"
dnspython = "^2.0.0"
colorlover = "^0.3.0"
pandas = "^1.1.4"
matplotlib = "^3.3.3"
networkx = "^2.5"
numpy = "^1.19.4"
plotly = "^4.13.0"
```

Fastai

fast.ai

Making neural nets
uncool again

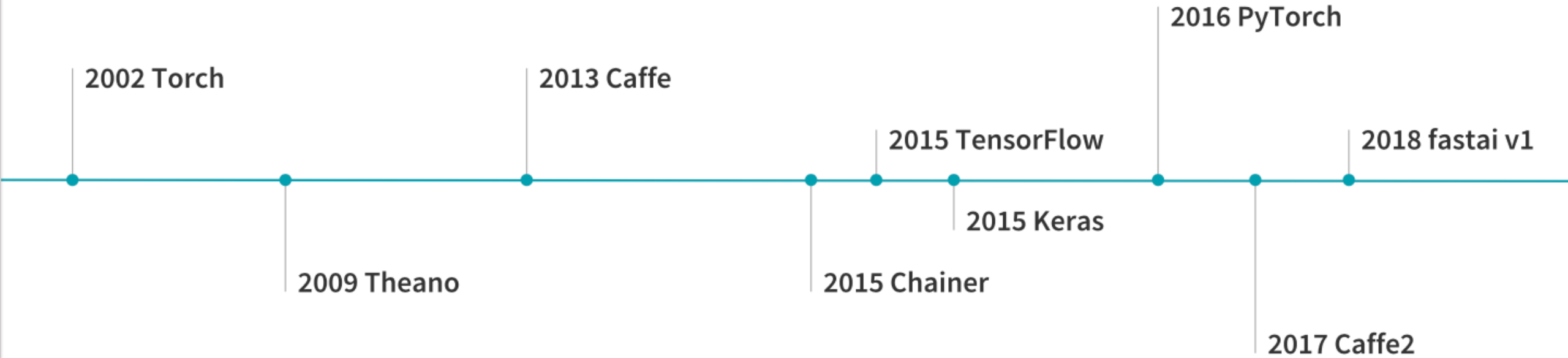
- <https://docs.fast.ai/>
- PyTorchのラッパ
- 短いコードで深層学習
- 無料の講義ビデオ付き
- (割と) 新しい手法が実装済み
- 画像, 自然言語処理, 表モデル, 協調フィルタリング
- 自分でアーキテクチャを設計したいときにはPyTorch (ソースを改変)
- 強化学習はなし
- データサイエンス練習問題集
<https://mikiokubo.github.io/analytics/14fastai.html>

fastai



Deep Learning Library

Brief History of DL Frameworks



Vision

1. import fastai

```
from fastai.vision import *
```

2. read MNIST data

```
mnist = untar_data(URLs.MNIST_TINY)  
tfms = get_transforms(do_flip=False)
```

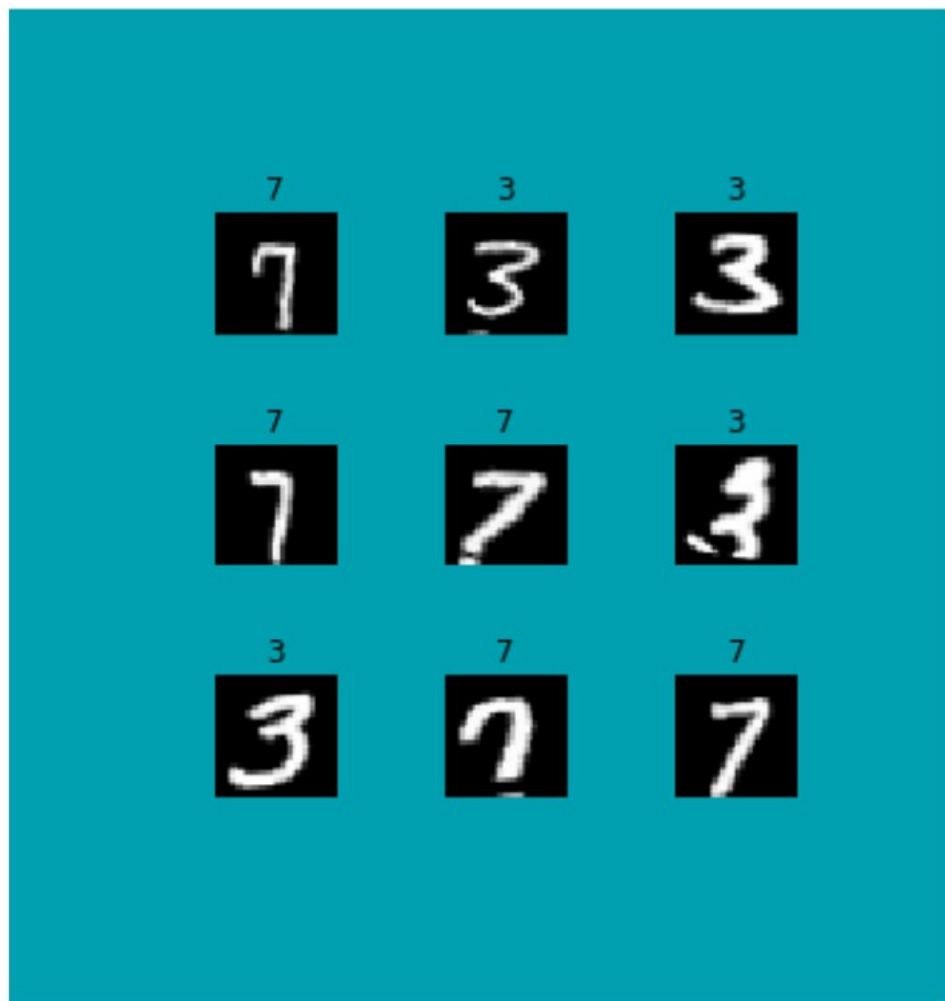
3. prepare data bunch

```
Data = (ImageItemList.from_folder(mnist)  
        .split_by_folder().label_from_folder()  
        .transform(tfms, size=32)  
        .databunch()  
        .normalize(imagenet_stats))
```

4. show data

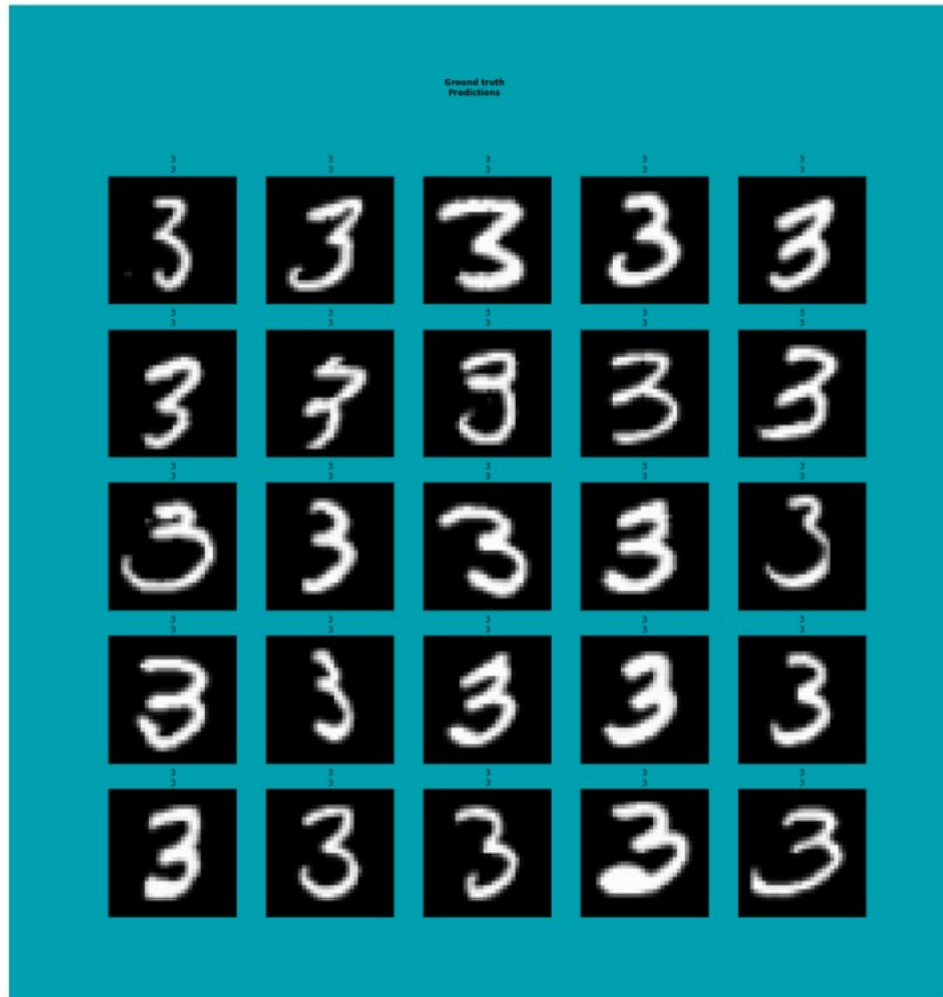
```
data.show_batch(rows=3, figsize=(4,4))
```

Show data



```
data.show_batch(rows=3, figsize=(4,4))
```

Train & show results



```
learn = create_cnn(data, models.resnet18, metrics=accuracy)
learn.fit_one_cycle(1, 1e-2)
```

```
learn.show_results()
```

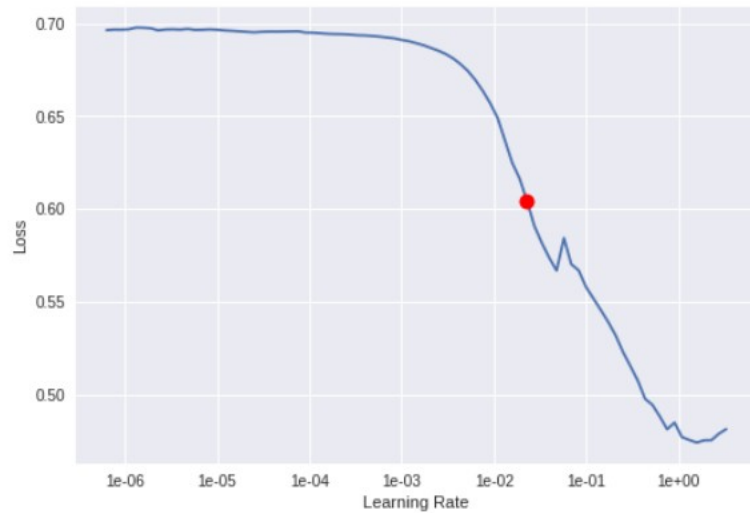
Make another model

```
model = simple_cnn((3,16,16,2))  
learn = Learner(data,model,metrics=[accuracy])
```

