ゼロからはじめるデータサイエンス入門

BPStudy #173

矢吹太朗（@yabuki） & 辻真吾（@tsjshg）

おしながき

- イントロダクション（辻）
- RとPythonの比較から見るデータサイエンス（矢吹）
- データサイエンスをはじめよう！（辻）

【実践Data Scienceシリーズ】

ゼロからはじめる

データサイエンス入門

―― R・Python一挙両得 ――

辻　真吾
矢吹太朗 著

RとPython両方学べる。コスパ最強の一冊!

コードが理解の試金石!

- 「データサイエンスの準備」にページを割いているから、プログラミング経験ゼロで大丈夫!
- 自分に合った言語を見つけたい、言語を乗り換えたいという方にもおすすめ!

講談社

全体の構成

- コンピュータとネットワークと環境構築（1, 2章）
    - シェルの操作やTCP/IPなど
- RとPythonの基本文法（3章）
- 統計入門と前処理（4, 5章）
- 機械学習アルゴリズム（6〜13章）
    - 深層学習と時系列データ解析を含みます

データサイエンスの準備（コンピュータの基礎）

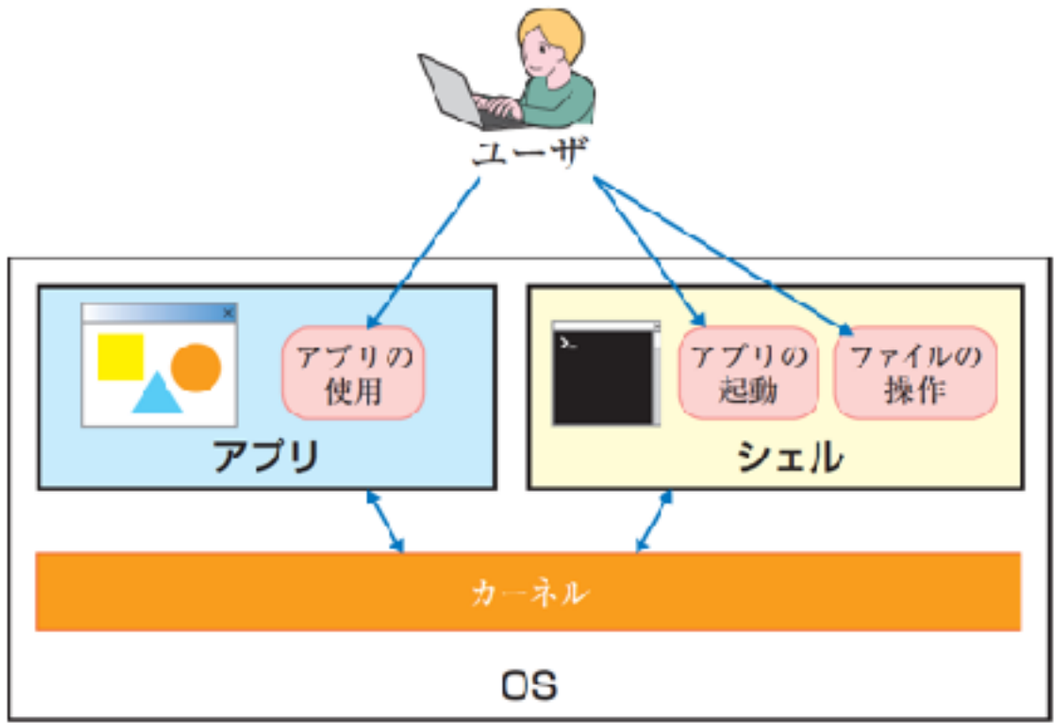


図 1.2　シェルの役割

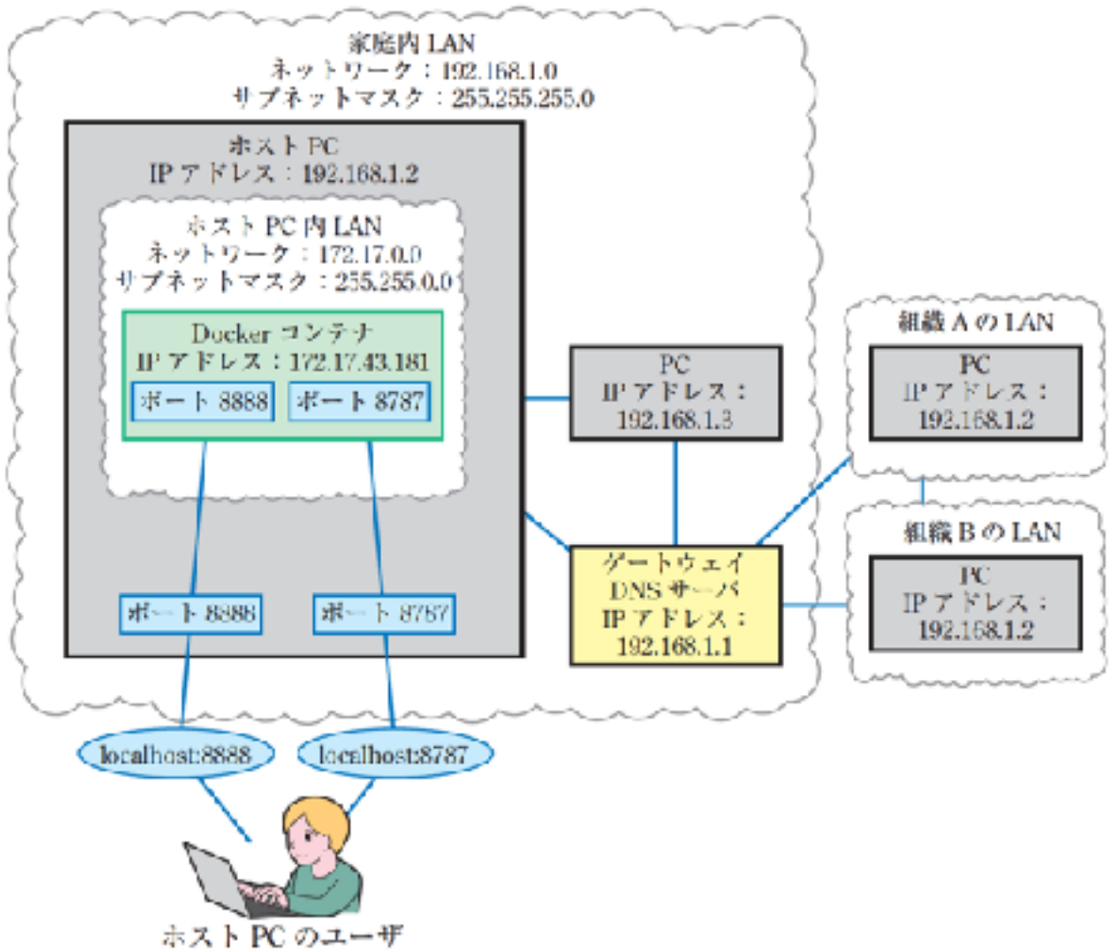


図 1.4　LAN について理解するための素材

1.1 コンピュータの基本操作（P5）

1.2 ネットワークの仕組み（P15）

この本のほんとうにすごいところは・・・

著者の2人は小学校1年生からの友達

辻真吾

担任の
酒井先生

矢吹太朗

1982年4月　東京都足立区立千寿第五小学校入学式の写真

ここで一旦演者交代

お前、誰よ？（自己紹介）

- 大学の研究所に勤めています
    - エネルギーシステムとバイオインフォマティクス
- 2005年ごろJavaからPythonに乗り換えました
    - 2010年に「Pythonスタートブック（初版）」をだしてから技術書を何冊か執筆しています
- 2021年一番の思い出は「7月にRustに入門したこと」
- 月に1回オンラインで「みんなのPython勉強会」を開催しています

