データサイエンス・シリーズ

Pythonが動く Google Colabで AI自習ドリル

牧野 浩二，足立 悠 共著

独学できる24の主要アルゴリズム

このPDFは，CQ出版社発売の書籍
「MicroPythonプログラミング・ガイドブック」の一部見本です．
内容・購入方法については，下記のWebサイトをご覧ください．
内容：https://shop.cqpub.co.jp/hanbai/books/45/45181.html

見本

CQ出版社

本書の読み方…AIアルゴリズムを使いこなすために

牧野 浩二

本書を読むとできるようになること

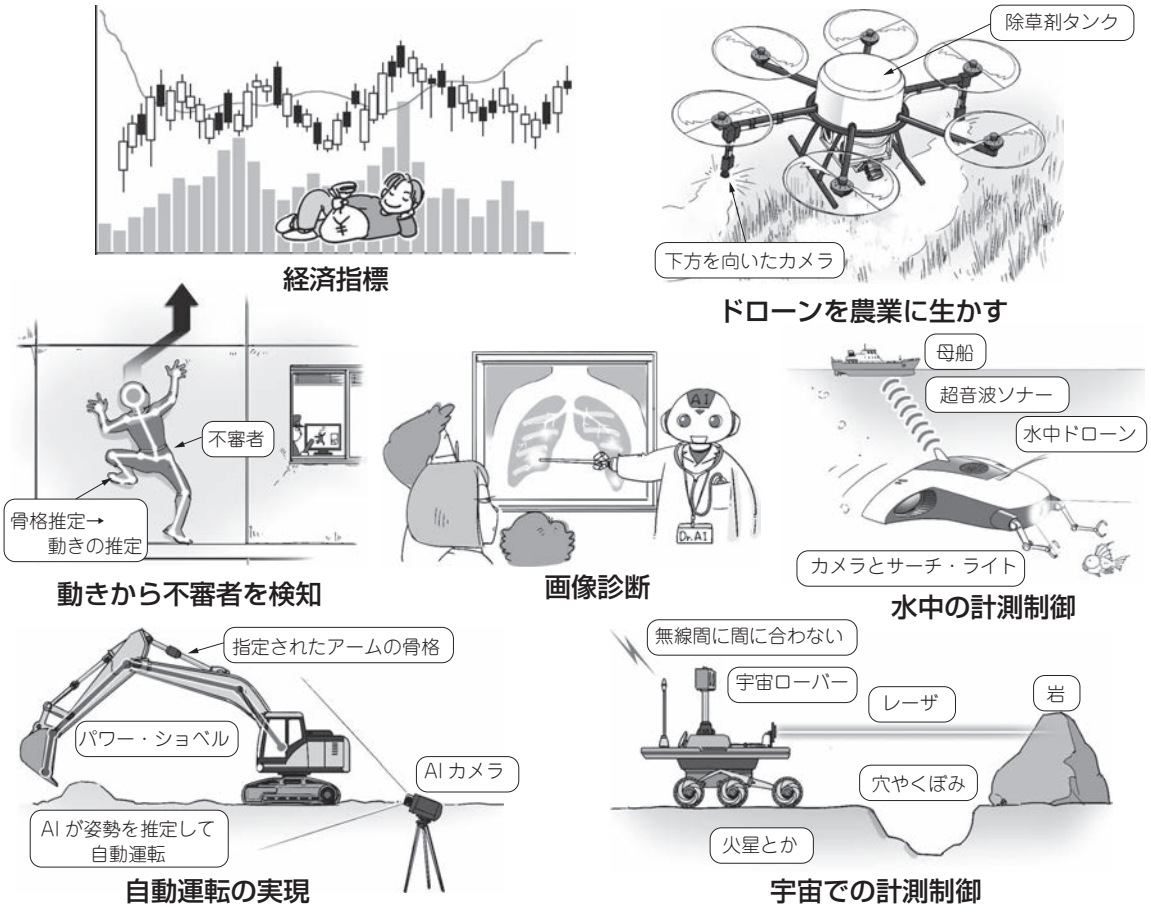


図1　AIはいろいろな可能性を秘めている

AIの未来と現在

AI(Artificial Intelligence，人工知能)と言うと，ロボットが人間と話したり，治療薬を自動的に見つけたりなど，雲の上の技術に感じるかもしれません(**図1**)．

AIと言えば，現在はディープ・ラーニング(深層学習)が一番有望な技術として考えられているようですが，機械学習やデータ分析などと呼ばれる手法も，AI技術の一部です．この機械学習やデータ分析の方法はとてもたくさんあり，社会人ならば会社の業務，学生ならば実験データの解析に使えるものばかりです．そして，これらはデータ・サイエンスやビッグデータ，IoTの技術とも密接に関わっています．

本書の目指すAIの学び

● 完全理解よりもとりあえず正しいイメージをもって使えることを目指す

本書は，AIの最先端を作る研究者向けではなく，AIを使いこなしたいと考えている人向けです．AIは高度な数学を使うことが多く，完全に理解するまでには相当な学習量が必要です．AIを学びたい，使えるようになりたいと思っても，以下のような点がハードルとなることが多いようです．

- AIを使ってみたいがどこから手を付けてよいか分からない
- AIの数学でいきづまった
- 教科書のアルゴリズムは動かせたが，自分のデータにはどう適用すればよいのか分からない

そこで本書では，読者の方がAIを使いこなせるようになるために，以下の点に注力して解説しました．

本書のポイント1…これ1冊でAIアルゴリズム全体を概観できる

AIにはいろいろな技術があります．よく知られているところとしては，機械学習，ディープ・ラーニングなどです．人工知能学会の資料「AIマップ」[(1)]によると，AIにはさまざまな技術，さまざまなその分類があります．**図2**に，この「AIマップ」22ページに掲載されている図を基にして本書で扱う技術をまとめました．本書では以下の4つから，基本としておさえておきたいアルゴリズムをいくつか抽出して解説します．

- 機械学習
- ディープ・ラーニング
- 強化学習
- 数理モデル（探索・論理・推論アルゴリズム，スケジューリング，群知能，ファジィ理論）注1

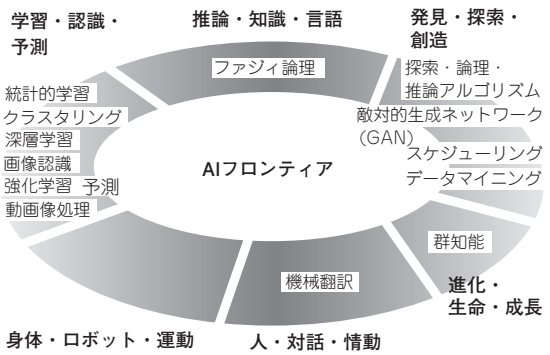


図2[(1)] AIにはさまざまな技術が関連している
人工知能学会のAIマップ22ページの図を筆者が改変

本書のポイント2…プログラムを動かしながら学習できる

AIを使いこなすには，以下の観点がが重要です．

①その仕組みを大まかに理解すること
②実際にやってみること
③それを応用してみること

本書では，上記の順番で説明を行っていきます．仕組みの説明では，難しい数学を使わずに書いてありますので，読み物のように読み進めていけるようになっています．仕組みを全く知らない状態で動かすと効果的に使いこなすことができなかったり，間違った解釈をしてしまうことになります．効果的にかつしっかり使いこなすには，仕組みを大まかに理解しておくことがとても重要です．

実際にやってみる部分はサンプル・プログラムがダウンロードできるだけでなく，手順と内容を説明することで，どのような仕組みかを動かしながらつかめるような構成としました．

本に書いてあるデータではできるけど，実際に自分のデータで使うときに困ることはよくあります．そこで，改造してご自身のデータで試して応用できるような部分まで説明するようにしました．それぞれの章は独立していますので興味のある所から読み進めてもらえるようになっています．

本書のポイント3…実行環境は面倒な初期設定が不要なGoogle Colaboratory

本来，ディープ・ラーニングのプログラムを動かすためには高性能なコンピュータ（計算機）が必要となります．本書ではGoogle Colaboratoryを利用します（**図3**）．これを使うとウェブ・ブラウザ上でプログラムを書いて，クラウドにあるかなり高速なコンピュータで実行できます．制限はありますが，この本で扱うプログラムは無料の範囲で動かすことができます．

ただし，Google Colaboratoryには苦手なことがあったり，インターネットにつないでないと動かせないといった問題もあります．そこで，一部はWindowsにAnacondaという統合環境をインストールした環境で動かすことも想定しています．

本書のポイント4…Pythonで解説

PythonにはAIを実装する上で便利なライブラリやフレームワークが豊富にあります．また文法が比較的簡単で処理内容が理解しやすいことから高い人気があります．このためAIの学習にはPythonがよく使われています．

注1：数理モデル：数学的手法により答えを出す方法．

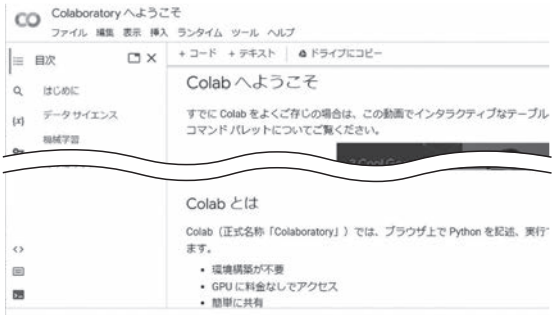


図3　Google Colaboratoryの初期画面

記事中のプログラムを試す方法

筆者提供プログラムの入手先

本書では筆者作成のプログラムを実際に動かして学習していきます．次の本書サポート・ページからダウンロードできます．

```
https://interface.cqpub.co.jp/bookai2024
```

開発環境…基本は Google Colaboratory

開発環境は主にGoogle Colaboratoryを使います．ネットワークに接続済みのPC以外に特別なものは必要ありません．

まずはGoogleのChromeブラウザを開き，Googleアカウント(Gmailアドレス)でログインします．検索窓にGoogle Colaboratoryと入力し，Google Colaboratoryへのリンクを開けば準備完了です．

図4　Google Colaboratoryと筆者提供プログラムがあれば，AIが手軽に試せる！

筆者提供プログラムをGoogle Colaboratoryで実行する手順

①GoogleのChromeブラウザでGoogle Colaboratoryのサイトにアクセスします．

②表示されたウィンドウ左の「アップロード」をクリックします(**図5**)．

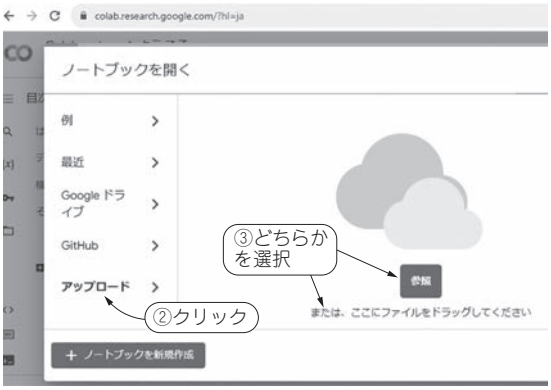


図5　筆者提供プログラムのアップロード画面

図6　筆者提供プログラムをアップロードした画面

第1部

第2章

たくさんの項目からなるデータを人間に分かりやすい形で表示してくれる「主成分分析」

牧野 浩二，足立 悠

主成分分析とは，たくさんの項目からなるデータ（アンケート・データや特徴データ）を人間に分かりやすい形で表示するための方法です．この章では簡単な例題を用いて説明し，イメージをつかむところから始めます．

1　できること

● 例1…選手や生徒の特性を抽出する

▶スポーツ選手

サッカー選手のデータとして，身長，体重の他にも持久走や短距離のタイム，パスやシュートの成功率などが蓄積されていたとします．これらを分類して特性を調べることができます．