```
learn.summary
```

```
opt_func=functools.partial(<class 'torch.optim.adam.Adam'>, betas=(0.9, 0.99)), loss_func=<fastai.layers.FlattenedLoss object at  
0x7fa5f5fd2080>, metrics=[<function accuracy at 0x7fa5f70738c8>], true_wd=True, bn_wd=True, wd=0.01, train_bn=True,  
path=PosixPath('/root/.fastai/data/mnist_sample'), model_dir='models', callback_fns=[<class 'fastai.basic_train.Recorder'>],  
callbacks=[], layer_groups=[Sequential(  
  (0): Conv2d(3, 16, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))  
  (1): ReLU(inplace)  
  (2): Conv2d(16, 16, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))  
  (3): ReLU(inplace)  
  (4): Conv2d(16, 2, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))  
  (5): ReLU(inplace)  
  (6): AdaptiveAvgPool2d(output_size=1)  
  (7): Lambda()
```

Find a good learning rate

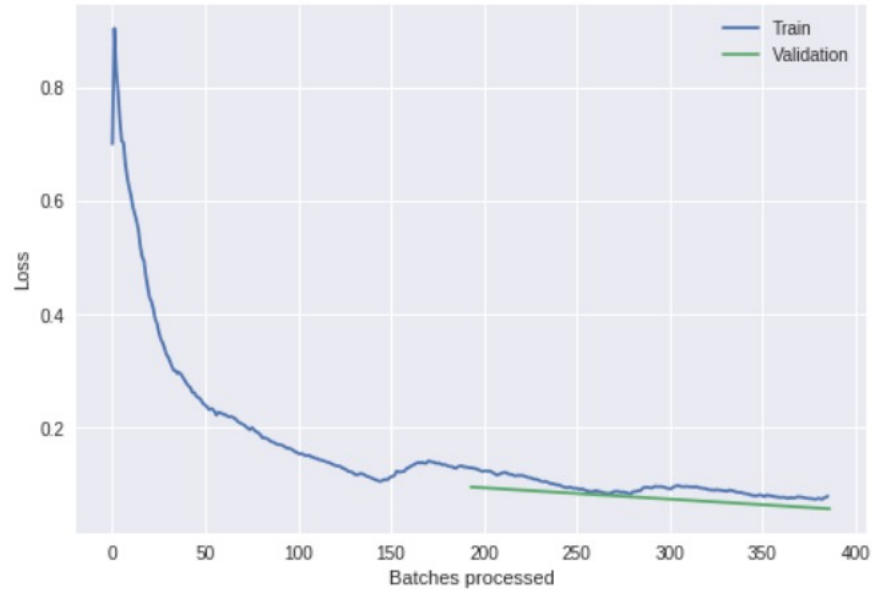


`learn.lr_find()`

`learn.recorder.plot()`

5e-2 is a good choice !

Train



- **Train 2 epochs using lr=0.05**

```
learn.fit(2,5e-2)
```

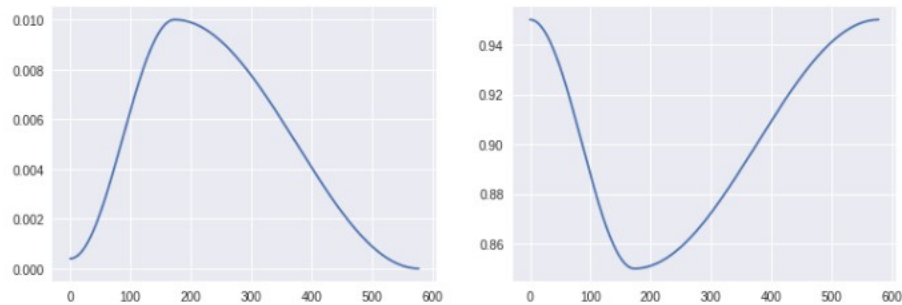
```
epoch train_loss valid_loss accuracy
```

```
1 0.129588 0.095395 0.968106
```

```
2 0.079258 0.056953 0.981845
```

```
learn.recorder.plot_losses()
```

Use fit-one-cycle



- **Train 3 epochs using fit-one cycle method**

```
learn.fit_one_cycle(3,max_lr=5e-2)
```

```
>>>
```

```
epoch train_loss valid_loss accuracy
```

```
1 0.104979 0.059402 0.976938
```

```
2 0.034898 0.020833 0.992149
```

```
3 0.020161 0.016893 0.992149
```

```
# lr (left), momentum (right)
```

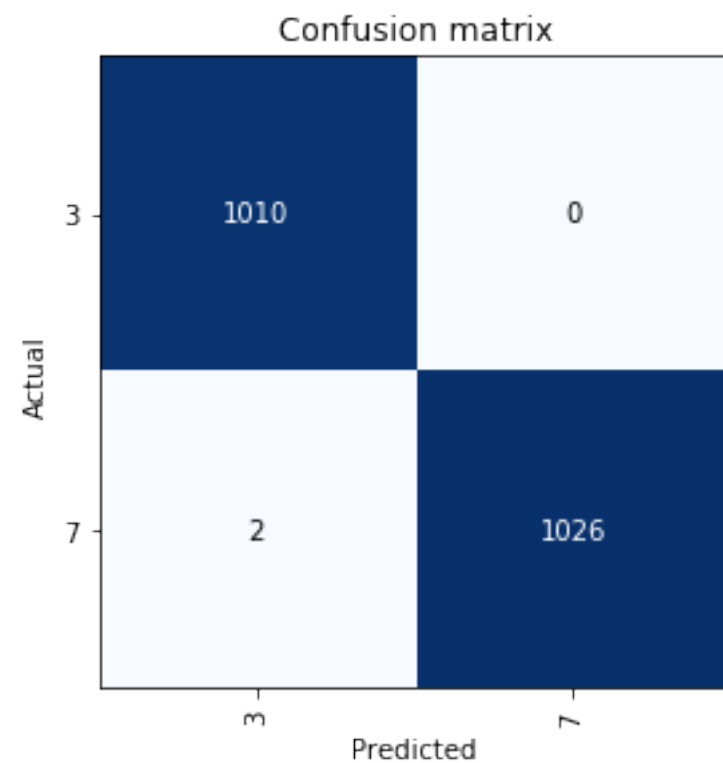
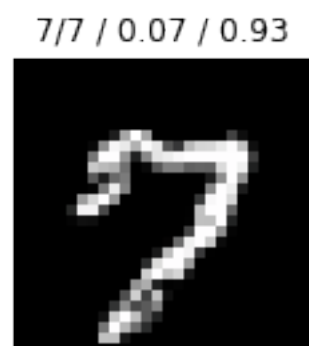
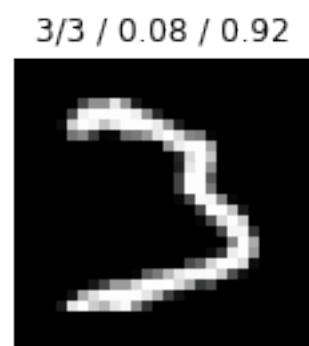
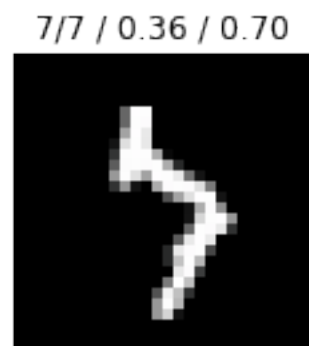
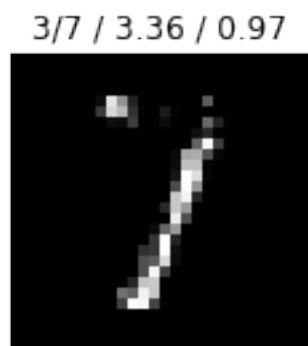
```
preds,y,losses = learn.get_preds(with_loss=True)
```