心得その1

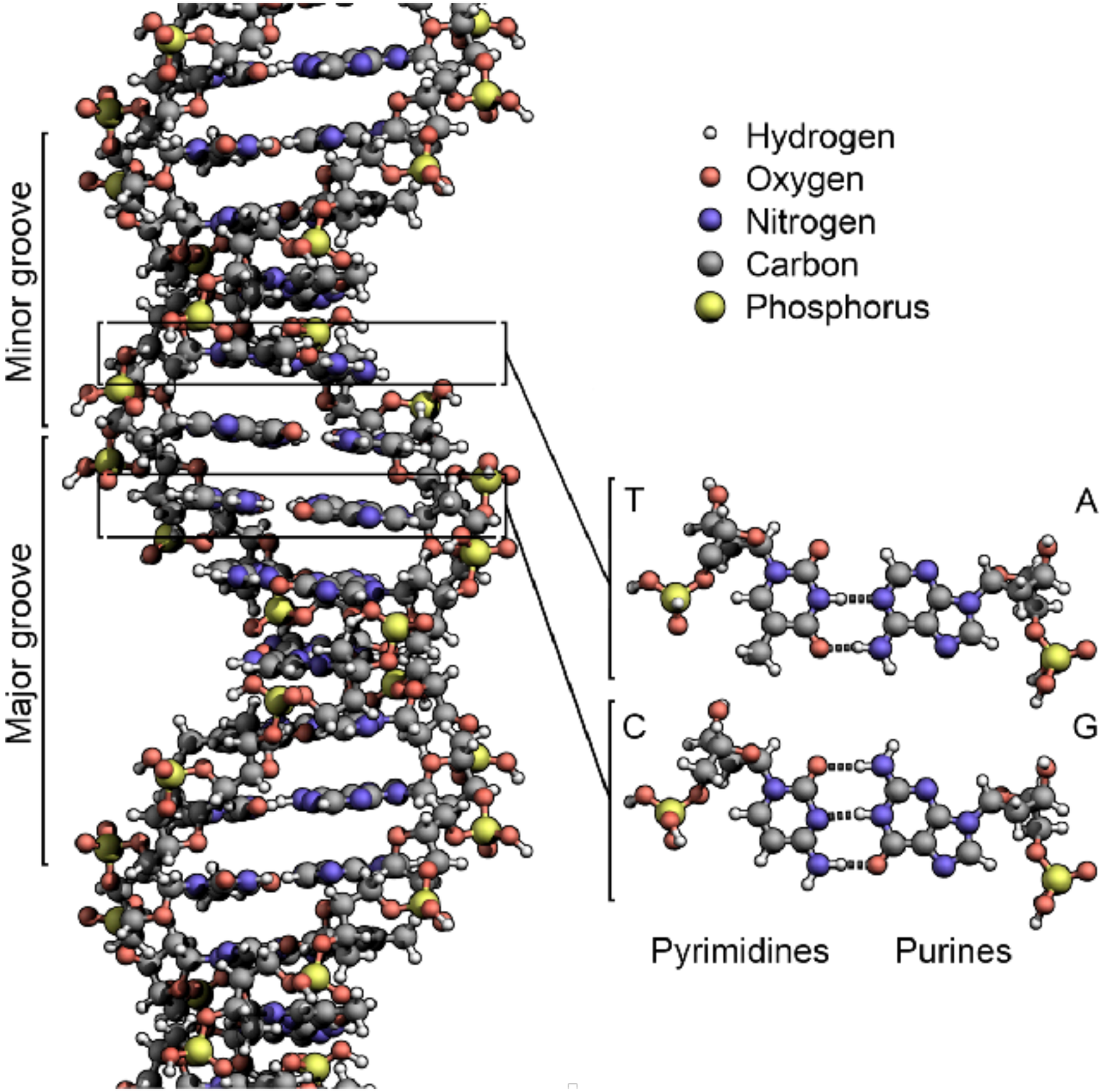
革命的なことが起きているという事実を認識する

そもそも、データサイエンスってなに？

データサイエンスは「データ駆動型サイエンス」に由来

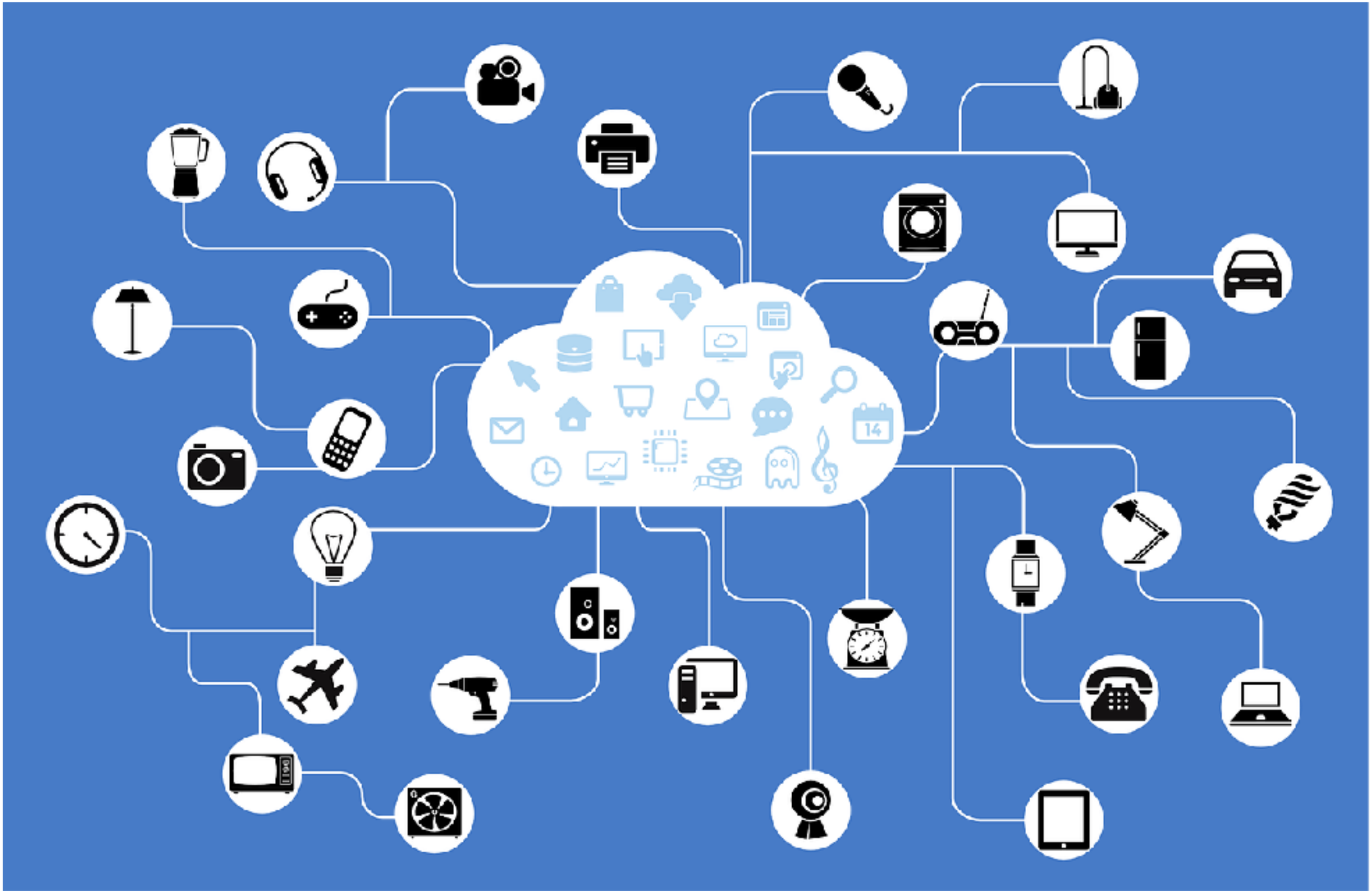
観測データを分析すると動きが全然違う惑星を発見できる

いま何がおきているか？



ヒト1人1つの細胞だけで32億文字

いま何がおきているか？



すべてのモノがインターネットに繋がって情報が蓄積される

なにが問題か？

- データが多すぎる・・・