▶学生の成績

成績は5教科で評価させることが多くありますし，場合によっては音楽や体育なども含めて評価することもあります．全部の評価を見ても，どの学生がどのような資質があるのか分かりにくいです．簡単な指標でみることで個々の学生の特徴をつかむことができます．

● 例2…ブランドや商品がどう思われているかを知る

▶企業ブランド

例えば腕時計を例に挙げると，セイコー，シチズン，カシオ，ロレックス，タグホイヤー，ノモス，スント，アップルなど多数のメーカがあります．それぞれに使いやすさ，機能，価格，大きさ，重さ，歴史などのアンケートを取って，見やすい形にまとめると，ブランドの傾向がつかめます．

▶ドリンクの味わい

飲み心地や味などについて調べて分類することもできます．その結果から新しいジャンルの飲み物を検討することもできます．本章でも紹介します．

2　イメージをつかむ

例1：ドリンクの味の特徴をつかみたい

主成分分析は複雑なデータを簡単に表現して傾向をつかむための手法です．例えば，**表2-1**に示す飲み物に関するデータ[注2-1]は，主成分分析を使うと**図2-1**のように散布図で「うまく」表すことができます．

この図を見ると，

- コーヒは苦くてリラックス効果がある
- エナジ・ドリンクは甘くて元気になる

といったことを推察できます．つまり，**表2-1**では飲み物の特徴として6個の指標がありましたが，**図2-1**のように2次元の図で傾向を表せています．なお，**図2-1**のグラフ右側/上側の目盛りの数値はデータの

表2-1　飲み物のデータ（drink_data.txt）

飲み物/特徴	甘味	苦味	栄養	リラックス	元気	コク
お茶	1	4	1	5	2	2
コーヒ	1	5	1	5	2	3
コーラ	5	1	3	4	4	2
オレンジ・ジュース	4	3	4	3	4	3
リンゴ・ジュース	4	1	4	3	3	4
水	2	2	1	1	2	1
牛乳	2	1	5	3	4	5
エナジ・ドリンク	4	3	4	1	5	4
スポーツ飲料	5	2	4	2	5	3

注2-1：サンプルとして筆者が作成したものです．

平均が0になるようにし，それを正規化したものが左側/下側の目盛りの数値です．Pythonで作りました．作り方は後で説明します．

例2：リンゴの色と甘さの特徴をつかみたい

紹介するアルゴリズム「主成分分析」のキー・ポイントは，「横軸と縦軸の項目をうまく選ぶ」ところにあります．まずは，項目を選ぶとはどのようなものか，イメージをつかみましょう．話を簡単にするために，**表2-2**に示す5個のリンゴの色と甘さをプロットしたデータを使います．

● ステップ1…データの次元数を減らす

2次元のデータを1次元に直す方法から紹介します．横軸を色，縦軸を甘さにしてプロットしたのが**図2-2**です．主成分分析のキー・ポイントは少ない次元数で表すことです．**図2-2**は2次元ですが，1次元で表すことを考えてみましょう．この考え方がたくさんのデータを2次元の散布図に表すときの基礎となります．

まず，**図2-3**(a)のように横軸だけで表す方法や，**図2-3**(b)のように縦軸だけで表す方法が考えられます．これはそれぞれ，色データと甘さデータだけで表現していることになります．

では，例えば**図2-4**のように斜めに引いた線に，各点から垂線を下ろして，その位置のデータを1次元のデータとしてみましょう．今度は色と甘さのデータが1対1で混ざったデータとなりました．

次に，もっと良い線を引いたものが**図2-5**です．そして，これまでの1次元データを並べたものが**図2-6**です．一番下の○印は△印とほぼ同じように見えますが，主成分分析では，データの分散(ばらつき)の大きさを評価します．従って，○印の方が△印よりも良い1次元データということになります．

ここで分散とは値のばらつきを計算して数値で表したものとなります．主成分分析はこのばらつきが最も大きくなる軸を探すことになります．

● ステップ2…次の軸を決める

さらに，主成分分析はデータの次元数(この場合は2次元)まで軸を順に探していくことになります．次の軸は最初の軸に垂直な軸ですので，2次元の場合は**図2-7**のように傾きが決まり，次の軸が自動的に決まります．

このようにして最終的に2次元の散布図に直したのが**図2-8**です．横軸が第1主成分，縦軸が第2主成分を表しています．よく見ると**図2-8**は**図2-7**を回転させた形になっています．ここまでは2次元の話ですが，3次元の場合は**図2-9**に示す手順で軸を引きます．

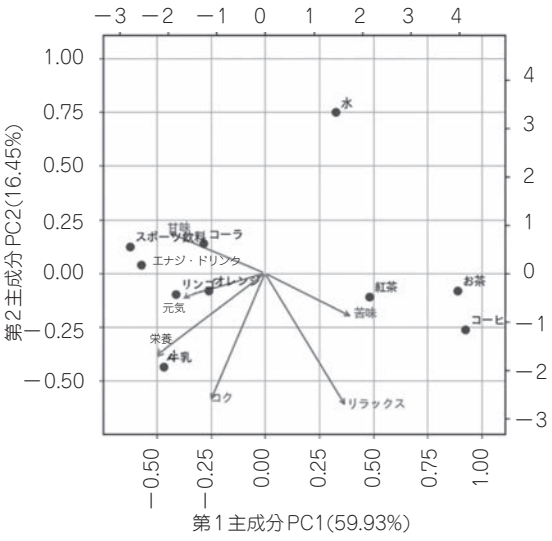


図2-1　主成分分析を使えば多くのデータから傾向が分かる

表2-2　リンゴ5個のデータ(apple_data.txt)

リンゴの種類／特徴	色	甘さ
A	2	2
B	4	3
C	3	4
D	3	2
E	4	5

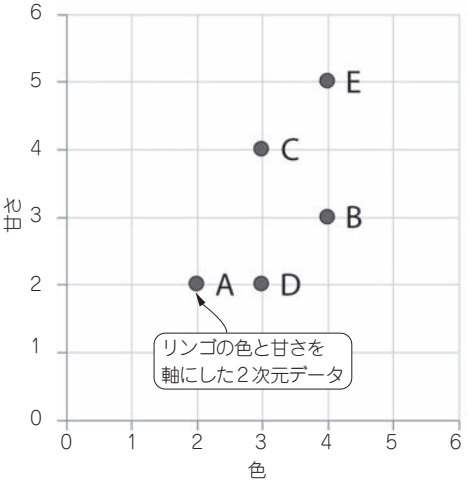
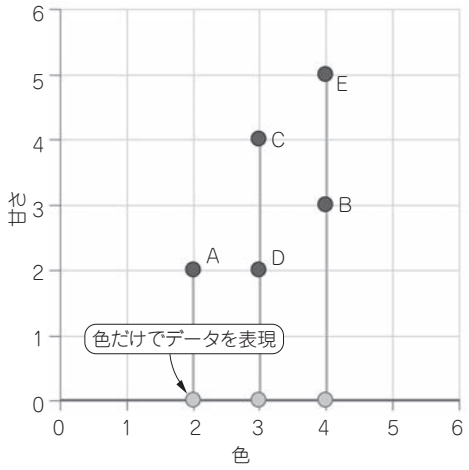
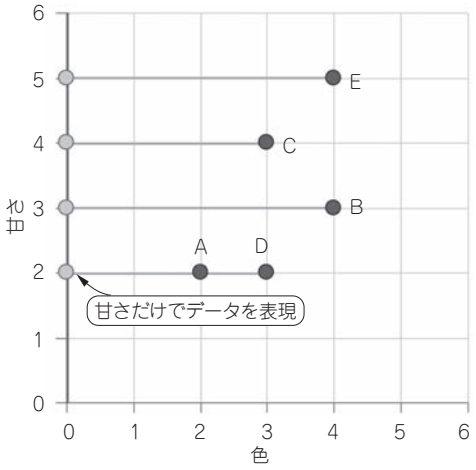