```
interp = ClassificationInterpretation(data, preds, y, losses)
```

```
interp.plot_top_losses(9, figsize=(7,7))
```

```
interp.plot_confusion_matrix()
```

Prediction/Actual/Loss/Probability



Pycaret



- <https://pycaret.org/>
- (自動) 機械学習
- 前処理 + モデル構築 + 訓練 + アンサンブル + デプロイ
- 回帰, 分類, クラスタリング, 異常検知, 自然言語処理, アソシエーション・ルール・マイニング
- 結果の可視化
- モデルの解釈 (SHAP: SHapley Additive exPlanations)
- データサイエンス練習問題集
<https://mikiokubo.github.io/analytics/19pycaret.html>

Prophet

PROPHET

Forecasting at scale.

Prophet is a forecasting procedure implemented in R and Python. It is fast and provides completely automated forecasts that can be tuned by hand by data scientists and analysts.

- <https://facebook.github.io/prophet/>
- (自動) 需要予測
- ベイズ推論 (pystan利用)
- データサイエンス練習問題集
<https://mikiokubo.github.io/analytics/15forecast.html>

(自動) 機械学習(Auto ML)と
深層学習(DL)と
ベイズ推論
を用いた予測手法について

東京海洋大学

久保幹雄

(量的) 予測と回帰

- 予測 (時系列) モデル

過去のデータが与えられたとき, 未来の値を予測

$$\hat{y}_t = f(y_{t-1}, y_{t-2}, \dots, y_1)$$

- 回帰モデル

訓練データ集合が与えられたとき, 検証データ集合における損失関数を最小にするパラメータを推定

$$\hat{y} = g(x, \theta)$$

↑ ↑
従属変数 独立変数
(目的変数) (説明変数)

時系列モデル

- ARIMAモデル

$$y_t - y_{t-d} = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \dots + \phi_p y_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \dots + \theta_q \varepsilon_{t-q}$$

integrated

autoregressive

moving average

- 3次指数平滑(Holt-Winter)モデル

$$s_0 = x_0$$

$$s_t = \alpha(x_t - c_{t-L}) + (1 - \alpha)(s_{t-1} + b_{t-1})$$

$$b_t = \beta(s_t - s_{t-1}) + (1 - \beta)b_{t-1}$$

$$c_t = \gamma(x_t - s_{t-1} - b_{t-1}) + (1 - \gamma)c_{t-L}$$

$$y_{t+m} = s_t + mb_t + c_{t-L+1+(m-1) \bmod L}$$

手法の比較

機械学習
(深層学習を含む)

ベイズ推論
(確率的プログラミング)



明確な推論
(ホワイトボックス)

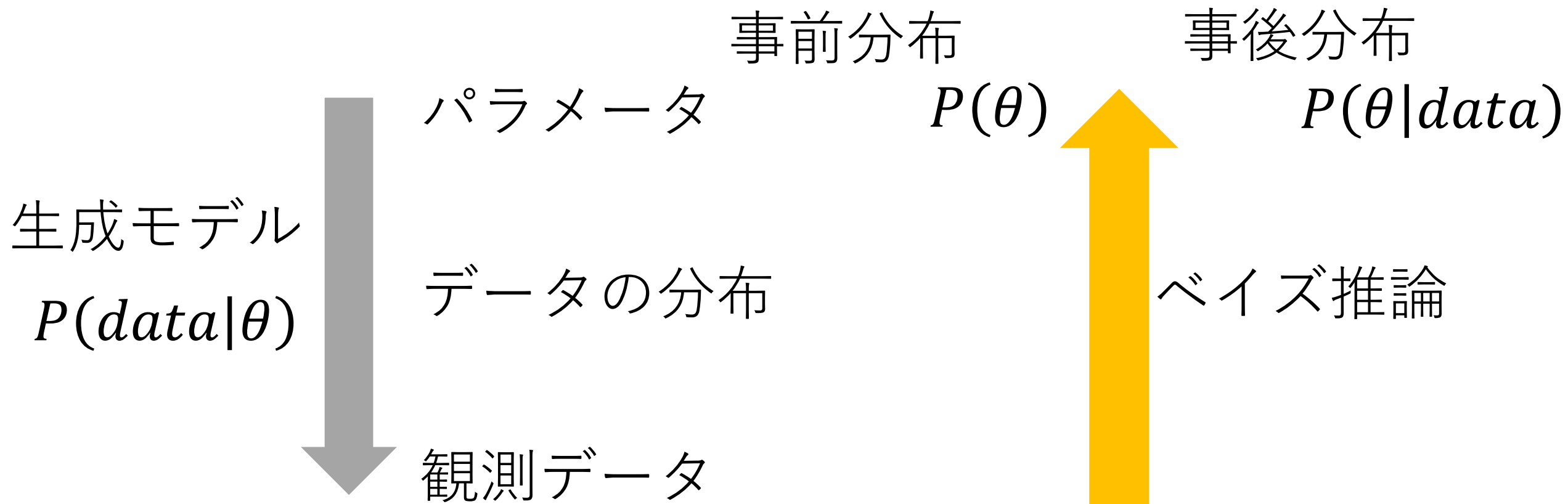


完全な柔軟性



分布まで考慮した
不確実性

確率的プログラミング



ベイズの公式 $P(\theta|data) \propto P(data|\theta)P(\theta)$

確率的プログラミングの手法と実装

- マルコフ連鎖モンテカルロ法
(Markov Chain Monte Carlo methods : MCMC)
= データに合った近似確率分布を生成するシミュレーション
- PyMC3 (4)
- Stan

予測に特化したとして実装として Prophet (Facebook)

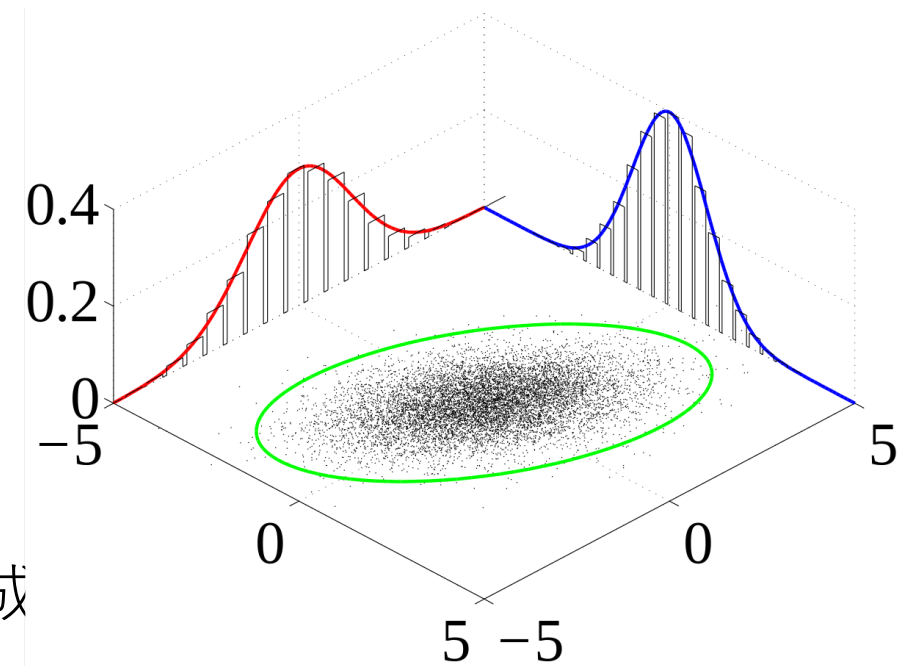
=> Stanを用いた予測モデル

MCMC

- 適当な初期パラメータ θ
- 収束するまで以下を反復
 - 提案分布にしたがい新しい点 θ^{new} を生成