- 夜空をぼーっと眺めていて惑星の存在に気が付くレベルの量では無い

- Webサイトのログを眺めて潜在顧客を見つけられるか？

- データ駆動型のサイエンスやデータ駆動型のビジネス（意志決定）を実践するにはスキルが必要

  - それがデータサイエンス

機械学習アルゴリズム革命の本質

- 機械学習アルゴリズムはデータサイエンスを実践するための道具の1つ

- プログラムのコードの一部が機械学習モデルに置き換わる

- 画像認識におけるDeep Learningの成功

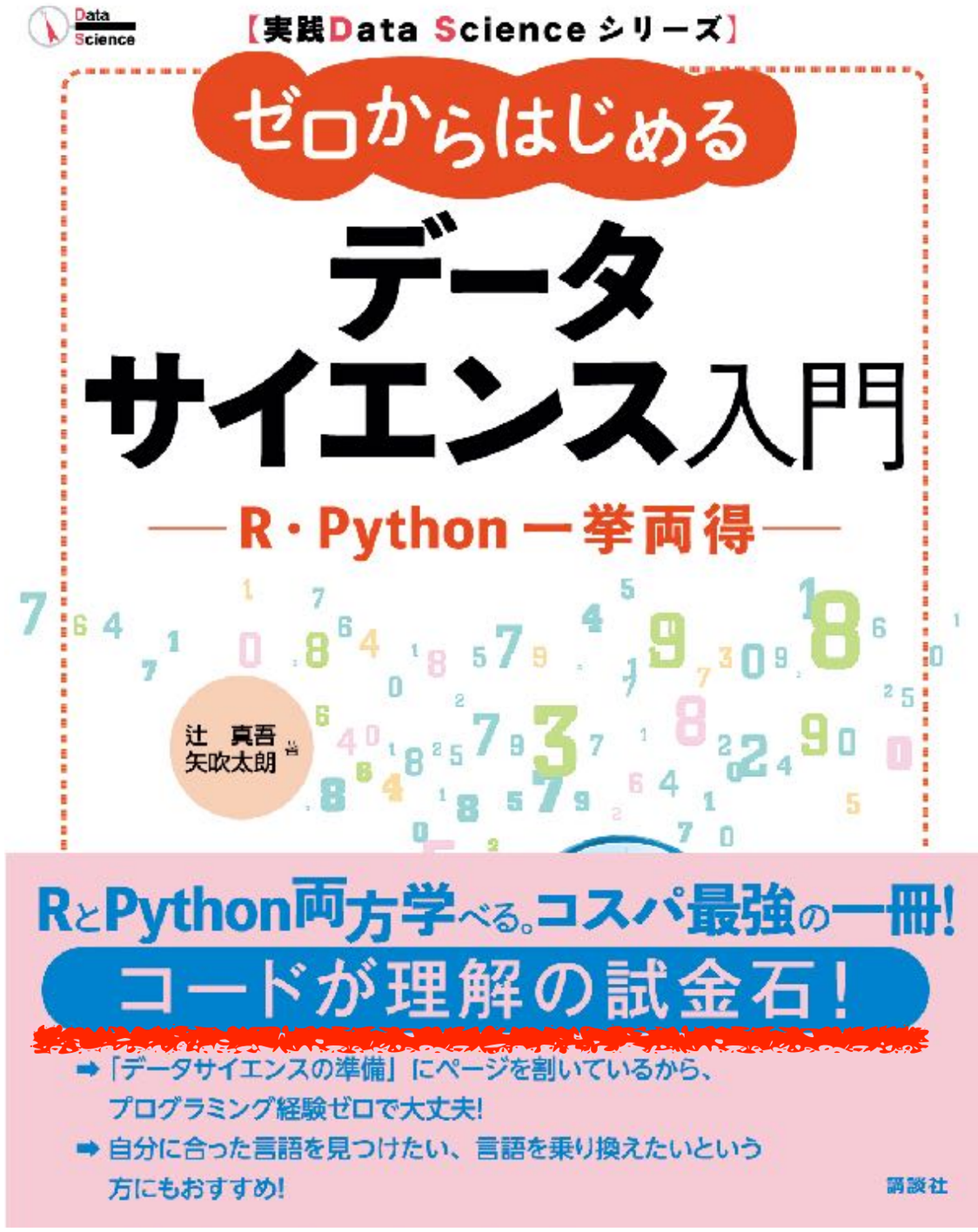
- データがあればコードを書かなくて済む

心得その2

データサイエンスはそんな大したもんじゃない（と思い込む）

実践あるのみ

- Webアプリ開発をしているプログラマのうち何人がHTTPサーバを実装したことがあるだろうか？
- 誰だってはじめは何も知らない
- まずコードを書いて動かしてみよう！
- 手法の種類や調整できるパラメータなどを調査することで知識を増やす



すこし慣れてきたら

- 深い知識を得るために
  - 全自動で山のような量の結果を返すツール（pandas-profilingやPyCaret）を実行してみて結果を解釈できるように頑張るというのも1つの方法