図2-2　この2次元データを1次元にする



（a） 横軸だけで表す



（b） 縦軸だけで表す

図2-3　1次元データにする方法①…どの軸でデータを表現するか決める

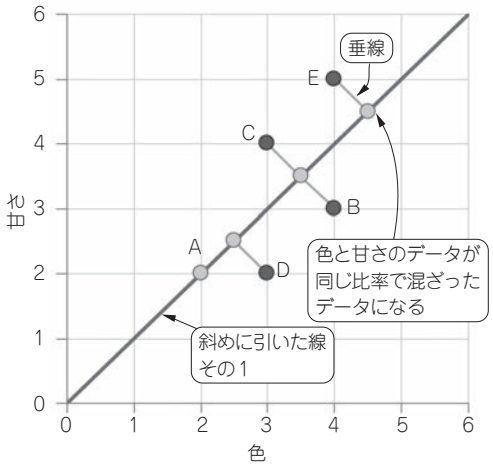


図2-4　1次元データにする方法②…斜めに引いた線に対して
データから垂線を下ろす（斜めに引いた線その1とする）

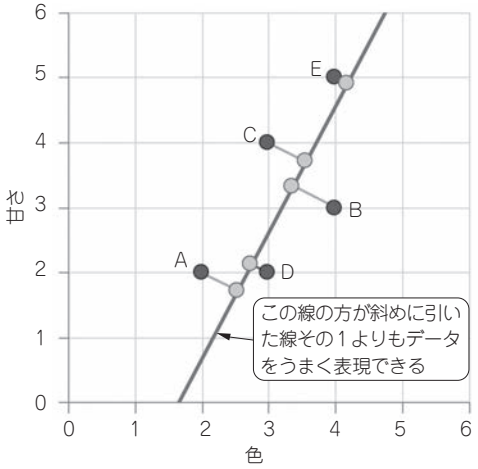


図2-5　斜めに引いた線その2

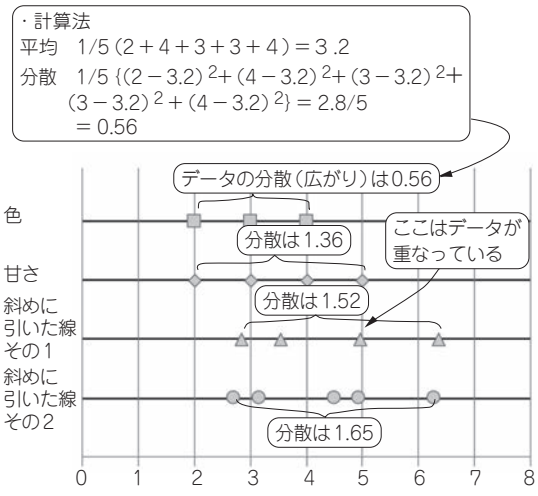


図2-6　1次元データの比較…主成分分析は分散が最も大きい軸のものが選ばれる

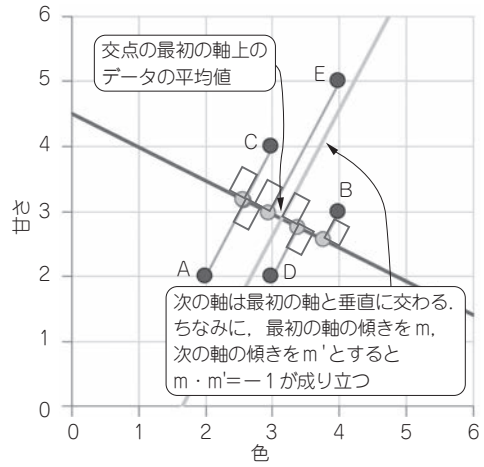


図2-7　最初の軸と次の軸は垂直に交わる

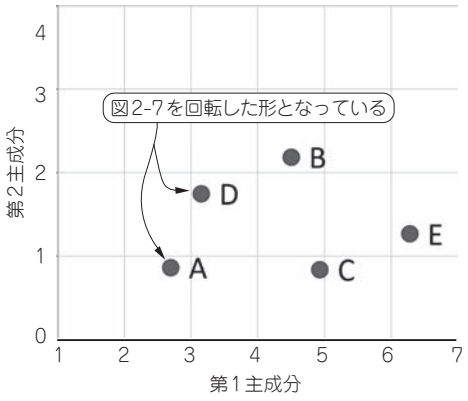


図2-8　最終的に得られた2次元データ

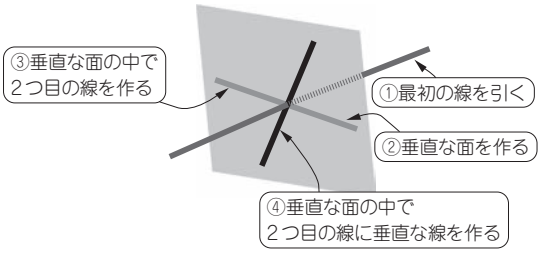


図2-9　第1～第3主成分の線を引くための手順

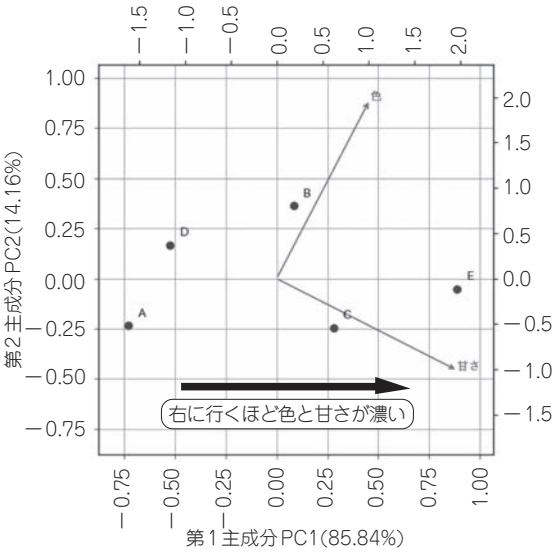


図2-10　主成分分析の結果をPythonでプロット

● ステップ3…Pythonでプロットする

図2-7で2つの軸が決まりました．**図2-10**はそれをPythonでプロットし直した図で，矢印も描かれています．この矢印はもともとの色と甘さの軸となっています．この図から右に行くほど色と甘さが濃くなることが分かります．

3　プログラムを動かしてみよう

リンゴの色と甘さを分析

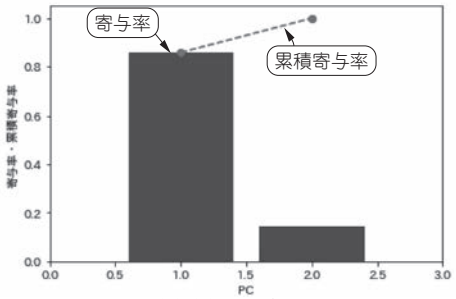
● 主成分分析の実行

表2-2に示したリンゴのデータ（apple_data.txt）を用いて，主成分分析を行います．**リスト3-1**のプログラムを実行すると，**図3-1**（a）～（c）が表示されます．

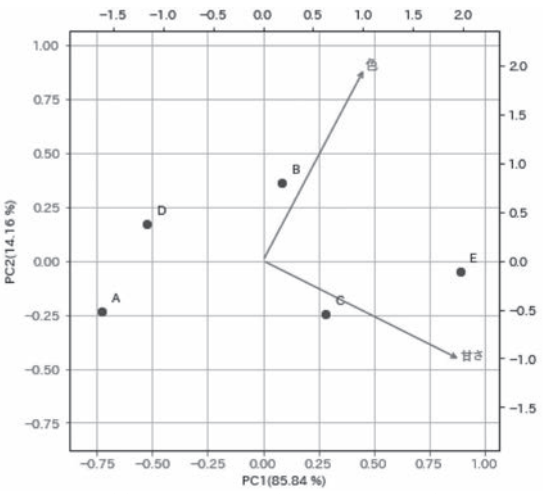
図3-1（c）の軸ラベルPC1やPC2の後の括弧内の数値が，**図3-1**（a）に示す寄与率となっています．**図3-1**（b）の横軸は主成分，縦軸は寄与率および累積寄与率です．寄与率と累積寄与率はこの後で説明します．apple_data.txtの値を変えるとまた違ったグラフが表示されます．

	PC1	PC2
標準偏差	1.435351	0.582896
寄与率	0.858430	0.141570
累積寄与率	0.858430	1.000000

（a）標準偏差や寄与率



（b）（a）を棒グラフで表した



（c）主成分分析結果（**図2-10**と同じもの）

図3-1 リンゴのデータの分析結果

リスト3-1　リンゴ・データの寄与率と主成分分析結果を表示するプログラム（pca_apple.py）

```
from sklearn.decomposition import PCA
                    #①主成分分析を行うためのライブラリ
import pandas as pd
from cq_modules import cq_pca
                    #②主成分分析結果を表示するためのライブラリ
                    #③データの読み込み
df = pd.read_csv("apple_data.txt", sep="\t",
                                        index_col=0)
                    #④主成分分析
pca = PCA()         #主成分分析の準備
pca.fit(df)         #主成分分析の実行
                    #⑤結果の表示
cq_pca.summay(pca)
cq_pca.plot(pca)
cq_pca.biplot(pca, df, 1, 2, scale=0,
                                    pc_biplot=True)
```

表3-1　追加する3つのリンゴ・データ（apple_test_data.txt）

新しいデータ／特徴	色	甘さ
a	1	3
b	4	2
c	5	5

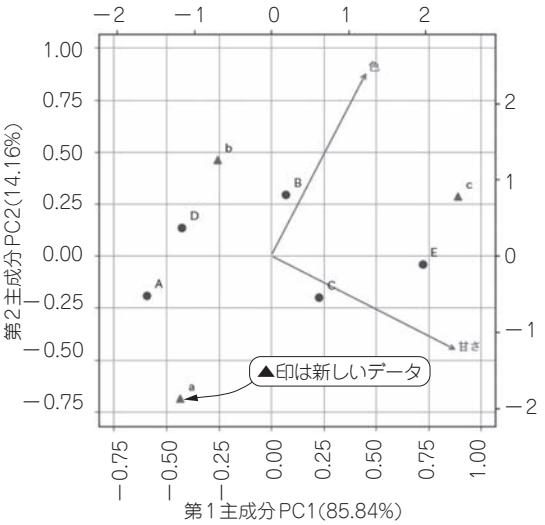


図3-2　新しいデータを追加したプロットもできる

リスト3-2　リスト3-1に新しいリンゴのデータを追加して主成分分析する（pca_apple_test.py）

```
                    #⑥テストデータの読み込み
df_test = pd.read_csv("apple_test_data.txt",
                              sep="\t", index_col=0)
cq_pca.biplot(pca, df, 1, 2, new_data = df_test,
                             scale=0, pc_biplot=True)
```

● 新しいデータを追加する

次に，新しいデータ（**表3-1**）を追加してどの部分にプロットされるかを調べてみましょう．**リスト3-1**に**リスト3-2**のプログラムを追加し実行すると**図3-2**のように表示されます．

飲み物データの分析

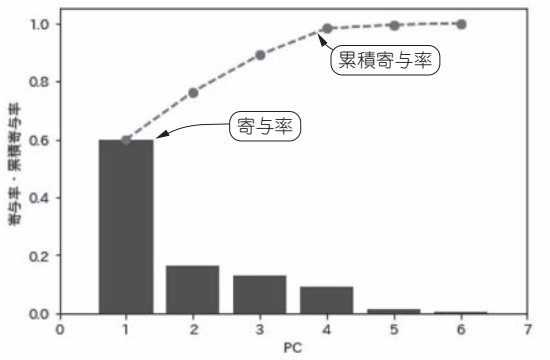
筆者が作成した飲み物データ**表2-1**についても分析してみましょう．**リスト3-3**のコマンドを実行すると，**図3-3**（**a**）～（**c**）が表示されます．

新しいデータとして，**表2-1**に対して**表3-2**に示す3つの飲み物を追加することにします．プログラムとしては**リスト3-3**に**リスト3-4**を追加します．すると，**図3-3**が表示された後に**図3-4**が表示されます．このグラフを詳しく見てみましょう．

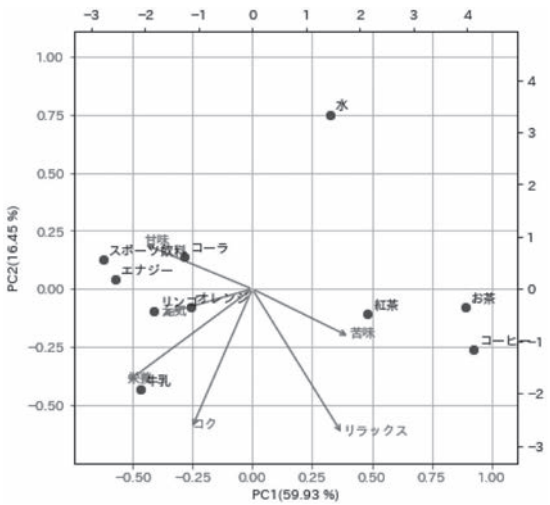
麦茶はお茶やコーヒなどと水との中間の特徴があることが見て取れます．ブドウ・ジュースはリンゴ・ジュースやオレンジ・ジュースと同じような特徴があり，イチゴ牛乳は牛乳と似たような特徴がありますが，ちょっと甘さがあります．

	PC1	PC2	PC3	PC4	PC5	PC6
標準偏差	2.670511	1.399017	1.231613	1.042458	0.379826	0.251524
寄与率	0.599297	0.164475	0.127468	0.091321	0.012123	0.005316
累積寄与率	0.599297	0.763771	0.891240	0.982560	0.994684	1.000000

（**a**）標準偏差や寄与率



（**b**）（**a**）を棒グラフで表した



（**c**）主成分分析結果（**図2-1**と同じもの）

図3-3 飲み物データの分析結果

表3-2　追加する3つの飲み物のデータ（drink_test_data.txt）

飲み物／特徴	甘味	苦味	栄養	リラックス	元気	コク
麦茶	1	3	2	2	1	2
ブドウ・ジュース	4	1	3	4	2	4
イチゴ牛乳	5	1	4	4	4	5

リスト3-3　飲み物データの寄与率と主成分分析結果を表示するプログラム（pca_drink.py）

```
from sklearn.decomposition import PCA
                    #①主成分分析を行うためのライブラリ
import pandas as pd
from cq_modules import cq_pca
                    #②主成分分析結果を表示するためのライブラリ
                    #③データの読み込み
df = pd.read_csv("drink_data.txt", sep="\t",
                                        index_col=0)
                    #④主成分分析
pca = PCA()         #主成分分析の準備
pca.fit(df)         #主成分分析の実行
                    #⑤結果の表示
cq_pca.summay(pca)
cq_pca.plot(pca)
cq_pca.biplot(pca, df, 1, 2, scale=0,
                                      pc_biplot=True)
```

3に変えると
第3主成分が表示される

リスト3-4　テスト・データを追加する（pca_drink_test.py）

```
#⑥テストデータの読み込み
df_test = pd.read_csv("drink_test_data.txt",
                              sep="\t", index_col=0)
cq_pca.biplot(pca, df, 1, 2, new_data =
                df_test, scale=0, pc_biplot=True)
```