 - 推移確率を計算

$$q = P(\theta^{new} | data) / P(\theta | data)$$

- 推移確率が1以上なら確率1で, そうでなければ確率 q で $\theta = \theta^{new}$



Prophetのモデル(一般化加法モデル)

$$y_t = g_t + s_t + h_t + \epsilon_t$$

y_t : 予測値

g_t : 傾向変動 (変化点ありの線形・ロジスティック曲線)

s_t : 季節変動 (フーリエ級数)

h_t : 休日 (イベント) 効果 (特定日を指定して分布を予測)

ϵ_t : 誤差項

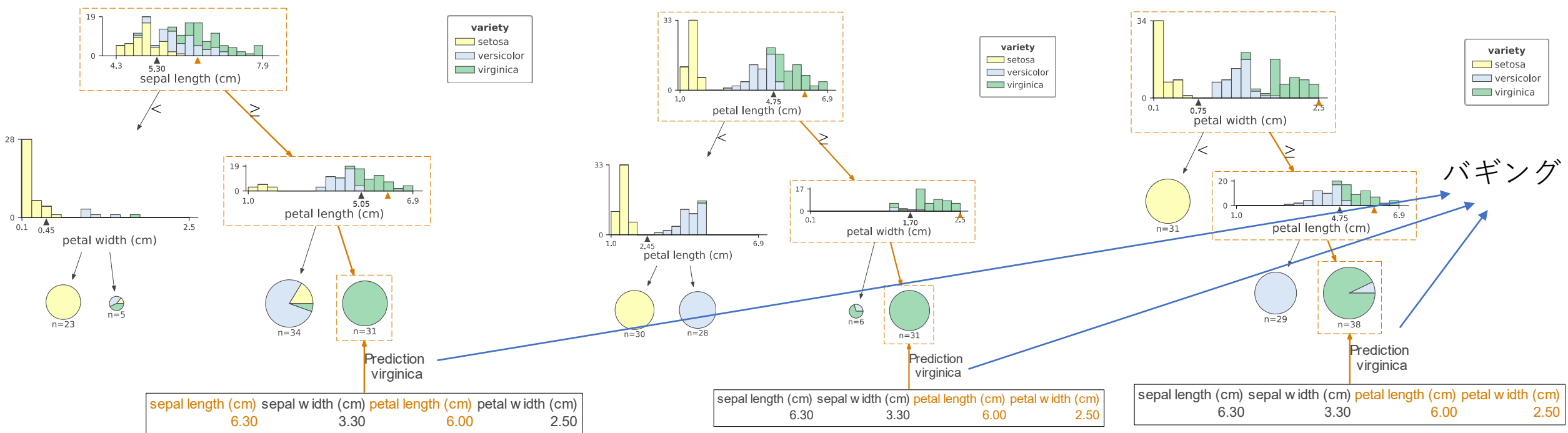
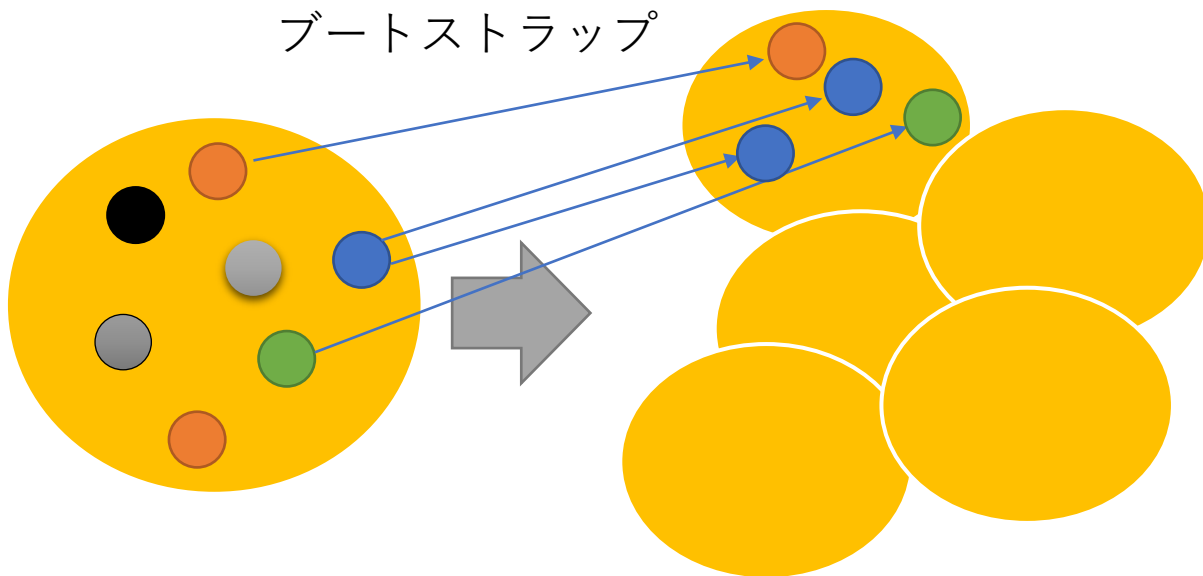
機械学習による予測（回帰）

- 決定木のアンサンブル
 - バギング = bootstrap aggregating（ランダム森, Extra木）
 - ブースティング（適応型ブースティング, 勾配ブースティング(GBM), Light GBM, Extreme GBM, カテゴリーブースティング）
- 深層学習
 - = 複雑な関数を任意の精度で近似して出力するアーキテクチャ
 - 畳み込みニューラルネット（2次元畳み込み）
 - LSTM（Long Short Term Memory）
= 実用的な回帰型ニューラルネットワーク(Recurrent NN)
 - 埋め込みニューラルネット

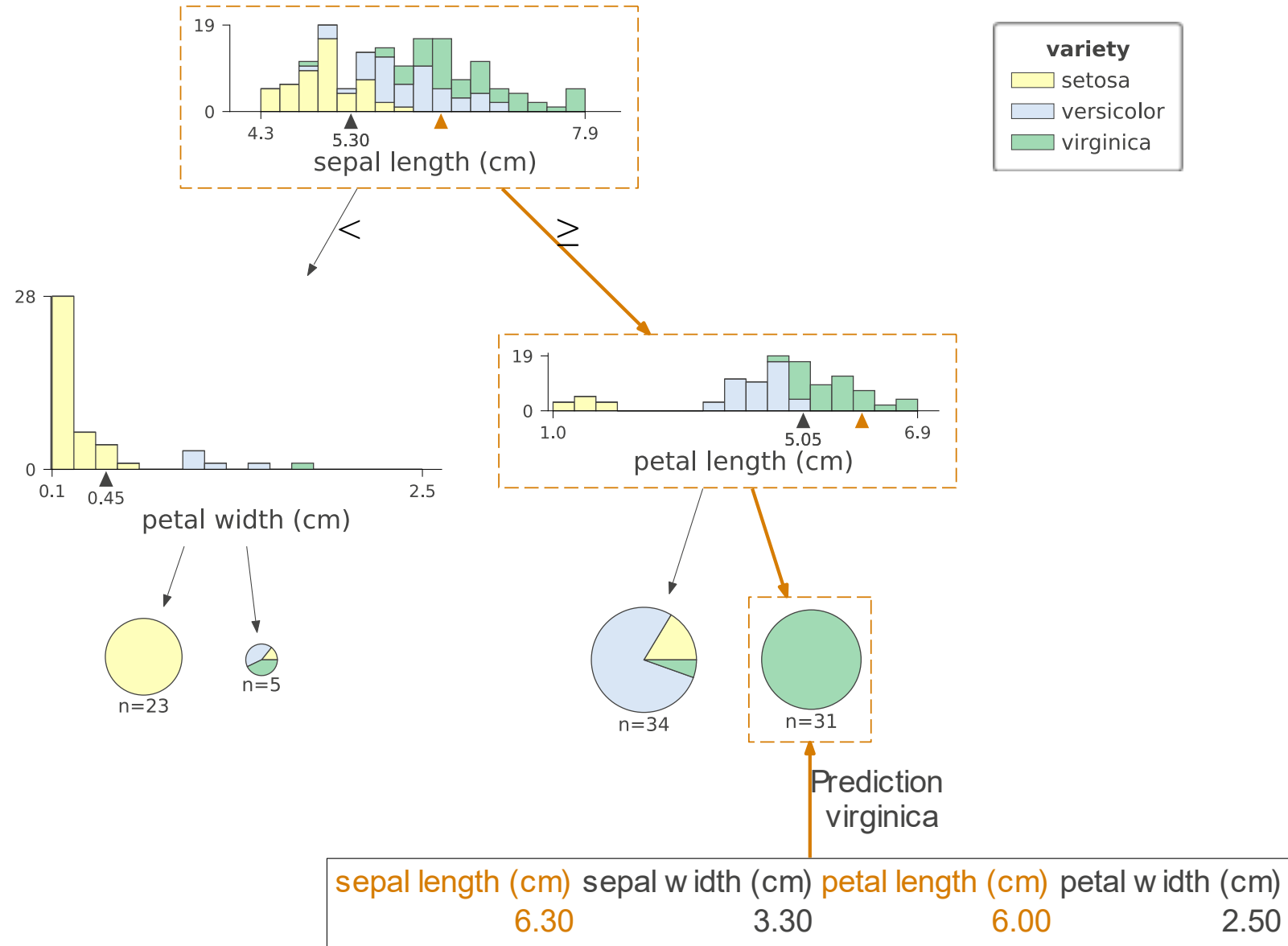
ランダム森

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5	3.6	1.4	0.2	Iris-setosa

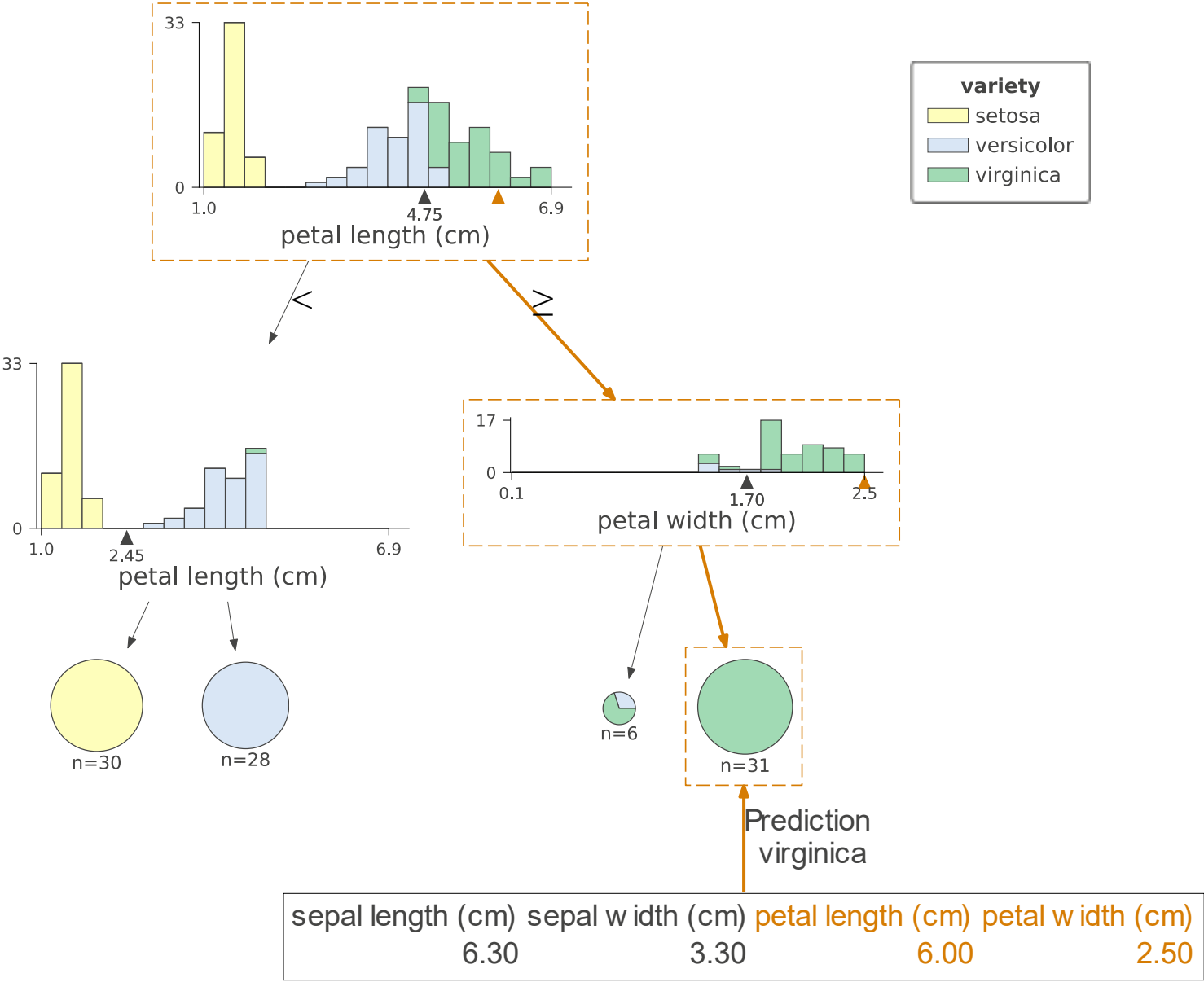
ブートストラップ



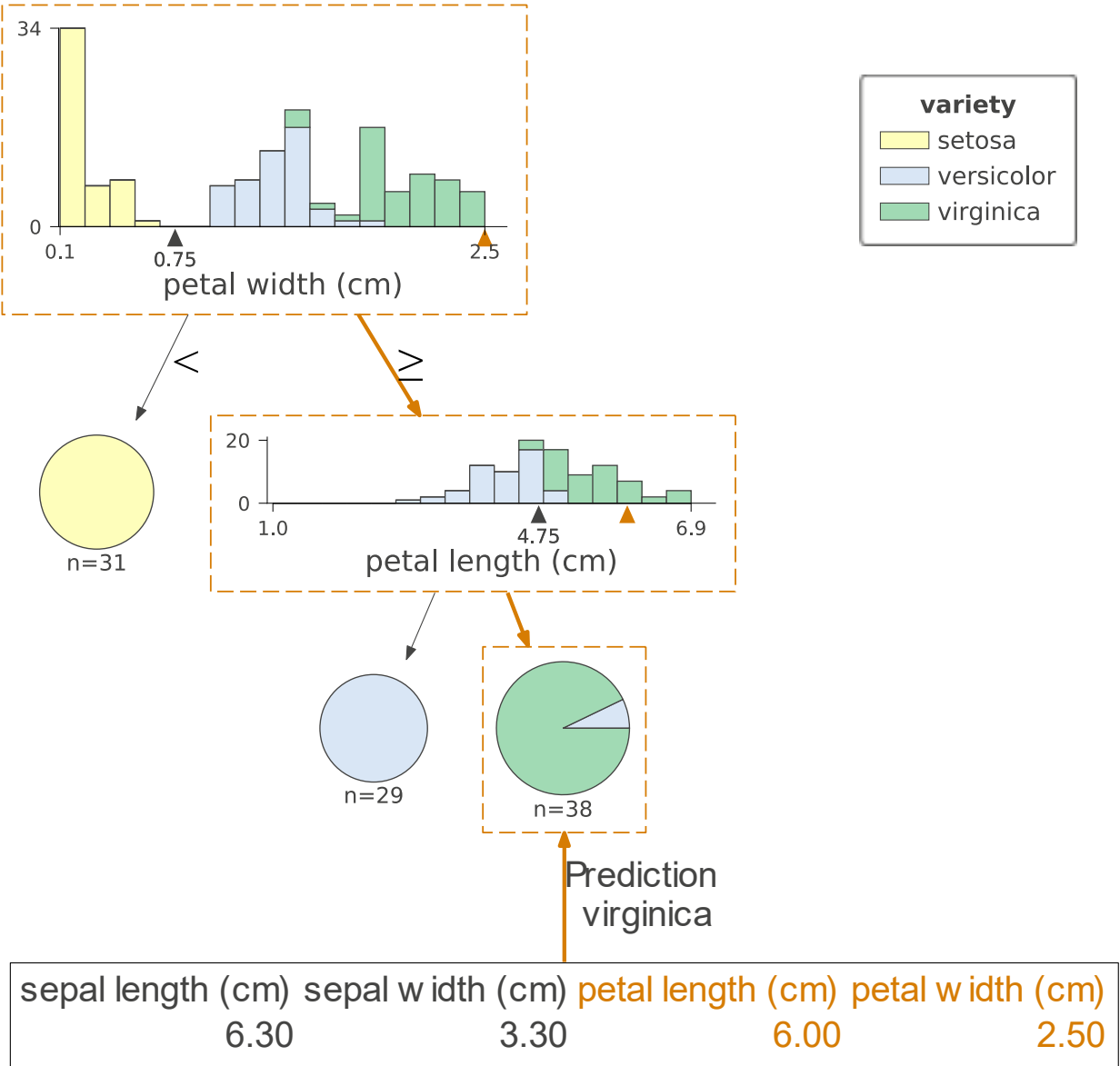
決定木 1



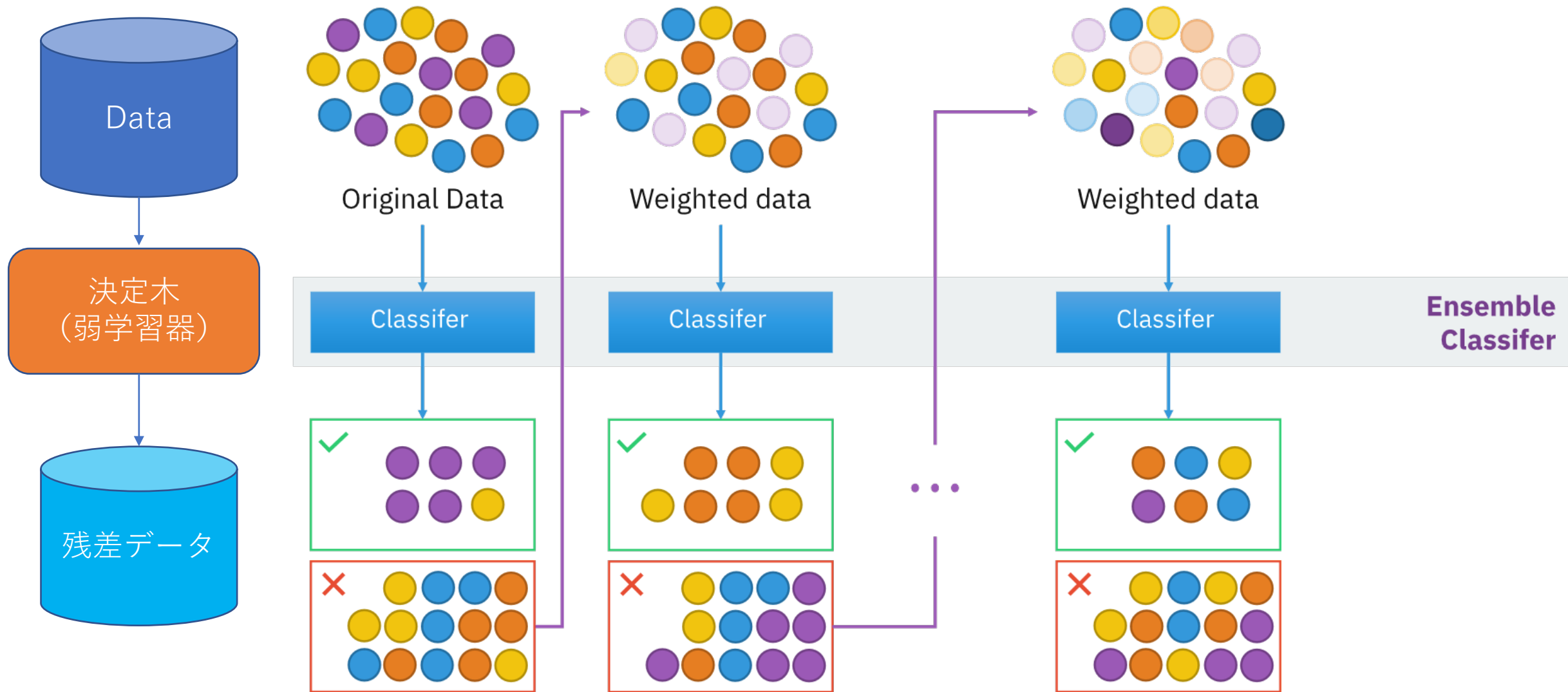
決定木 2



決定木 3



ブースティング

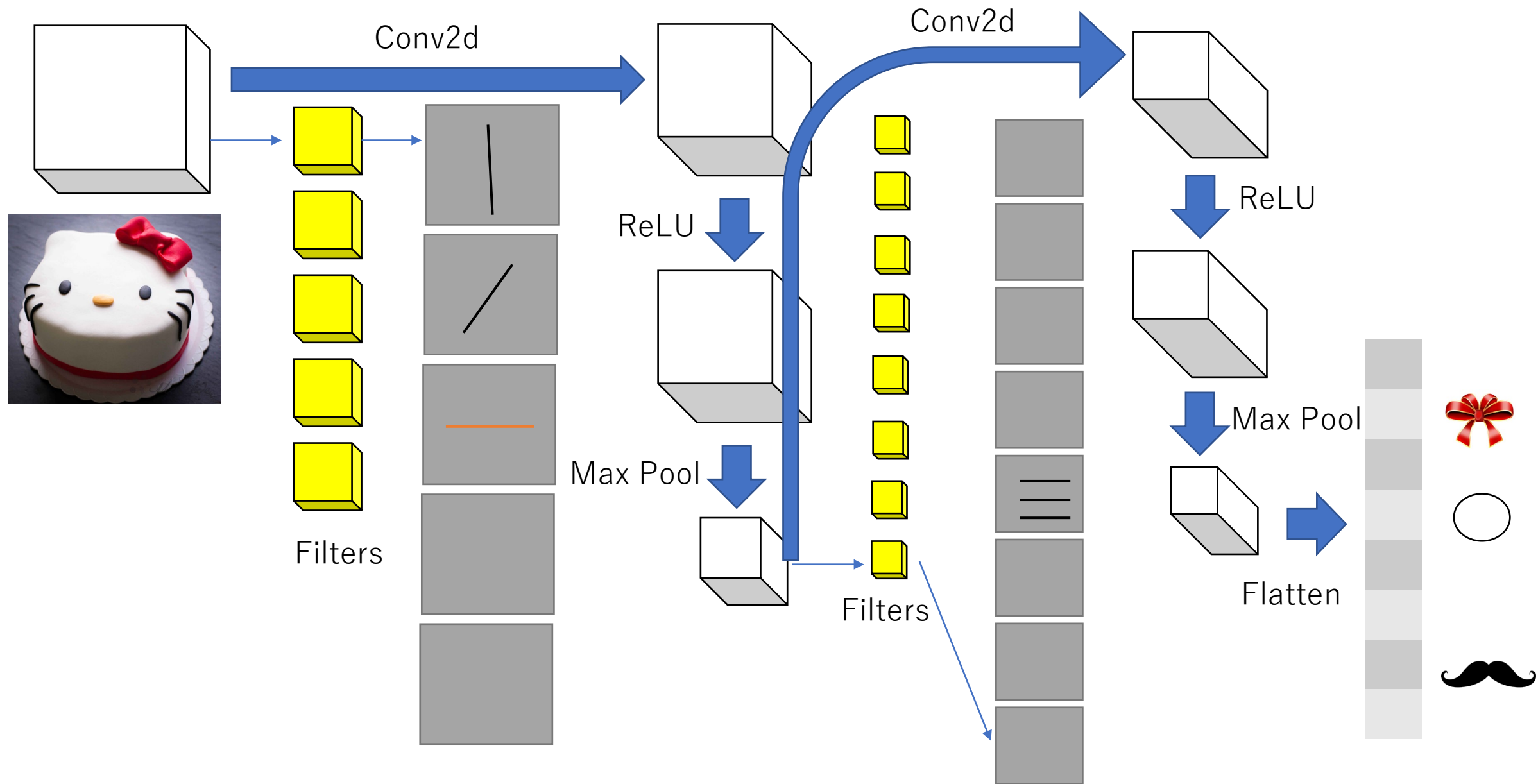


自動機械学習 Auto ML

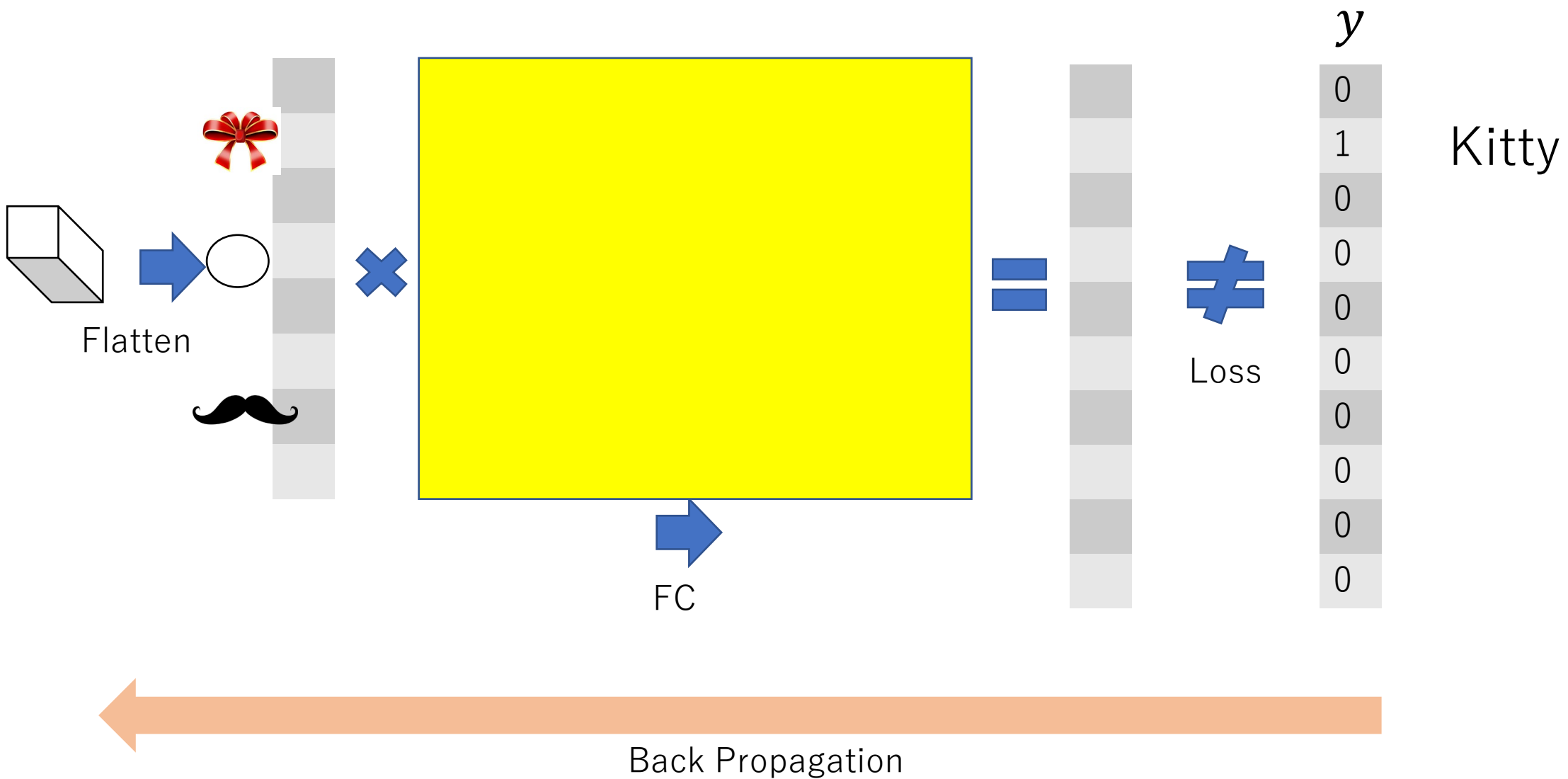
- たくさんの機械学習の手法を同時に行い良いものを選ぶ。
- 良いものをアンサンブル

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
et	Extra Trees Regressor	0.4636	0.4248	0.6334	0.9837	0.0705	0.0523	0.0440
gbr	Gradient Boosting Regressor	0.6601	0.7945	0.8653	0.9704	0.0910	0.0709	0.0100
xgboost	Extreme Gradient Boosting	0.6984	0.8608	0.9045	0.9675	0.0893	0.0698	0.0360
rf	Random Forest Regressor	0.7123	0.8845	0.9210	0.9667	0.0945	0.0750	0.0500
catboost	CatBoost Regressor	0.5846	0.9041	0.8906	0.9655	0.1063	0.0776	1.4960
ada	AdaBoost Regressor	0.9116	1.3416	1.1366	0.9485	0.1020	0.0903	0.0160
dt	Decision Tree Regressor	0.9474	1.4913	1.2042	0.9418	0.1032	0.0836	0.0060
lightgbm	Light Gradient Boosting Machine	1.1357	2.4086	1.5262	0.9037	0.1636	0.1345	0.0120

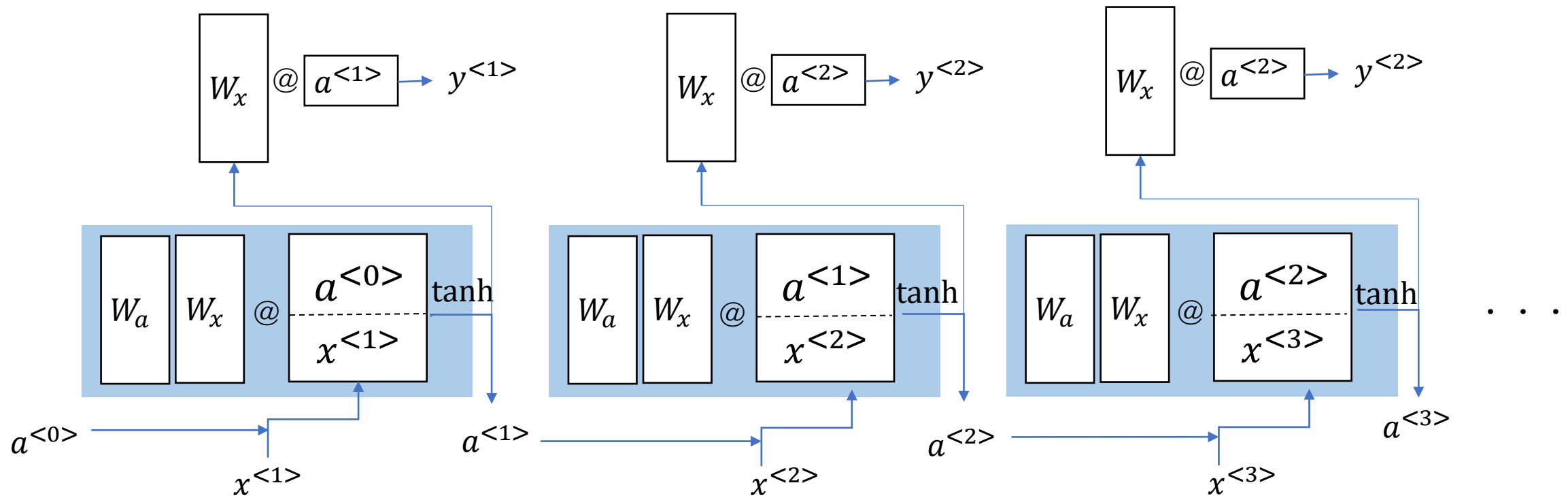
畳み込みニューラルネット (1)



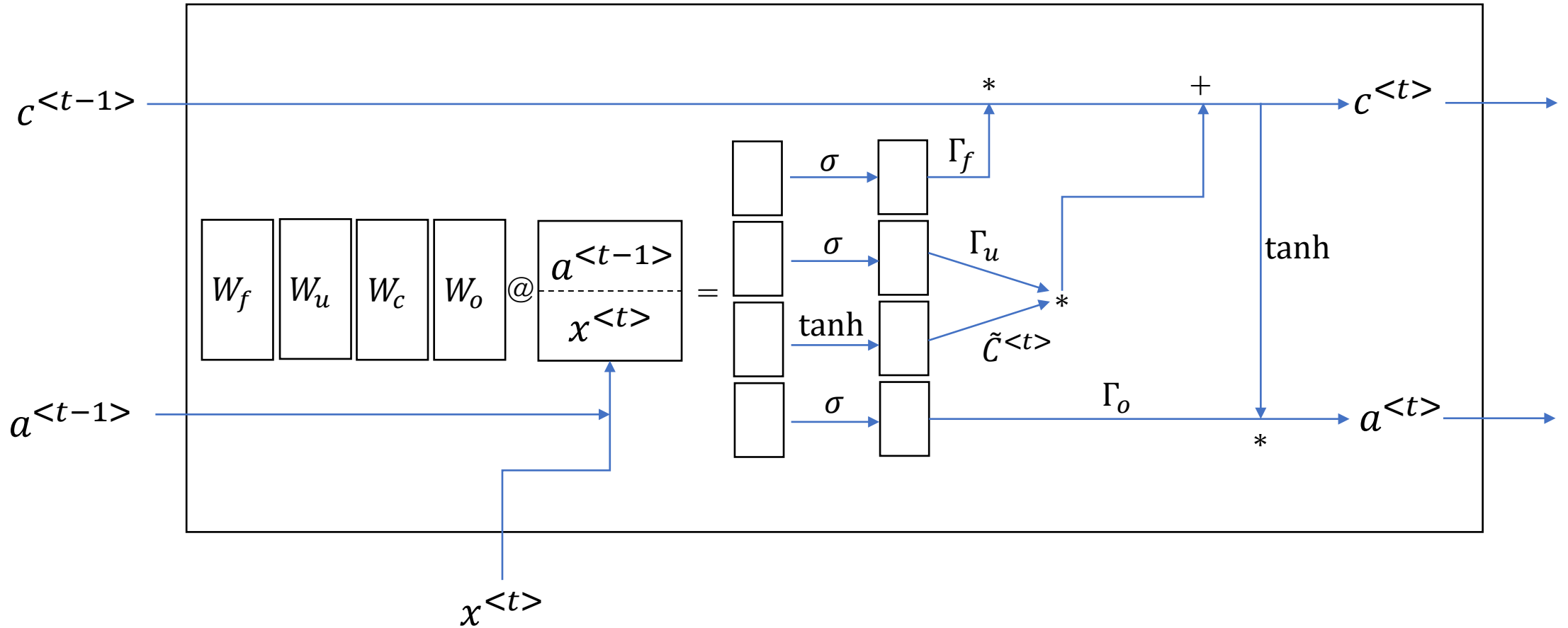
畳み込みニューラルネット (2)



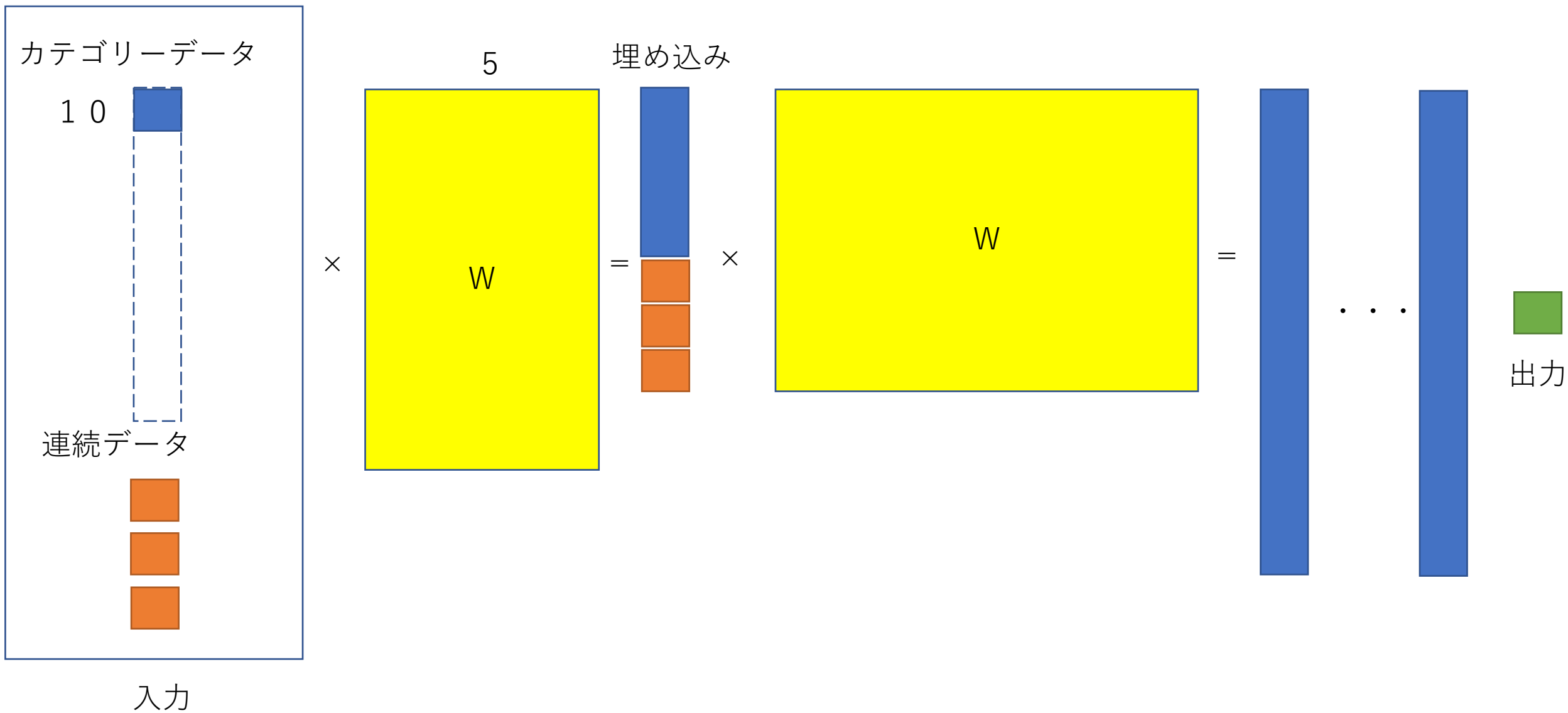
回帰型ニューラルネット



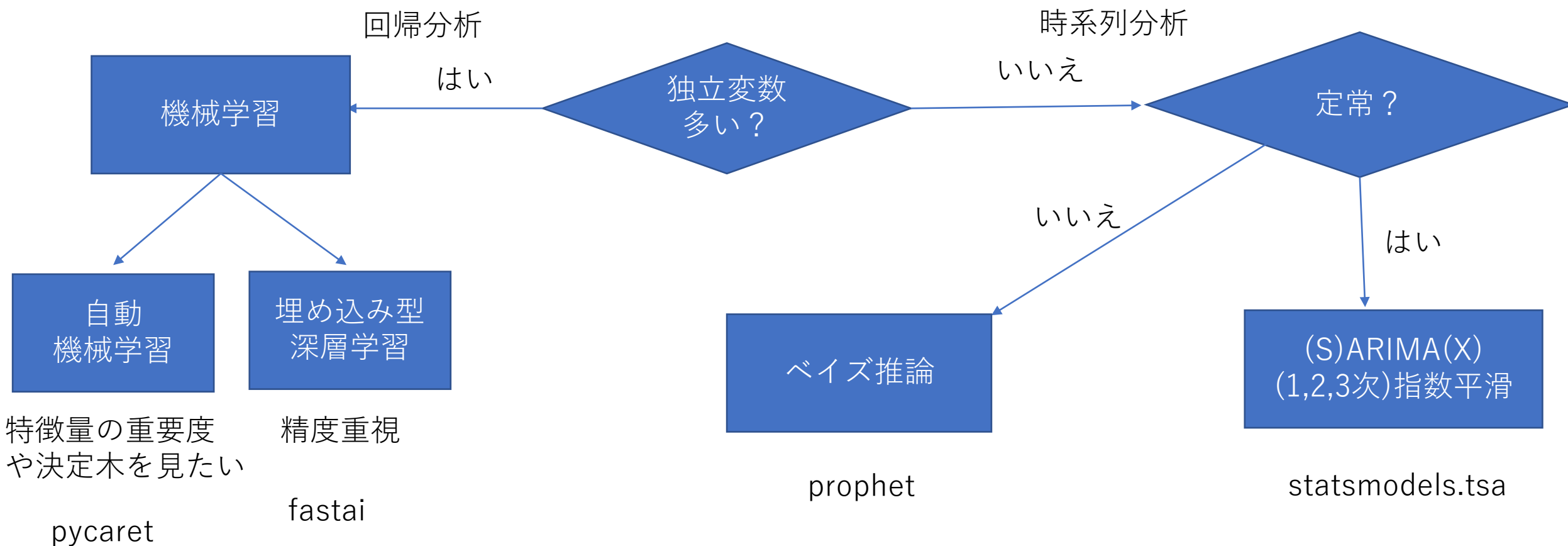
LSTM (時系列データに対する予測)



埋め込み層を用いた深層学習



使い分け



Gurobi



- <https://www.gurobi.com/>