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
lightgbm	Light Gradient Boosting Machine	0.9743	0.9933	0.9899	0.9797	0.9848	0.9032	0.9040	0.0490
gbc	Gradient Boosting Classifier	0.9668	0.9890	0.9901	0.9710	0.9804	0.8723	0.8745	0.2400
ada	Ada Boost Classifier	0.9585	0.9840	0.9823	0.9686	0.9754	0.8425	0.8436	0.0810
rf	Random Forest Classifier	0.9515	0.9843	0.9904	0.9535	0.9716	0.8054	0.8126	0.1510
dt	Decision Tree Classifier	0.9360	0.8838	0.9609	0.9626	0.9618	0.7647	0.7651	0.0240
et	Extra Trees Classifier	0.9196	0.9626	0.9909	0.9194	0.9538	0.6459	0.6744	0.1410
lda	Linear Discriminant Analysis	0.9110	0.9267	0.9657	0.9308	0.9479	0.6436	0.6493	0.0280
ridge	Ridge Classifier	0.9045	0.0000	0.9842	0.9094	0.9453	0.5742	0.6038	0.0170

PyCaret（AutoMLの一種）の出力例

もし余裕があれば

- “何かを理解する最良の方法はそれをプログラムに組むこと、そして、それがコンピュータ上で動くかどうかを確かめることなのです”（グレゴリー・J・チャイティン著「知の限界」黒川利明訳, 2001, P83）

- 何か1つの機械学習アルゴリズムを実装してみることはかなりおすすめ

  - K-meansクラスタリング、決定木、Random Forests

心得その3

結果が同じにならないことに寛容になる

```
# data: 説明変数のDataFrame
# y: 目的変数
from sklearn.model_selection import train_test_split

# データを訓練用とテスト用に分割
X_train, X_test, y_train, y_test = train_test_split(d_data, y, test_size=0.3)
```

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

# Random Forestsのインスタンスを用意
rfc = RandomForestClassifier(max_leaf_nodes=20)
```

```
# 訓練データで学習済みモデルを作る
rfc.fit(X_train, y_train)
# テストデータの説明変数で予測する
pred = rfc.predict(X_test)
# 性能を評価する
accuracy_score(y_test, pred)
```

```
0.9111549851924975
```

```
# まったく同じことをする
rfc.fit(X_train, y_train)
pred = rfc.predict(X_test)
accuracy_score(y_test, pred)
```

```
0.907864429088516
```

同じデータに同じアルゴリズムを適用しているのに結果が違う

原因は大きく分けて2つ

- アルゴリズムがランダムな挙動を含む
- 数値計算の誤差

疑似乱数列を同一にする

```python
# random_stateを固定する
rfc = RandomForestClassifier(max_leaf_nodes=20, random_state=0)
```

```python
# 訓練データで学習済みモデルを作る
rfc.fit(X_train, y_train)
# テストデータの説明変数で予測する
pred = rfc.predict(X_test)
# 性能を評価する
accuracy_score(y_test, pred)
```

```
0.9095097071405067
```

```python
# まったく同じことをする
rfc.fit(X_train, y_train)
pred = rfc.predict(X_test)
accuracy_score(y_test, pred)
```

```
0.9095097071405067
```

同じ結果になる

数値計算の誤差

- 小数を誤差無く表現することはできない
  - 10進数で1/3は0.333333・・・だけど3進数なら0.1
  - 10進数で1/10は0.1だけど

```
from decimal import Decimal

Decimal(0.1)
```

```
Decimal('0.1000000000000000055511151231257827021181583404541015625')
```

コンピュータでデータ（数値）を扱うときは
常に頭の片隅に置いておきたい

心得おまけ

数学は、数式をみて概念が理解できることが重要

データサイエンスと数学

- 数学ができるとは？
    - 計算が得意とか、複雑な積分を筆算で解けるなどの能力はひとまずいらない
- 機械学習アルゴリズムを利用する側は、数式を見て概念がわかることが重要
    - 数式は単なる表現方法
    - 微分は引き算、積分は足し算、ベクトル・行列・テンソルは数の規則正しい並びと一括四則演算

データサイエンス入門の心得

1. 革命的なことが起きているという事実を認識する

2. 大したもんじゃないと思い込む

3. 結果が同じにならないことに寛容になる

4. （おまけ）数式は単なる表現方法、身構えず数式から概念を読み取れるようになろう

気軽にデータサイエンスに入門してみよう！