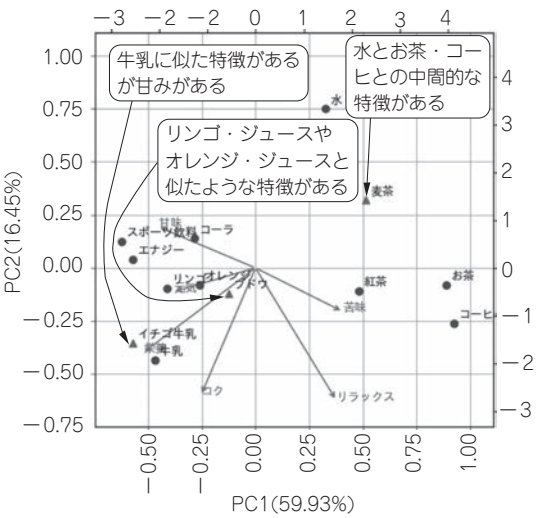


図3-4　主成分分析することで追加した飲み物データの特徴が推測できる

4　結果の読み取り方

寄与率と累積寄与率

● 寄与率は軸に含まれる情報量

飲み物データに焦点を当てて，結果を読み取る方法を詳しく説明します．まず，実行したときに表示される寄与率［**図3-3**（b）で示した棒グラフ］について説明します．

寄与率とは，その軸にどのくらいの情報量があるかを示しています．飲み物データの場合は，約60％が第1主成分（**図2-1**の横軸）に含まれていて，約16％が第2主成分（**図2-1**の縦軸）に含まれています．第3主成分以降は**図2-1**のグラフに表していませんが，計算はできます．

リスト3-3に示すpca_drink.pyの最終行の数値を1，2から1，3に変更します．**図4-1**（a）に示す実行結果から，飲み物の例では第3主成分の寄与率は約13％ですので，第2主成分と近い情報量が含まれています．これを数字だけで読み取るのは大変なので，グラフに表したものが**図3-3**（b）となります．

● 累積寄与率を見れば必要な主成分の数が分かる

累積寄与率は，寄与率を合計した「折れ線グラフ」となります．第1主成分は約60％，第2主成分は約16％が含まれていますので，第2主成分までの累積寄与率は約76％となります．最後の主成分まで合計すると100％になります．このグラフを見ることで，幾つの主成分まで考慮したらよいのかが分かります．

別の主成分を横軸と縦軸にしてグラフを作ってみる

図2-1に示した飲み物データについて，第1と第3主成分のグラフを作ると**図4-1**（a）となります．第3主成分と第2主成分のグラフを作ると**図4-1**（b）となります．**図2-1**に示した第1主成分と第2主成分のグラフだけでなく，この2つも分析に加えると，より詳しく分析できるようになります．

図4-1（b）は**リスト3-3**の最終行にある1，2を3，2に変えることで表示できます．

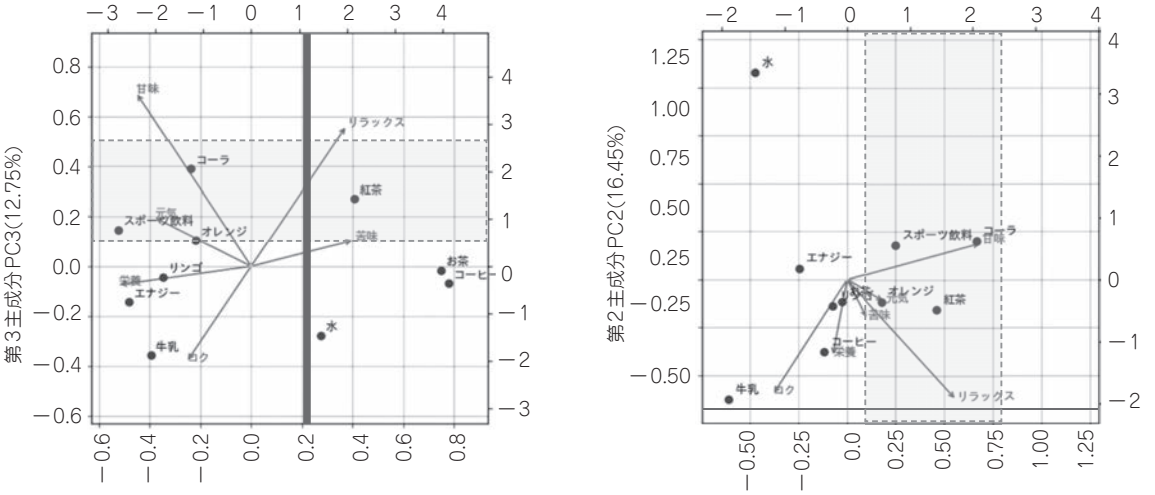
ここまでの知識を元に図2-1の結果を読み取る

さて，読み取るべき**図2-1**の結果について説明します．横軸，縦軸はそれぞれ第1主成分，第2主成分を表しています．そして黒い点は主成分得点と呼ばれるもので，それぞれの軸へ写像したものとなります．一方，矢印は主成分の方向を示しています．

● 味の特徴

▶苦み成分

どの順番で考えてもよいのですが，ここではまず，苦味の矢印に着目しながら考えてみます．右に行くと苦み成分が強くなることを示しています．確かに紅茶，お茶，コーヒは苦み成分がありますね．



（a）横軸：第1主成分　縦軸：第3主成分

（b）横軸：第3主成分　縦軸：第2主成分

図4-1　横軸／縦軸を別の主成分にしてグラフ化すると詳細な分析ができる

第2部

第5章

画像中のどこに何が写っているのかが分かる「YOLO」

牧野 浩二

YOLO（You Only Look Once）は物体検出に役立つアルゴリズムです．YOLOは，画像から80種類の物体しか検出できませんが，追加学習をすることで，さまざまな物体を検出できるようになります．

本章では追加学習の方法，サンプル画像を利用した物体検出，任意の画像や動画の物体検出を行います．

1　できること

● 画像に写っているものを検出できる

YOLOは，画像中のどこに何が写っているのかを見分けることができます．例えば**図1-1**のように，多くの車や人がどこにある（いる）のかを見分けたり，**図1-2**のように犬/スキー/リュックサック/人を見分けたりできます．

● 応用例

これを使った応用例としては以下が考えられます．

▶果物/野菜

- 木に何個なっているのか
- 食べごろになったか
- 葉に虫が付いているのか
- お米が病気になっていないか

▶人

- 人が来たのか
- 何人くらい集まっているのか

▶車

- 渋滞しているのか
- 何台通ったのか

▶病気

- 検査
- 骨折の箇所

図1-3のようにカプセル内視鏡の画像を使って病気を発見する研究も行われています．

YouTubeに，ナミブ砂漠に設置した人工の水飲み場をライブ・カメラで配信（Namibia：Live stream in the Namib Desert）しているものがあります．稀にダチョウやキリンが来ることがありますが，なかなか見ることはできません．そこで，希少動物が現われたら通知するといった使い方もできそうです．

図1-1　多くの車や人を検出できる
YOLOv3の公式ホーム・ページから

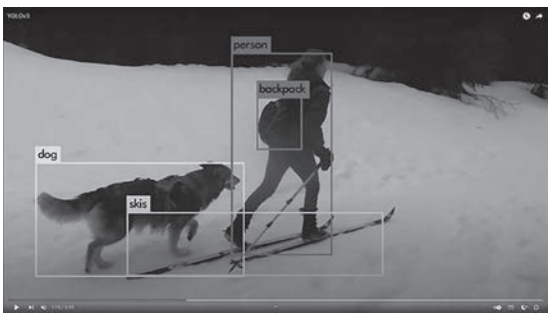


図1-2　人や犬のほかにスキーなどの物も検出できる
YOLOv3の公式ホーム・ページから

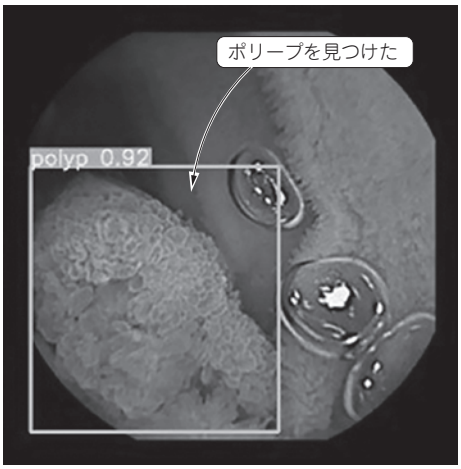


図1-3　医療分野でも応用されている（山梨大学 渡辺寛望研究室 提供）

2　イメージをつかむ

● 物体認識と物体検出の違い

本章で取り上げるYOLOは，物体を検出してくれるアルゴリズムです．本書で紹介してきた物体認識とはどのように異なるのでしょうか．まず，物体認識とは画像の中に何が映っているのかを分類するものでした．例えば，**図2-1**のように犬または猫の画像を入力すると，犬とか猫とかを答えるものでした．一方，物体検出とは，**図2-2**のように画像の中のどこに何が映っているかを示すものです．

YOLOは，犬と人間が一緒に写っている画像でも，犬と人間が別々に囲まれ，

「1 person, 1 dog, 1 frisbee」

といった具合に結果が得られます．なお，**図2-2**の画像では左上のものがフリスビーとして検出されています．

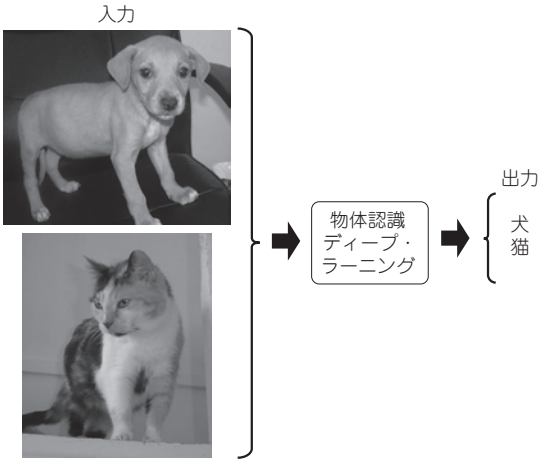


図2-1　物体認識は画像中に何が写っているのかを分類する
https://www.robots.ox.ac.uk/~vgg/data/pets/の画像を使用

● YOLOの検出精度

YOLOはどのくらいの物を見分けることができるのでしょうか．追加で学習しない場合は，80種類の物

図2-2　物体検出は画像中のどこに何が映っているのかを示す
https://www.robots.ox.ac.uk/~vgg/data/pets/の画像を使用

表2-1　YOLOv5で最初に検出できるのは80個(例として一部を表示)

名　称	意　味
person	人
bicycle	自転車
car	車
motorcycle	オートバイ
airplane	飛行機
bus	バス
train	電車
truck	トラック
boat	ボート
traffic light	信号機
fire hydrant	消火栓
stop sign	一時停止の標識
parking meter	パーキング・メータ
bench	ベンチ
bird	鳥
cat	猫
dog	犬
horse	馬
sheep	羊
cow	牛

名　称	意　味
elephant	象
bear	クマ
zebra	シマウマ
giraffe	キリン
backpack	バックパック
umbrella	傘
handbag	ハンドバッグ
tie	ネクタイ
suitcase	スーツ・ケース
frisbee	フリスビ
skis	スキー
snowboard	スノーボード
sports ball	スポーツボール
kite	カイト
baseball bat	野球のバット
baseball glove	野球のグローブ
skateboard	スケート・ボード
surfboard	サーフボード
tennis racket	テニス・ラケット
bottle	ボトル

名　称	意　味
wine glass	ワイン・グラス
cup	カップ
fork	フォーク
knife	ナイフ
spoon	スプーン
bowl	ボウル
banana	バナナ
apple	リンゴ
sandwich	サンドイッチ
orange	オレンジ
broccoli	ブロッコリ
carrot	にんじん
hot dog	ホットドッグ
pizza	ピザ
donut	ドーナツ
cake	ケーキ
chair	椅子
couch	ソファ
potted plant	鉢植え
bed	ベッド