- 混合整数最適化ソルバー
- 作成者: Zonghao Gu, Edward Rothberg, Robert Bixby
- 大規模な混合整数二次(錘)最適化をマルチコアで求解
- 商用(でもアカデミックフリー)
- (おそらく)最速
- Pythonインターフェイスあり
- (ほぼ)同じインターフェイスのオープンソース・パッケージ
mypulp (pulpのラッパ;ソルバーはCBC)

数理最適化ソルバーの性能向上

- CPLEX 1.2 (1991) -> CPLEX 11 (2007) : 29000倍
- Gurobi 1.0 (2009) -> Gurobi 9.0 (2019) : 59倍
- 合わせると. . . 170万倍
- 計算機の速度向上
59.7 Gflops/s (1993) -> 93.0 Pflops/s (2016)
- 合わせると. . . 2.2兆倍

以前は 7 万年かかっていた計算が 1 秒以下！

最適化の効果

- 人間（ベテラン）の作った解の10%程度の削減
- 物流費用はGDPの8%程度（途上国ではその倍以上）
- 例えば日本だとGDPは560兆円程度

最適化を使えば、物流費だけで年間4兆円削減可能！

サプライ・チェーンの費用はさらに大きい！

Streamlit

- <https://www.streamlit.io/>
- Pythonだけで簡単にWebアプリ
- プログラムを上から実行していただくだけ
(コールバック不要)
- 関数をキャッシュすることによる高速化 (デコレータ使用)
- github経由で簡単にデプロイ (無料)

The fastest way to build and share data apps

Streamlit turns data scripts into shareable web apps in minutes.
All in Python. All for free. No front-end experience required.

こういうのが簡単に作れる (DLで予測)

アプリ選択 (Select an App)

Deep Forecast

Select Language

日本語

予測する顧客名と製品名を選択

製品選択

A

顧客選択

仙台市

予測パラメータ

期間

1D

予測期間数

1

追加データ選択

promo_0 × promo_1 ×

検証パラメータ

検証開始期

2020/10/19

予測データ生成

深層学習

学習器生成

Deep Forecast

[Sample excel file](#)

Excelファイルアップロード

Drag and drop file here
Limit 200MB per file • XLSX

Browse files

需要データ

需要データアップロード

Drag and drop file here
Limit 200MB per file • CSV, TXT

Browse files

date	cust	prod	promo_0	promo_1	demand
2019-01-01	札幌市	A	0	0	0
2019-01-01	札幌市	B	0	0	1
2019-01-01	札幌市	C	0	0	0
2019-01-01	青森市	A	0	0	2
2019-01-01	青森市	B	0	0	14

1 to 5 of 21,900 |< < Page 1 of 4,380 > >|

[Download csv file](#)

データ探索

プロモーションデータ

プロモーションデータアップロード

Drag and drop file here
Limit 200MB per file • CSV, TXT

Browse files

date	promo_0	promo_1
2019-01-01	0	0
2019-01-02	1	0
2019-01-03	0	0
2019-01-04	0	0
2019-01-05	0	0

1 to 5 of 1,095 |< < Page 1 of 219 > >|

[Download csv file](#)

データ探索

demand	promo_0	Week	promo_1	Year	Month	Day	Dayofweek	Dayofyear	Is_month_end	Is_month_start	Is_quarter_end	Is_quarter_start	Is_year_end	Is_year_start	El
13	0	1	0	2019	1	1	1	1	false	true	false	true	false	true	15
19	1	1	0	2019	1	2	2	2	false	false	false	false	false	false	15
17	0	1	0	2019	1	3	3	3	false	false	false	false	false	false	15
12	0	1	0	2019	1	4	4	4	false	false	false	false	false	false	15
19	0	1	0	2019	1	5	5	5	false	false	false	false	false	false	15

1 to 5 of 730 |< < Page 1 of 146 > >|

[Download csv file](#)

データ探索

MIRO



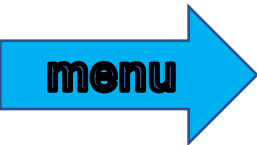
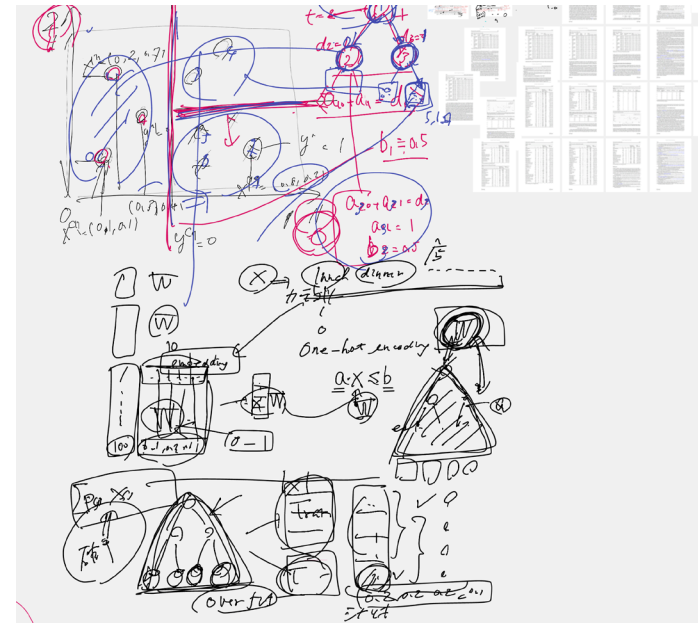
The online collaborative whiteboard platform to bring teams together, anytime, anywhere.

Start a whiteboard →

Free forever — no credit card required



- <https://miro.com/>
- オンライン・ホワイトボード（画面共有）
- コロナ禍でSPRINTを行うためのツール（zoomと併用）
- 授業にも便利（学生参加型の演習）
- アカデミック登録で無制限



Notion

- <https://www.notion.so/>
- ゼミ学生との研究共有
- 講義の簡易HP

データサイエンス演習

📄 [データサイエンス演習](#)

ゼミ生個別ページ

🗨️ [コロナ禍の避難計画](#)

🏠 [深層・機械学習を用いた ホテル予約件数の予測](#)

👤 [サプライ・チェーン途絶条件下でのリスク分析](#)

🚚 [AGV 最適パス設計のための最適化手法](#)

📄 [時間枠付きTSPの実験](#)

研究室ゼミ用

📄 [SCMOPT-zero間の連携システム](#)

📄 [実務における大規模配車計画問題](#)

📄 [SCIPとPuLPの比較実験](#)

📄 [賞味期限を考慮した動的価格・在庫最適化](#)

🔵 [ガスボンベの配送計画問題](#)

📄 [pyvisを用いたOptSeqの時間制約の可視化](#)

📄 [勤務表作成アプリ](#)

🟡 [回転を考慮した配送最適化](#)

📄 [scmopt0 on GCP](#)

🔪 [SCMOPTにValidationを追加し、fastapiでREST APIを作る](#)

📄 [サービスネットワーク設計をSCOPで解く](#)

📄 [予測システムの比較](#)

📄 [複数の製品・顧客に対して予測するML Ops](#)

📄 [大規模グラフに対する時刻依存最短路](#)

📄 [二次割当問題](#)

📄 [時間枠付き巡回セールスマン問題](#)

🚚 [OptSeqの実験的解析](#)

📄 [階層的巡回セールスマン問題](#)

📄 [動的確率的ビンパッキング問題](#)

📄 [分割配送問題に対する近似解法](#)

🔍 [Interpretable AI](#)

データサイエンス演習

アナリティクス（データサイエンス）練習問題集

データを入手したとき、最初に行うことはデータの概要を把握することである。これをデータの探索(exploration)

🔗 <https://mikiokubo.github.io/analytics/>

環境整備

【初心者向け】PythonとJupyterの環境構築手順2021年版(Windows編) - Qiita

本記事はWindowsユーザーが対象となります デフォルトの状態から環境構築していきます。コマンドプロンプトをインストールしてJupyter ...

🔗 <https://qiita.com/Nate0928/items/d4b8cc091f7f1510f3c8>

【初心者向け】Pythonとjupyterの環境構築手順2021年版 (Mac編) - Qiita

本記事はMacユーザーが対象となりますが、M1搭載のMacについては未だ未知数なことも多いため対象外です。 Mac上にPythonでのデータ分析環境を整えることを目的とします。 ...

🔗 <https://qiita.com/Nate0928/items/40bc828fa4ef28fa560e>

Streamlit + github

[簡単爆速]Pythonだけでデプロイ。Streamlit1.0の使い方を紹介 - Qiita

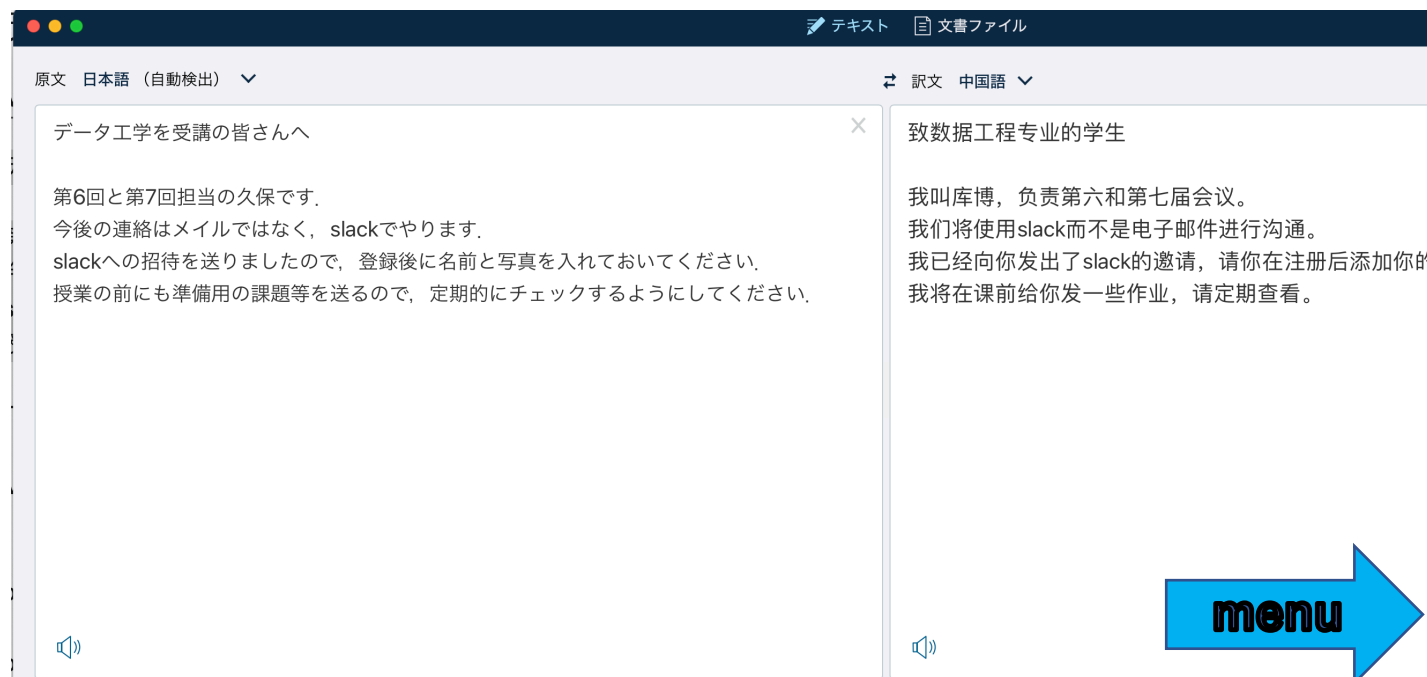
データ分析の結果や研究成果を公開するとき、多くの方に使ってもらえるように、何かしらツール化したほうが良いムワークです。 ...

🔗 <https://qiita.com/Nate0928/items/0b192c903a41d75cb7c6>