体を見分けることができます．**表2-1**に一部を示します．これは`coco128.yaml`に書かれています．

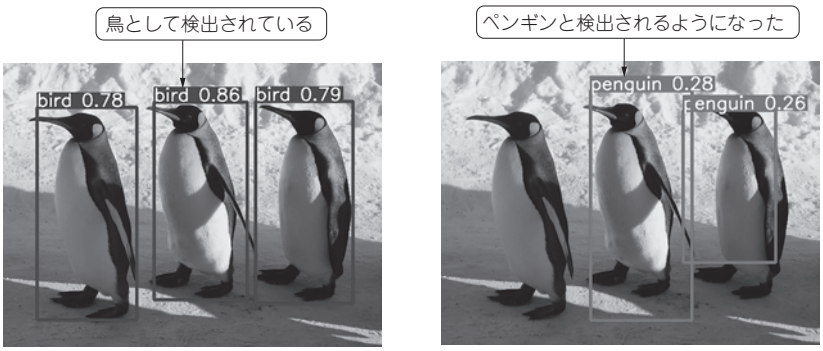
● 追加学習ができることがメリット

YOLOのメリットは，例えばペンギンやトマトといった学習済みデータの中に入っていないものでも，追加で学習をすると見分けることができるようになる点です．

図2-3(**a**)のように，追加で学習していない場合は，ペンギンは鳥と検出されてしまいます．

一方で，追加で学習するとペンギンと判定できるようになります［**図2-3**(**b**)］．追加で学習というところがYOLOの良いところで，既にある学習モデルを少しだけ学習し直すことで，新たな物体を見分けられるようになります．

この学習方法が人間に似ていて，例えば初めてトマトを見た人は，リンゴとかミカンに似ているけど，これはトマトというものなのだとして覚えることでしょう．このように，今までの知識をうまく使って再構成することで，ほんの少しの学習データ量で見分けることができるようになります．



(**a**) 学習前　　(**b**) 追加学習後

図2-3　YOLOは追加学習すれば未知の物体も検出できるようになる
`https://www.photo-ac.com/main/detail/1917360?title=%E3%83%9A%E3%83%B3%E3%82%AE%E3%83%B3%E3%81%AE%E8%A1%8C%E9%80%B2`の画像を使用

3　プログラムを動かしてみよう

YOLOは現在v5(バージョン5)がよく使われています(2022年執筆当時)．本章では，このYOLOv5を使って物体検出を行います．筆者提供のプログラムは本書サポート・ページから入手できます．

● ColabでYOLOを使えるようにする

まずは，Colabを起動して，YOLOv5を使う準備を行います．次のコマンドを入力して実行してください．

```
!git clone https://github.com/
                    ultralytics/yolov5
%cd /content/yolov5/
```

yolov5/requirements.txtを開いて，Pillow>=10.0.1をPillow>=10.0.0に変更して保存し，以下を実行します．

```
!pip install -qr requirements.txt
```

Colabは起動するたびに，このインストール作業が必要となります．実行すると，**図3-1**のようにyolov5フォルダが出来上がります．これで準備完了です．なお，cd /content/yolov5/としているので，yolov5ディレクトリにいることを前提とします．

● サンプル画像を使う

使用するプログラムはYOLO_test.ipynbです．まずは用意されているサンプル画像で物体検出を行ってみましょう．使う画像はdata/imagesフォルダ(dataフォルダの下にあるimagesフォルダを意味します)にあるbus.jpg［**図3-2**(**a**)］です．実行は次のコマンドで行います．

```
!python detect.py --weights yolov5s.pt
--source data/images/bus.jpg
```

Colabで実行するときは「!」を最初に付けます．対象とするファイルは，「--source」の後ろに書きます．

▶実行結果

実行するとruns/detect/exp/フォルダにbus.jpgができます．これを開くと**図3-2**(**b**)となります．人とバスが検出できています．

▶フォルダ内に複数の画像がある場合のコマンド

ここで，フォルダ内に多くの画像がある場合，画像を1つずつ指定することは面倒な作業です．この場合は，--sourceにファイルでなくフォルダを指定すると，フォルダの中にある画像が全て処理されます．

例として，**図3-3**(**a**)のようにdata/imagesフォルダには，bus.jpgとzidane.jpgの2つの画像があるとします．これを，次のように指定すると**図3-3**(**b**)のように2つの画像が処理されます．

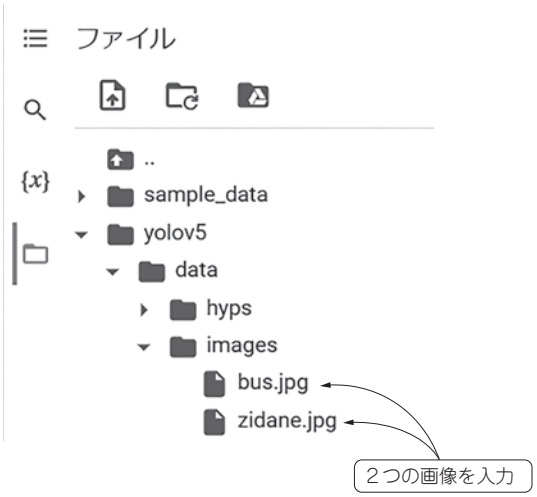
```
!python detect.py --weights yolov5s.pt
--source data/images
```

なお，出力フォルダを別のものにするために，自動的に番号が付きます．この場合は2回分の実行ですので，出力先はexp2となります．

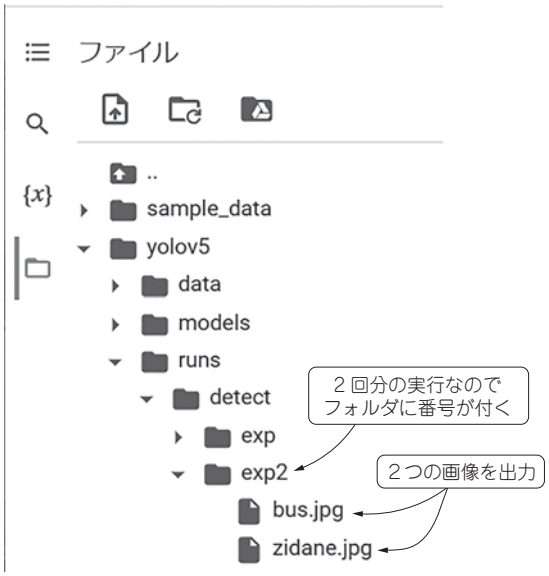
図3-1　YOLOv5をインストールするとフォルダができる

(**a**) 元画像（bus.jpg）　　(**b**) YOLOの実行結果

図3-2　サンプル画像で物体検出した結果(YOLOv5のインストール・フォルダに含まれている)



（a）2つの画像を入力とする



（b）出力先フォルダは変わるので注意

図3-3　複数の画像をまとめて処理することもできる

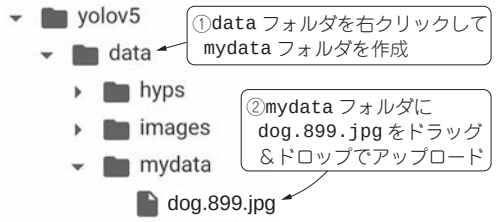


図3-4　まずは任意の画像をアップロードする

● 任意の画像を使う

▶画像の準備

自分で用意した任意の画像を使ってみましょう．まずは画像を用意してください．物体検出は多くの物が写っていた方が面白いので，例えばグーグルの画像検索で，

- たくさんの犬
- 登山
- 自転車レース

などのキーワードで検索します．ここでは`dog.899.jpg`という名前で画像を保存します．

▶画像のアップロード

まず，**図3-4**のように`data`フォルダの下に，`mydata`という名前のフォルダを作ります．その中に用意した`dog.899.jpg`をアップロードします．アップロードはドラッグ＆ドロップでできます．

▶プログラムの実行と実行結果

画像をアップロードしたら，次のコマンドを実行します．

```
!python detect.py --weights yolov5s.pt
--source data/mydata
```

実行後は，`runs/detect/exp3`フォルダに分類結果が出力されます．なお，`exp`の後ろの番号は，ここまで実行した回数によって異なります．実行結果を**図3-5**に示します．確かに多くの犬が検出できていることが分かります．それと同時に犬のベッドも検出できています．

● 動画を使う

▶撮影した動画

YOLOv5では，画像だけではなく動画（mp4形式）も対象となります．コマンドの変更は簡単で，`--source`で動画ファイルを選択します．

縦向きで動画を撮影するとうまく処理できませんので，横向きに撮影した動画を使ってください．ここでは，車やトラックが通過する動画（`cars.mp4`）で物体検出を行います．コマンドは次の通りです．使うプログラムは`YOLO_mydata.ipynb`です．

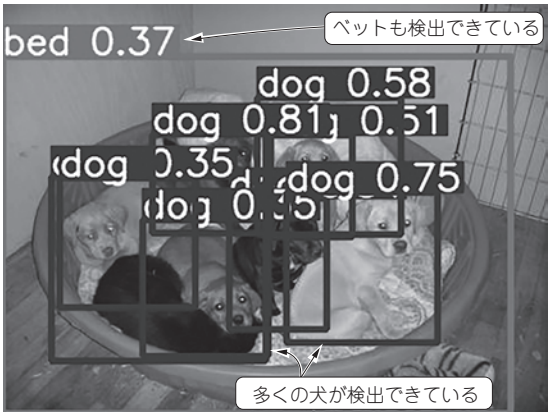
```
!python detect.py --weights yolov5s.pt
--source data/mydata/cars.mp4
```

実行結果を**図3-6**に示します．`runs/detect/exp`フォルダに分類結果が出力されます．

▶YouTubeの動画

撮影した動画以外にも，YouTubeにアップロードされている動画も使うことができます．ただし，長い動画は変換に時間がかかるので，30秒以下の画像を使って試すことをお勧めします．なお，YouTubeを使った場合は動画は滑らかではなくなってしまいます．

（a）本節用に用意した画像



（b）YOLOの実行結果

図3-5　任意の画像で物体検出した結果
https://www.robots.ox.ac.uk/~vgg/data/pets/の画像を使用

例として，東京ズーネットYouTubeチャネルには動物の動画がたくさん載っています．ここでは，

ある日のアルン―体重測定編（2021年8月12日撮影）
https://www.youtube.com/watch?v=PhMnCrPIAx0

という11秒の動画を対象とします．YouTubeの場合は，次のように--sourceにYouTubeのアドレスを書きます．使用するプログラムはYOLO_YouTube.ipynbです．

```
!python detect.py --weights yolov5s.pt
--source https://www.youtube.com/
watch?v=PhMnCrPIAx0
```

実行が完了すると，runs/detect/exp4に動画が生成されます．これを再生するとゾウが検出されます．そして，ゾウの足だけでもゾウと検出できたり，人間の片足だけでもpersonと検出できたりしています．他にも人が多く写っている動画などを用いると，それらが検出されてYOLOのすごさがよく分かります．

図3-6　動画でも物体検出できる
動画は編集部で撮影

● モデルを変更する

モデルの変更の仕方をここで紹介しておきます．物体検出は次のコマンドで実行しました．

```
!python detect.py --weights yolov5s.pt
--source data/images
```

このモデルを変えて実行したいときは，weightsの後ろのyolov5s.ptをyolov5n.ptやyolov5x.ptなどにすることで実現できます．

もっと体験したい方へ
電子版「AI自習ドリル：YOLOで物体検出」では，より多くの体験サンプルを用意しています．全24ページ中，10ページは本章と同じ内容です．
https://cc.cqpub.co.jp/lib/system/doclib_item/1589/

4　実践！自分のデータをYOLOで分析

既に述べたように，YOLOは用意された80個の分類はできますが，それ以外の分類をさせたいときは新たに学習をし直す必要があります．ここで，「学習をし直す必要」と書きましたが，まっさらな状態から学習するのではなく，学習済みのパラメータを使って追加で学習すると，うまく分類できるようになります．