menu

DeepL

- <https://www.deepl.com/>
- 深層学習による翻訳
- 読み上げ機能
- 無料アプリあり



- イタリア語
- エストニア語
- オランダ語
- ギリシャ語
- スウェーデン語
- スペイン語
- スロバキア語
- スロベニア語
- チェコ語
- デンマーク語
- ドイツ語
- ハンガリー語
- フィンランド語
- フランス語
- ブルガリア語
- ポーランド語
- ポルトガル語
- ポルトガル語 (ブラジル)
- ラトビア語
- リトアニア語
- ルーマニア語
- ロシア語
- 英語 (アメリカ)
- 英語 (イギリス)
- 中国語 (簡体字)
- 日本語

Slack

- <https://slack.com/>
- 授業やゼミの連絡や質問（チャット）
- リアルタイムに関係者全員に（頻繁に）連絡可能



10月12日 (火)



久保幹雄 14:40

11/10 と 11/17 のデータ工学の講義担当の久保です。

テーマはビッグデータ時代のサプライ・チェーンです。youtubeで予習してもらい、講義ではzoomでdiscuss
ビッグデータやデータサイエンスについて知りたい人は <https://www.youtube.com/watch?v=CH01VxmTeK8>
サプライ・チェーンのケーススタディについては <https://www.youtube.com/watch?v=vqXadn-gt2s>
を暇なときにでもみておいてください。

My name is Kubo, and I will be teaching Data Engineering on 11/10 and 11/17.

The theme is "Supply Chain in the Age of Big Data", and we will have a discussion and exercises using data in the
If you want to know more about Big Data and Data Science, please visit <https://www.youtube.com/watch?v=CH01VxmTeK8>
If you want to know the supply chain case study, please visit <https://www.youtube.com/watch?v=vqXadn-gt2s>

我叫Kubo, 我将在11/10和11/17教授数据工程。

主题是“大数据时代的供应链”，讲座后将通过放大数据进行讨论和练习。

如果你想了解更多关于大数据和数据科学的信息，请访问 <https://www.youtube.com/watch?v=CH01VxmTeK8>

如果你想了解更多关于供应链案例研究，请访问<https://www.youtube.com/watch?v=vqXadn-gt2s>。

在你的业余时间。（編集済み）



SPRINT



Noriko Yoshida

Tokyo university of marine science and technology

SPRINTとは

今までにない革新的な問題発見と解決の方法

SPRINTの始まりと現在



- GVで用いられる
- スタートアップ企業が活用

• Googleでスタート

- GVデザインパートナー
Jake Knappが開発



• 日本で出版

- 2017年4月

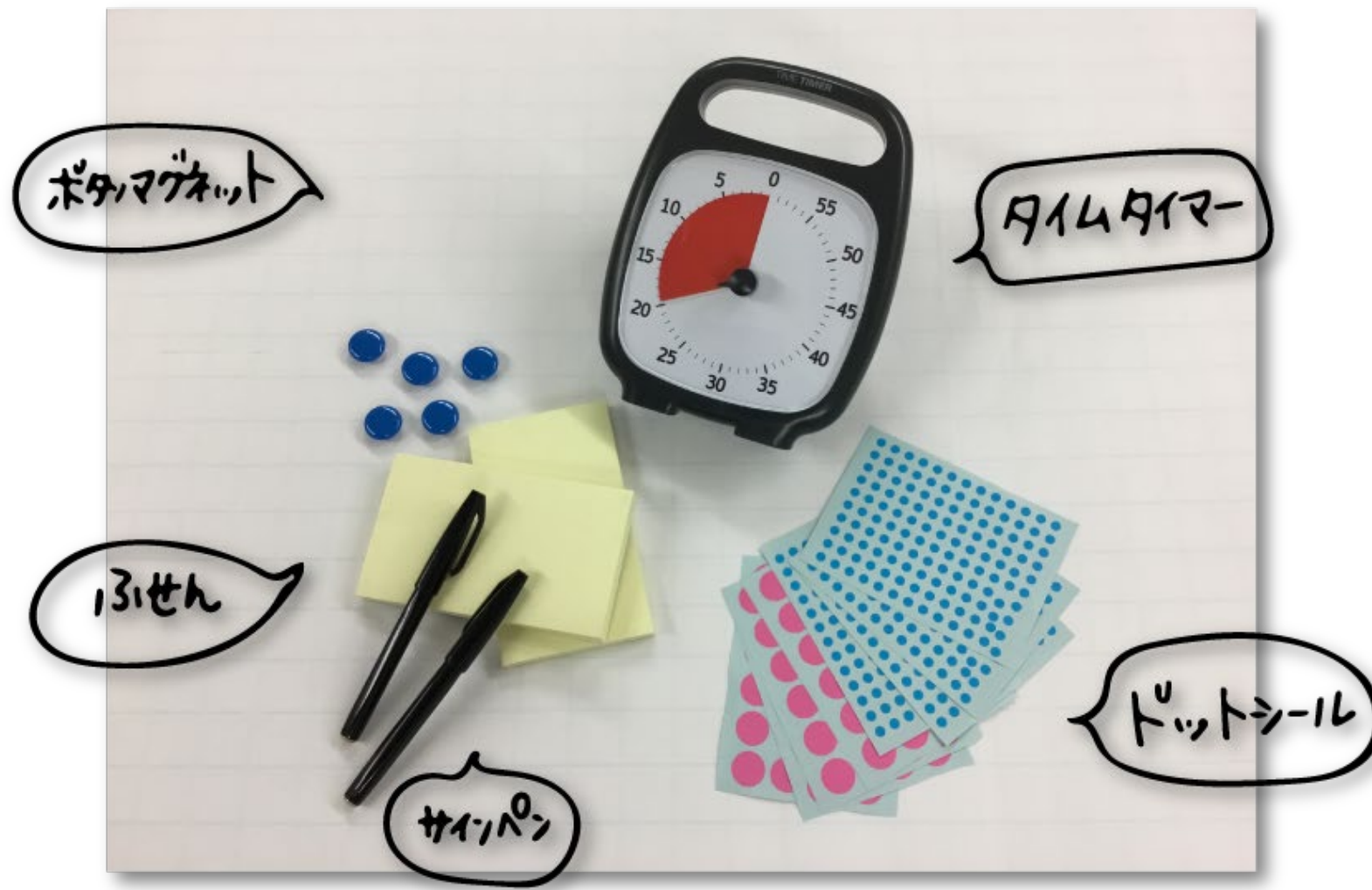


2017年

2012年

2010年

使う道具が決まっている



あとはコーヒー  とヘルシーなスナック  も！

ホワイトボードを2台使用する

スマホ専用の画面イメージ作成
手描きの絵で作成

画面まわりの作成ツール
スケッチと一緒に
UXも担当。動き

キャンセルweb上で可
カレンダーが見やすい
オープンソースで出ている

皆手UIが優れている。
料金設定-自動(変動)
2Dの国自動ほんやく

お年寄り(65歳以上)シェアビジネス
敬老パス・社会貢献
(無料)・地下鉄完配

webクラウド画面イメージが iPhone
作れる。フォト作成・グループ共有

Pratty
InVision
Airbnb
Airbnb
Moqups

経路検索
時速設定(自転車考)
中央分離帯あり
道路別制限速度なし

移動してデータをクラウドへ
世界中、1時間おき
ポラリアによる作成
ダウンロード使用可
経路検索も Open Street Map
可能

グローバル仕様
日本語・電話番号・姓名順
海外旅行者向け?
UIがいろいろ(?)

ユーザー登録なし
起動後すぐ使える
タブが全部おける(動かせる)
お迎えまで
料金検索

ピクセルに