■ 体験1…トマトを追加学習する

トマトはオリジナルのYOLOで分類できる80個には含まれていません．そのため，トマト画像を対象とすると**図4-1**に示すようにリンゴ(apple)やオレンジ(orange)として検出されます．ちなみに，リンゴやオレンジの画像はちゃんと分類できます．

そこで，トマトのデータを集めてきて学習させることで，**図4-2**のようにトマトとして検出できる「学習済みモデル」を作るための手順を説明します．なお，トマトだけで学習すると，リンゴやオレンジもトマトとして検出されます．

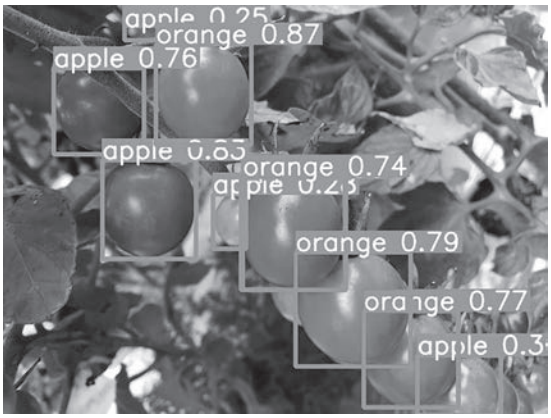


図4-1　最初はトマトはアップルやオレンジと認識される
https://www.photo-ac.com/の画像を使用

図4-2　追加学習してトマトと検出できるようにする
https://www.photo-ac.com/の画像を使用

● ステップ1：データを集める

トマトの画像データを多く集めます．筆者は画像検索でトマトを検索し，その画像を20枚使いました．

● ステップ2：学習データを作成する

学習するためのデータを作成するには，LabelImageというソフトウェアを使用します．ここでは使い方から説明します．

▶1：本体のインストール

ここではAnacondaとWindowsで実行する方法について示します．Anaconda Promptを起動し，下記のコマンドでインストールします．

```
pip install labelImg
```

インストール時は**図4-3**が表示されます．インストールが終わった後，以下のコマンドで起動します．

```
labelImg
```

▶2：YOLO用のデータに設定する

画像の処理を行う前にYOLO用のデータを作るための設定をしておきます．`labelImg`を実行し，**図4-4**のように左側の［PascalVOC］と書かれた部分をクリックしてYOLOにしておきます．これは画像を読み込んでからでもできますが，忘れてしまうことがありますので最初に行っておくことをお勧めします．

▶3：画像の読み込み

以上が終わったら，画像を読み込みます．左上の［Open］をクリックして画像を選択すると，**図4-5**のように画像が表示されます．

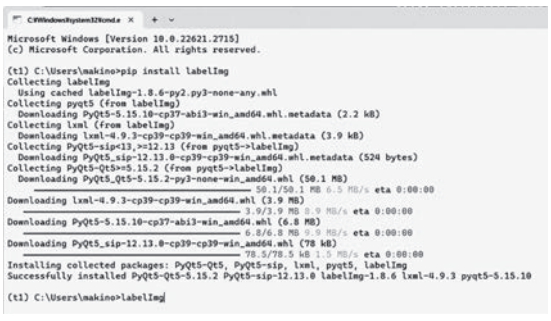


図4-3　labelImgのインストールと実行

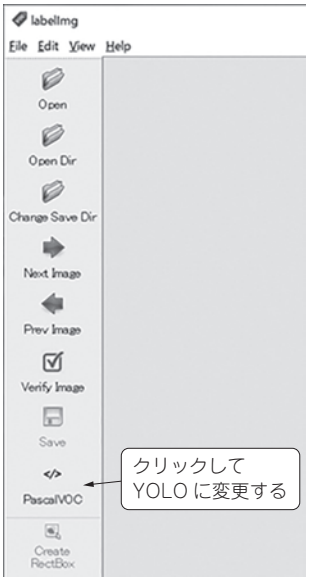


図4-4　labelImg.exe を実行したら PascalVOC をYOLOに変更する

図4-5 ［Open］をクリックして画像を読み込む

図4-6 ［CreateRectBox］をクリックしてトマトを囲む

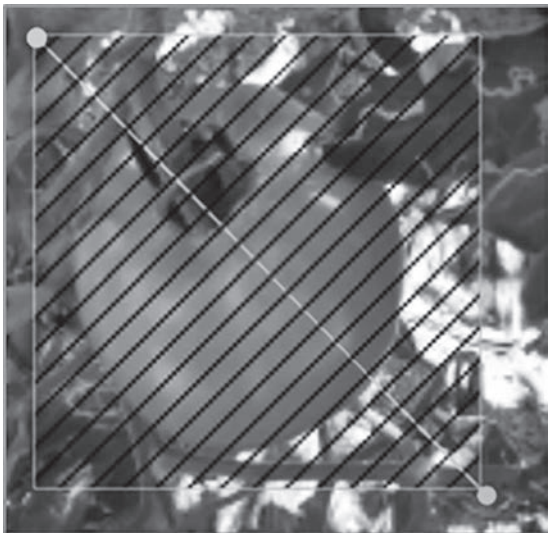


図4-7　ドラッグすると網掛けができる

▶4：学習対象の選択

画像を読み込んだら，この画像のどこにトマトがあるかといったデータを作成します．［CreateRectBox］をクリック（または「w」キーをクリック）すると，**図4-6**のようにマウス・ポインタの位置で交わるような線ができます．ここでトマトを囲むようにドラッグします．ドラッグすると**図4-7**のように網掛けができます．

マウス・ボタンを離すとボックスが表示されますので，**図4-8**のように「tomato」と入力して［OK］をクリックします．すると，**図4-9**のようにトマトを囲む線が表示され，右側にtomatoと書かれます．

▶5：全ての学習対象を選択

同じようにして，別のトマトも囲みます．今度は，**図4-10**のようにラベルを選べるようになります．本節ではラベルがtomatoだけですが，複数のラベルがある場合は目的の物を選択して［OK］をクリックします．

大きさの変更は角にある（緑の）丸をドラッグすることで行います．また，囲み線を削除したい場合は，ボックスをクリックした後で［DeleteRectBox］をクリックします．

▶6：保存する

以上を繰り返して，全てのトマトを囲み，最後に［Save］をクリックして保存します．繰り返しの注意となりますが，YOLO用になっていることを確認してください．

［Save］をクリックすると，読み込んだ画像と同じフォルダに画像と同じ名前のテキストができます．

▶一気にラベルを付ける

ラベルがtomatoしかない場合は，毎回ボックスが出てきて選ぶのは面倒な作業です．そこで，右上の「Use defaut label」の右のテキスト・ボックスに「tomato」を入力しておいて，左のチェック・ボックスにチェックをしておくと，囲んだらすぐにtomatoラベルが付きます．

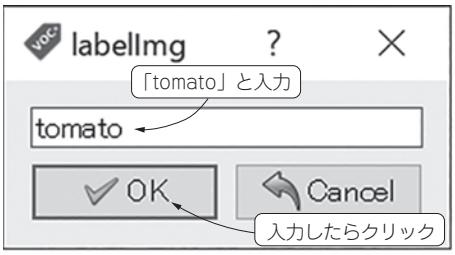


図4-8　ボックスを表示して「tomato」と入力する

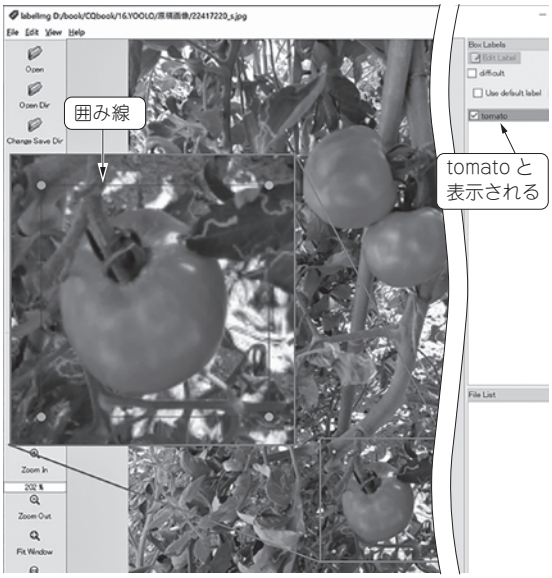


図4-9　トマトの囲み線が表示される

図4-10　別のトマトを囲むとラベルを選択できるようになる

リスト4-1　ラベルの内容…ラベルは0から順につく

```
0 0.578675 0.813281 0.209110 0.157812
0 0.654244 0.368750 0.260870 0.168750
0 0.508282 0.248438 0.275362 0.212500
```

リスト4-2　tomato.yamlの内容

```
train: /content/drive/MyDrive/YOLO/MyData/
val: /content/drive/MyDrive/YOLO/MyData/
test: /content/drive/MyDrive/YOLO/MyData/

# number of classes
nc: 1

# class names
names: ['tomato']
```

▶ラベルの確認

テキストの中身は**リスト4-1**であり，1文字目がラベルです．ラベルは0から順につきます．本節ではトマトだけなので0だけが書かれています．2つ目と3つ目は囲み線の中心位置，4つ目と5つ目は囲み線の大きさです．中心位置や囲み線の大きさは画像の大きさで割っているため，0～1までの値となります．

▶他の画像を選んで繰り返す

他の画像も同じように行います．［Next Image］をクリックすると次の画像が表示されて，作業ができます．また，［Prev Image］をクリックすると，前の画像が表示されます．このとき，ラベルや囲み線の情報も一緒に読み込まれます．以上を繰り返して学習データ画像を作ります．

● ステップ3：拡張子がyamlのファイルを作る

以上が終わったら，もう1つファイルを作る必要があります．それは，拡張子をyamlとしたファイルで，中身は**リスト4-2**となっています．なお，#の後ろはコメントです．

train，val，testの後ろにそれぞれ学習データ，検証データ，テスト・データの保存先を書きます．本節では作成したデータの数が少ないので，全て同じとしています．この後で，MyDataはGoogleドライブのマイドライブに作ったYOLOフォルダにアップロードすることにしています．

その下のncは，分類するクラスの数です．本節ではトマトだけなので1となります．そして，namesの後ろにはクラスの名前を並べます．本節ではtomatoだけです．

● ステップ4：学習データをアップロードする

学習データのアップロード方法は，Colabに直接アップロードする方法とGoogleドライブにアップロードしておいたものを使う方法があります．Colabへ直接アップロードすると，ウィンドウを閉じたときや接続が切れたときにアップロードしたデータが消えてしまいます．そこで，ここではGoogleドライブにアップロードして，それを使う方法を説明します．

▶1：データをGoogleドライブへアップロード

Googleドライブを開き，ドラッグ＆ドロップでデータをアップロードします．ここでは，マイドライブ

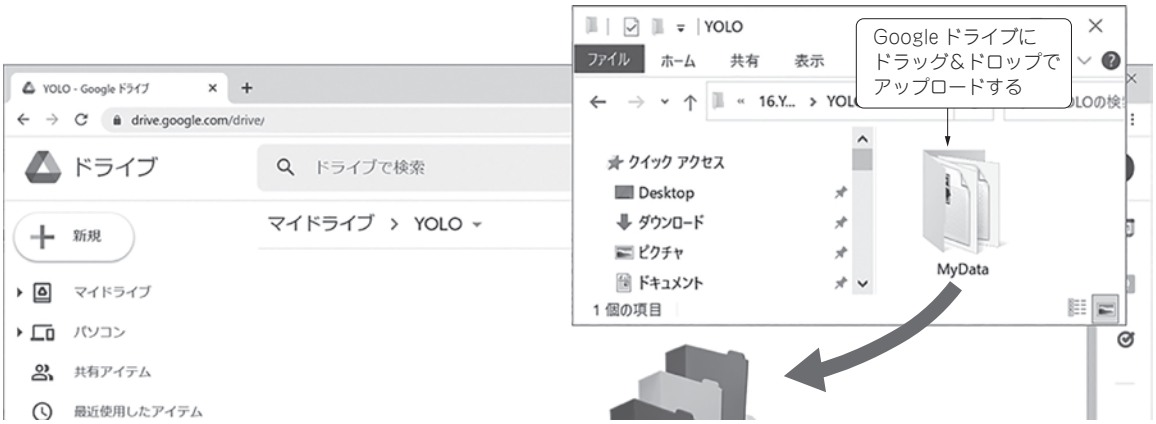


図4-11　Googleドライブに学習データをアップロードする

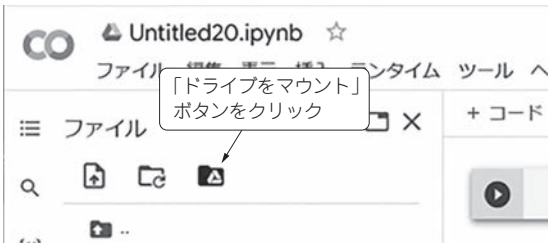


図4-12　マウント・アイコンをクリックする

図4-13　マウントできればdriveフォルダ表示される

にYOLOフォルダを作り，その中にMyDataをアップロードすることとします(**図4-11**)．

▶2：Colabから使えるようにする．

左上のドライブのマウント・アイコンをクリック(**図4-12**)して，ドライブと接続します．接続するとdriveフォルダが表れ，その中にMyDataが入っていることが確認できます(**図4-13**)．

● ステップ5：学習する

アップロードした学習データを用いて学習しましょう．学習はGPUを使わないと時間がかかります．そこで，［ランタイム］-［ランタイムの変更］をクリックし［GPU］を選択します．

ドライブのマウントがしてあれば，ランタイムの変更後に自動的に接続されますが，YOLOv5のインストールはしていない状態に戻ります．そのため，インストールから行います．GPUを使って計算できる時間は限られていますので，その時間を超えるとしばらくGPUが使えなくなります．

▶コマンド

学習は次のコマンドで行います．

```
!python train.py --batch 16 --epochs 200
--data /content/drive/MyDrive/YOLO/MyData/
tomato/tomato.yaml --weights yolov5s.pt
```

ポイントは，dataの引数にtomato.yamlを指定する点と，エポック数(学習回数に相当)を200としている点です．

● ステップ6：学習結果を利用して物体検出

学習後の物体検出は，次のコマンドで行います．

```
!python detect.py --weights /content/
yolov5/runs/train/exp/weights/best.pt
--source data/images/tomato.jpg
```

本章第3節で行った物体検出との違いは，--weightsの後ろが学習したパラメータの中で最も良いパラメータ(best.pt)を使うように設定しているところです．

best.ptは，学習後に表示された学習パラメータが保存されているフォルダ(runs/train/exp)の下のweightsフォルダの中に生成されます．これを実行すると，**図4-2**が表示され，確かにトマトを見分けることができています．

◆参考文献◆

(1) 写真のフリー素材サイト．
https://www.photo-ac.com/

まきの・こうじ

6　数式で理解するQラーニング

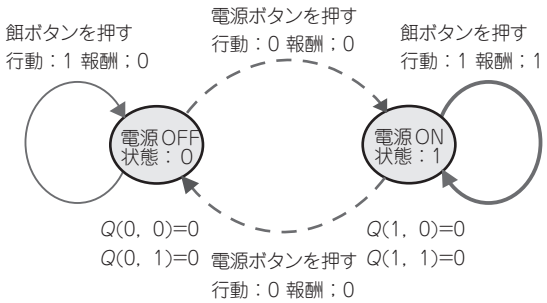
Qラーニングとは，各状態にQ値と呼ばれる道しるべを付ける方法です．道しるべを試行錯誤しながら自動的に更新して作成します．まずQ値とは何かについて説明し，その後Q値を更新する方法をスキナーの箱を例にとり，ネズミの行動と併せて数式と計算で説明します．

Q値とは

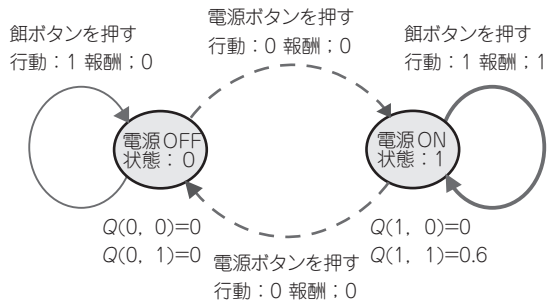
Qラーニングで最も重要な役割を果たすのがQ値です．Q値は「ある状態である行動を選ぶ基準」となる値であり，$Q(s, a)$と表します．なお，sは状態，aは行動です．

図5-3にQ値や状態，行動を書き込んだ**図6-1**を用いて説明します．

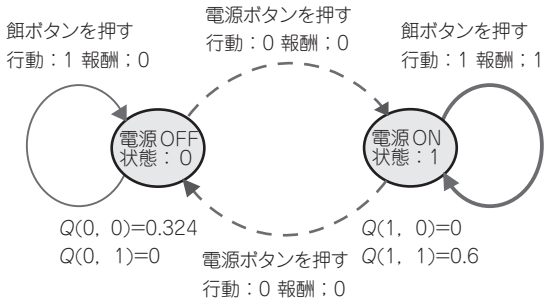
まず，電源OFFのときには2つの行動をとることができます．前節で状態と行動を数値で表しました．これを使うと電源OFFは$s = 0$となり，電源ボタンを押す行動は$a = 0$，餌ボタンを押す行動は$a = 1$と表すことができます．



(a)電源OFF



(b)電源ONのときに餌ボタンを押す



(c)もう1度電源OFFにしておいて電源ボタンを押す

図6-1　図5-3の状態と行動を数字で表した

▶電源OFF

そこで，電源OFFの状態のときのQ値は以下のように表すことができます．

- $Q(0, 0)$：電源OFFの状態($s = 0$)で電源ボタンを押す($a = 0$)場合のQ値
- $Q(0, 1)$：電源OFFの状態($s = 0$)で餌ボタンを押す($a = 1$)場合のQ値

▶電源ON

次に電源ONの状態($s = 1$)を考えます．このときも取りうる行動は2つありますので、Q値は以下のように表すことができます．

- $Q(1, 0)$：電源ONの状態($s = 1$)で電源ボタンを押す($a = 0$)場合のQ値
- $Q(1, 1)$：電源ONの状態($s = 1$)で餌ボタンを押す($a = 1$)場合のQ値

Q値に従った行動

Q値はエージェントが行動を選ぶ基準として使う値です．ここでは例としてQ値を以下のようにしたとき，ネズミがどのような行動をとるかを考えてみます．

- $Q(0, 0) = 0.8$　$Q(0, 1) = 0.2$
- $Q(1, 0) = 0.1$　$Q(1, 1) = 0.5$

▶電源OFF

初期状態で電源オフの状態($s = 0$)にあるとします．このときに参照するQ値は$Q(0, 0)$と$Q(0, 1)$となります．それぞれ0.8と0.2ですので，値が大きいQ値は$Q(0, 0)$です．

このことから0の行動，すなわち電源ボタンを押す行動をとることになります．これにより電源ONの状態になります．

▶電源ON

次に電源ONの状態($s = 1$)にあるときを考えます．

このときに参照するQ値は$Q(1,\ 0)$と$Q(1,\ 1)$となります．値が大きいQ値は$Q(1, 1)$ですので，1の行動(餌ボタンを押す行動)をとることになります．

以上のように，Q値を参照することで餌(報酬)が得られる行動が得られます．

Q値の更新の基本式

Qラーニングはこの Q値を行動するたびに自ら更新していく学習方法です．Q値を更新するための式を次の式(1)に示します．

$$Q(s,\ a) \leftarrow (1-\alpha)\,Q(s,\ a) + \alpha\,(r + \gamma \max Q) \quad \cdots (1)$$

ただし，s：状態，a：行動，r：報酬，α(学習率)とγ(割引率)：0 ～ 1までの定数，$\max Q$：行動aをとった際に遷移する次の状態の中で最も大きいQ値とする．

Q値は式(1)に従ってエージェントが行動するたびに更新されます．しかし，この式を見ただけでどのように更新されるのか分かりにくいかと思います．以下では，スキナーの箱を例にとり、ネズミの行動と合わせて更新の仕方を数値を用いて計算していきます．

● 初期状態

まず，初期状態でのQ値は全て0とします．そして初期状態のQ値を**図6-1(a)**に示します．この図のように各状態にQ値が設定されていて，行動の数だけQ値があります．スキナーの箱の場合は行動として電源ボタンを押す($a = 0$)と餌ボタンを押す($a = 1$)の2 種類があります．

● 行動1…電源ボタンを押す

早速，ネズミが行動したとします．Q値は道しるべなので，値の大きい方の行動をします．しかし，最初は両方とも0ですから，ランダムで行動を選ぶことになります．ここでネズミは電源ボタンを押したとしましょう．行動する前の状態は電源OFF($s = 0$)であり，電源ボタンを押した($a = 1$)ため，電源ON($s = 1$)に遷移します．この行動では報酬は得られません($r = 0$)．これにより式(1)に値を当てはめるとQ値は以下のように更新されます．なお，$a = 0.6$，$\gamma = 0.9$としました．

$$Q(0,\ 0) \leftarrow (1-0.6) \times 0 + 0.6\,(0 + 0.9 \times \max Q) \quad \cdots (2)$$

ここで$\max Q$が残っています．これは遷移先のQ値の最大値という意味です．遷移先は**図6-1(a)**の右の丸で囲まれた状態です．Q値はどちらも0［$Q(1,\ 0) = 0$，$Q(1,\ 1) = 0$］ですので，$\max Q$は0となります．

$$Q(0,\ 0) \leftarrow (1-0.6) \times 0 + 0.6\,(0 + 0.9 \times 0) = 0 \quad \cdots (3)$$

電源OFFの状態($s = 0$)で電源ボタンを押した($a = 0$)ときのQ値の更新を**表6-1**に示します．今回の行動ではQ値は変更がありませんでした．

表6-1　行動1によって更新されたQ値

Q(状態，行動)	更新前のQ値	更新後のQ値
$Q(0,\ 0)$	0	0
$Q(0,\ 1)$	0	0
$Q(1,\ 0)$	0	0
$Q(1,\ 1)$	0	0

● 行動2…餌ボタンを押す

次に電源がONになったときの行動を考えます．この場合もQ値がともに0なので，偶然にネズミが餌ボタンを押したとしましょう．このときは状態が電源ON($s = 1$)，行動が餌ボタン($a = 1$)となり報酬が得られますので，以下の式(4)のようになります．なお，$\max Q$は状態遷移図から電源ONの状態ですので，先ほどと同じように$\max Q$は0となります．

$$Q(1,\ 1) \leftarrow (1-0.6) \times 0 + 0.6\,(1 + 0.9 \times 0) = 0.6 \quad \cdots (4)$$

この後続けて学習してもよいのですが，説明のために初期状態から始めることとします．このときのQ値を書き込んだ状態遷移図は**図6-1(b)**となります．

先ほどと同じように電源ONの状態($s = 1$)で餌ボタンを押した($a = 1$)ときのQ値の更新を**表6-2**に示します．今回の行動では報酬が得られたためQ値が更新されました．

表6-2　行動2によって更新されたQ値

Q(状態，行動)	更新前のQ値	更新後のQ値
$Q(0,\ 0)$	0	0
$Q(0,\ 1)$	0	0
$Q(1,\ 0)$	0	0
$Q(1,\ 1)$	0	0.6