Webデザイン
作業分担
CSSが使

乗車場所を指定
東京

港区

全国マップ(日本専用)

SRM
Open Street Map
UBER
スケ

長期目標
システムを特化する前提

ドライバー
ルート管理者
西配送セマ
クラウド

顧客
注文者

最適化
モデルで解けるか。
ドライバーが納得する最適化
適切な情報が入っているインターフェース

11月より無効が2回と使った

ピックアップを含む
配達経路決定

注文・交通・ドライバー現在地
不在情報

時間までに届く

荷物受取

1日の方
常に情報のアップデート

最適化の
配送情報

荷物ごと
セマ間の
ルート最適化
住地とビジネス街

到着予定時刻
のお知らせ連絡

当日の朝まで
コール便
生花のお届け
は連絡が11月9日

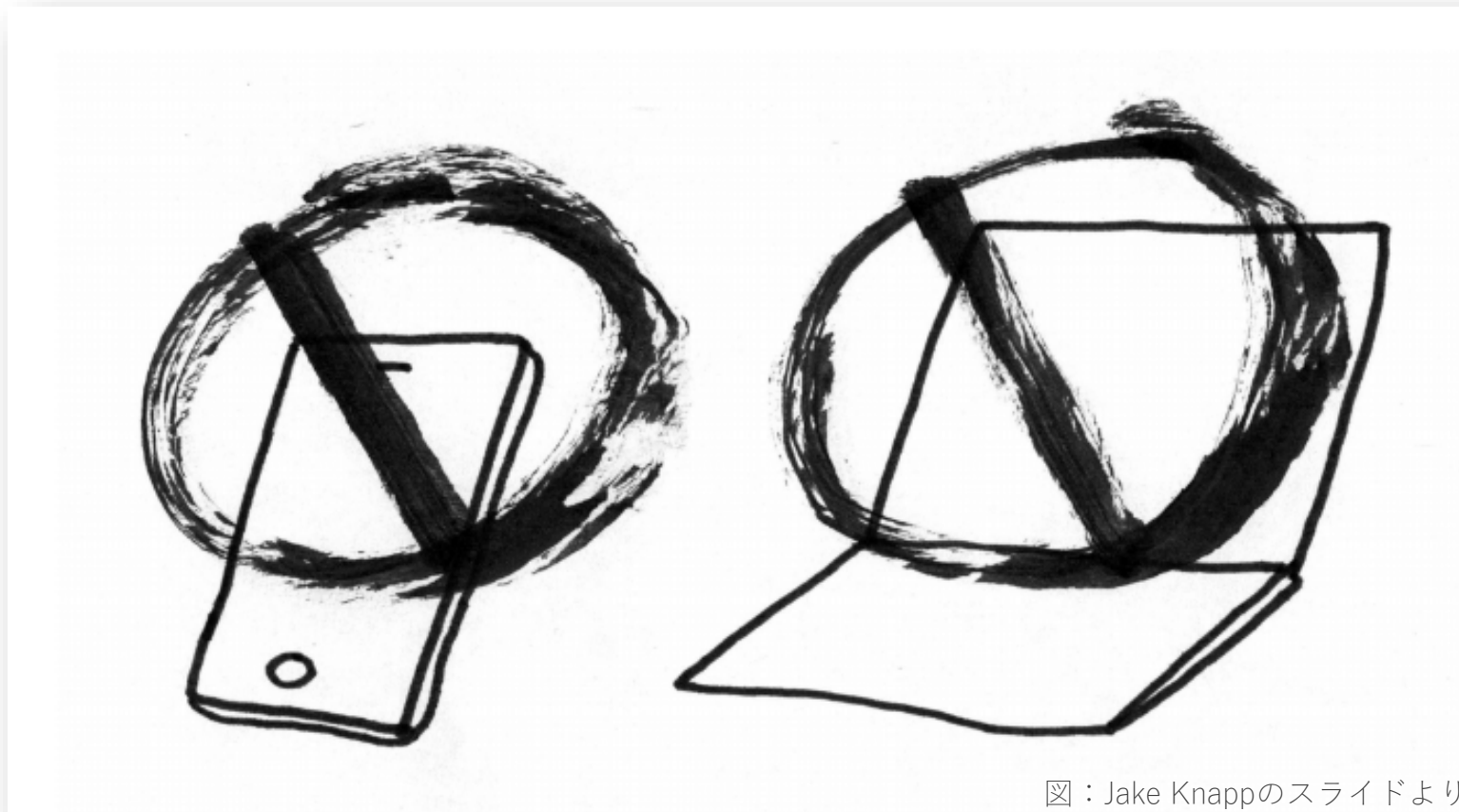
高松の
お知らせ



メンバーの知識と知恵は全部ここに記録し、ホワイトボードが共通の頭脳となる

SPRINT中はデバイス使用禁止

SPRINTの工夫 その⑤



図：Jake Knappのスライドより

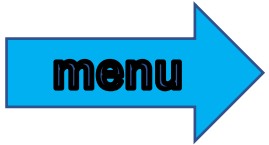


気が散るものは部屋の外で使ってね。休憩時間は自由に使用できます！

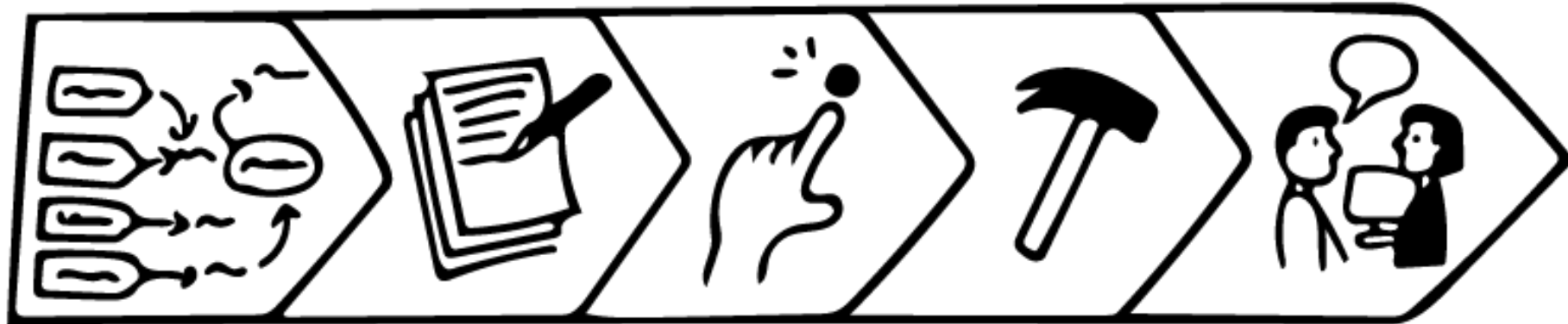
SPRINTのプロセス

1. 問題の神髄を見極める
2. 出来るだけ多くの解決策を考えてみる
3. いくつかのアイデアを選び、研ぎ澄ます
4. 最終的なアイデアに絞り込む
5. プロトタイプの作成
6. 結果を知って、学習する

5日間のプロセスが決まっている



月 火 水 木 金



マップを
つくり、
ターゲットを
決める

ソリューション
を
スケッチする

どれか
一番
よいかを
決める

リアルな
プロタイプ
をつくる

ターゲット
顧客で
テストする

MIROのSPRINTテンプレート

The image displays a Miro workspace for a SPRINT template, organized into several key sections:

- Start here:** Includes "Miro Intro" and "Quick Exercise" cards.
- Overview:** A central hub for "Presenting our team & challenge" with a "Let's get started!" button.
- Expert Tasks & HW Statements:** Contains "Independent Work Area - HWs" and "Categories & HW Voting" sections.
- Map:** A central hub for "Drawing the Map + Placing HWs" with a "Map" section.
- Day 2:** Includes "投票" (Voting) and "Sketches" sections.
- HMW Board:** A section for "Lightning Demo" and "Long Term Goal" with a "HMW Board" section.
- Concept Sketching:** A section for "Concept Sketching" with a "スケッチ" (Sketch) section.
- Sprint Questions:** A section for "Sprint Questions" with a "失敗の原因" (Reasons for failure) section.
- Sketch Example:** A section for "Sketch Example" with a "Catchy title" and "Words matter" section.
- Other sections:** "Solution presentation", "Synchronized voting", "User Test Flow", and "User test Flow Matrix".

The workspace includes a toolbar on the left with various drawing tools and a top navigation bar with "miro", "スプリント", and "Share" options. A vertical sidebar on the left contains icons for navigation and tool selection.