● 行動3…初期状態から始める

もう1度，初期状態から始めた場合もQ値は両方0ですので，ランダムに行動が選ばれます．ここでもたまたま電源ボタンが押されたとします［**図6-1(c)**］．このとき$\max Q$は0.6となります．そこでQ値は以下となります．

$$Q(0,\ 0) \leftarrow (1-0.6) \times 0 + 0.6\,(0 + 0.9 \times 0.6) = 0.324 \quad \cdots (5)$$

電源ON状態になった後の行動を考えます．電源ボタンを押すQ値は0で，餌ボタンを押すQ値は0.6ですので，餌ボタンを押すことになります．

表6-3　行動3によって更新されたQ値

Q(状態，行動)	更新前のQ値	更新後のQ値
$Q(0,\ 0)$	0	0.324
$Q(0,\ 1)$	0	0
$Q(1,\ 0)$	0	0
$Q(1,\ 1)$	0	0.6

これを**表6-3**に示します．今回の行動では報酬が得らていないにもかかわらず，行動によって遷移した先のQ値を使ってQ値が更新されました．

● 行動4…Q値が高い方を選ぶ

もう1度，初期状態から始めましょう．**表6-3**に従って行動することを考えます．

▶電源OFF

電源OFFの状態では，

- 電源ボタンを押すQ値である$Q(0,\ 0)$は0.324
- 餌ボタンを押すQ値である$Q(0,\ 1)$は0

ですから，電源ボタンを押すQ値の方が高いので，エージェントは電源ボタンを押します．これにより，電源ONになります．

▶電源ON

電源ONの状態では，

- 電源ボタンのQ値である$Q(1,\ 0)$は0
- 餌ボタンのQ値である$Q(1,\ 1)$は0.6

であり，比べると餌ボタンのQ値が高いのでエージェントは餌ボタンを押します．このようにして電源ボタンを押すべきかどうかは教えていなくても電源ボタンを押せるようになります．

稿末に演習問題を用意しました．そちらもご覧ください．

7　プログラムの説明

強化学習は便利なライブラリがあるわけではありません．そのため，プログラムを全て作る必要があります．ここではスキナーの箱を対象としたプログラムの説明を行います．フローチャートを**図7-1**に示します．

● プログラムの構成

プログラム(**リスト7-1**)は大きく分けて3つの部分から成り立っています．

▶1．シミュレータ・クラス

環境を設定する部分(シミュレータ・クラス)です．これは外箱に相当します．この部分でボタンを押したり，状態を変えたり，報酬を与えたりします．

▶2．Q値クラス

行動を決める部分(Q値クラス)です．これはネズミに相当します．現在の状態を入力するとQ値に従って行動を出力したり，Q値を更新したりする部分です．

▶3．実際に行動する部分

実際に行動する部分です．今の状態から行動を決めて，行動によって状態を変化させ，Q値を更新することを行います．決まった回数(5回)を1回の試行(エピソードとも呼ぶ)として，10回試行します．

なお，`skinner_QL.py`は本書サポート・ページからダウンロードできますので，Colabで試せます．

● プログラムの中身1…シミュレータ・クラス

スキナーの箱の箱に相当する部分です．それぞれのボタンを押したら状態がどのように変化するのかをシミュレーションし，電源ONのときに餌ボタンを押すと報酬を与えることを行います．ここで使われる変数とメソッドを以下に示します．

▶`self._state`：状態を表す変数(8行目)

電源OFFのとき0，電源オンのとき1となる変数です．

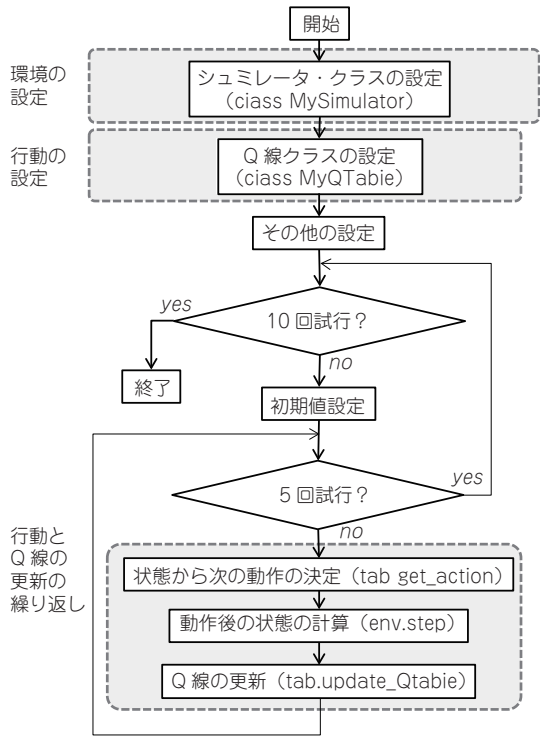


図7-1　スキナーの箱のねずみの動作を作る

▶`reset`メソッド：初期化

5回行動すると1回の試行が終わったとしてこのメソッドが呼ばれ，初期状態に戻します(7～9行目)．

▶`step`メソッド：行動による状態変化

行動(`action`)を引き数として与えて，状態(`self._state`)を変化させます．まず報酬を0(`reward=0`)としています(12行目)．その後，状態遷移図に従ってif文で場合分けが行われています．電源がONのときに餌ボタンを押す(`action=1`)の場合に

第1章

アンケート分析でよく使われる「ポジ・ネガ解析，単純集計，クロス集計」

牧野 浩二，足立 悠

● 人工知能とデータ・サイエンスとの関係

データ・サイエンスはAIを活用するために必要なものであり，今後の社会人スキルとして望まれる技術です．AIに入力するデータを扱う方法や，AIから得られたデータを扱うときにも，直接的ではないにしろ，データ・サイエンスの技術が役に立ちます．AIを活用するためにもさまざまなデータ・サイエンスの手法を知っておくことが大切です．

● 本章でトライすること

雑誌やテレビといったメディア，駅前やショッピング・モールといった外出先など，さまざまな場面でアンケートが行われています．アンケートは集めた後，それを活用することが重要です．本章ではアンケート結果を集計する方法として，単純集計，クロス集計，およびポジ・ネガ解析を紹介します．この集計と分析を行うことで，人間がアンケート結果を解析しやすくなります．アンケートはさまざまな場面で利用されていますので，データの分析方法を身に着けることで，仕事や研究に役に立つと思います．

本章ではGoogleスプレッドシートを使います．Googleスプレッドシートでは，Excelとほぼ同じことが無料でできます．Excelがインストールされていないパソコンもあるでしょうから，本章ではGoogleスプレッドシートを使いました．

1　できること

● 単純集計…最も基本的な集計法

単純集計とはアンケートの項目ごとに集計した結果のことです．例えばレストランで，「これまでに来店した回数」や「来店のきっかけ」などの，幾つかアンケートを取った項目について，それぞれ円グラフ(**図1-1**)や棒グラフ(**図1-2**)で表したものです．**図1-1**と**図1-2**から，単純集計を行うことで項目ごとの回答の傾向を把握できます．他方で，単純集計の方法では異なる項目間の関係性は分かりません

▶例1…レストランで「これまでに来店した回数」(図1-1)

▶例2…レストランへの「来店のきっかけ」(図1-2)

● クロス集計…項目間の関係性を調べる集計法

クロス集計は「睡眠時間と年代の関係」(**表1-1**)や，「会話の頻度と都市規模の関係」(**表1-2**)のように，異なる2つの項目間の関係性を調べるための集計方法です．さらに表にまとめるだけでなく，表中に網掛けをして強調することも行われています(**表1-1**)．クロス集計は2つの項目の関係性を，人間が見やすい形にまとめることができるため，さまざまな場面で使われています．

▶例1…睡眠時間と年代の関係(表1-1)

表1-1からは多くの年代で6～7時間の睡眠が多いことが分かります．他方で40代では6～7時間の睡眠が少ないことが分かります(5～6時間の睡眠が多い)．50代以降は，年齢が進むにつれて睡眠時間が延びる傾向にあることが読み取れます．このように，異なる2つの項目の関係性を人間が見やすい形にまとめることができるため，さまざまなところで使われています．

なお，このデータは筆者が作った架空のデータです．

▶例2…会話の頻度と都市規模の関係(表1-2)

「電話やEメールを含めて，普段どの程度，人(同居の家族を含む)と会話するか」という質問に対して，**表1-2**のように大・中・小都市と町村に分けた場合の比率や，男女の比率を表にしています．**表1-2**か

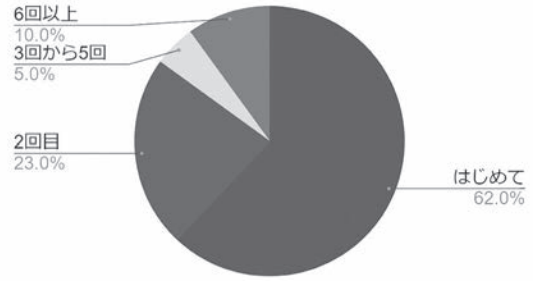


図1-1　単純集計の例1…レストランに「これまでに来店した回数」を円グラフで表現

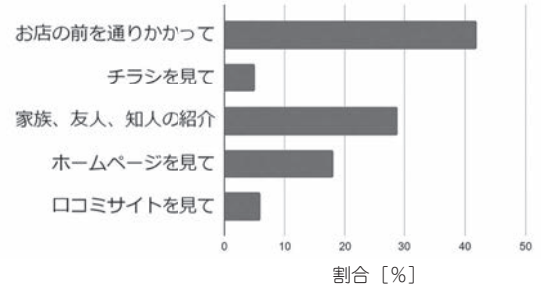


図1-2　単純集計の例2…レストランに「来店したきっかけ」を棒グラフで表現

表1-1　クロス集計の例1…睡眠時間と年代の関係

年齢／睡眠時間	5時間以下	5～6時間	6～7時間	7～8時間	8～9時間	9～10時間	10時間以上
10代	2	12	39	29	12	5	1
20代	5	23	33	22	11	4	2
30代	15	29	31	17	4	2	2
40代	18	35	24	15	4	3	1
50代	12	28	34	15	5	3	3
60代	3	7	19	40	21	5	5
70代以上	1	6	12	24	29	23	5

表1-2(1-1)　クロス集計の例2…会話の頻度と都市規模の関係

条　件		毎日［%］	2～3日に1回［%］	1週間に1回［%］	ほとんど話をしない［%］
都市規模	大都市	89.7	5.0	2.9	2.1
	中都市	93.2	4.3	1.1	1.4
	小都市	91.1	5.9	0.8	2.1
	町村	94.5	2.8	1.0	1.7
性別	男性	91.3	4.8	1.7	2.0
	女性	92.8	4.5	1.1	1.5
年齢別	55～59歳	94.6	2.2	1.1	1.6
	60～64歳	93.2	3.6	1.7	1.4
	65～69歳	91.1	5.6	1.2	2.1
	70～74歳	92.6	4.6	0.7	2.1
	75～79歳	89.8	7.0	2.3	0.9
	80歳以上	90	6.0	1.2	2.8

ら以下を読み取ることができます．

都市規模別にみると，「毎日」は大都市(89.7％)でやや低くなっています．性別に見ると大きな差はみられません．年齢別に見ると79歳以下では年齢が下がるほど「毎日」の割合がやや高くなる傾向がみられます．

● ポジ・ネガ解析…自由記述欄の解析

アンケートにはたいてい，自由記述欄があります．サンプルとなる文章を10例用意し，それぞれポジティブな単語とネガティブな単語，それ以外の単語がどの程度の割合で含まれているかを調べます．

◆引用文献◆

(1-1)人や地域とのつながりに関する事項，会話の頻度(Q 35)，内閣府．
https://www8.cao.go.jp/kourei/ishiki/h23/sougou/zentai/pdf/2-7.pdf