次世代AI人材育成訓練プログラム
テキスト

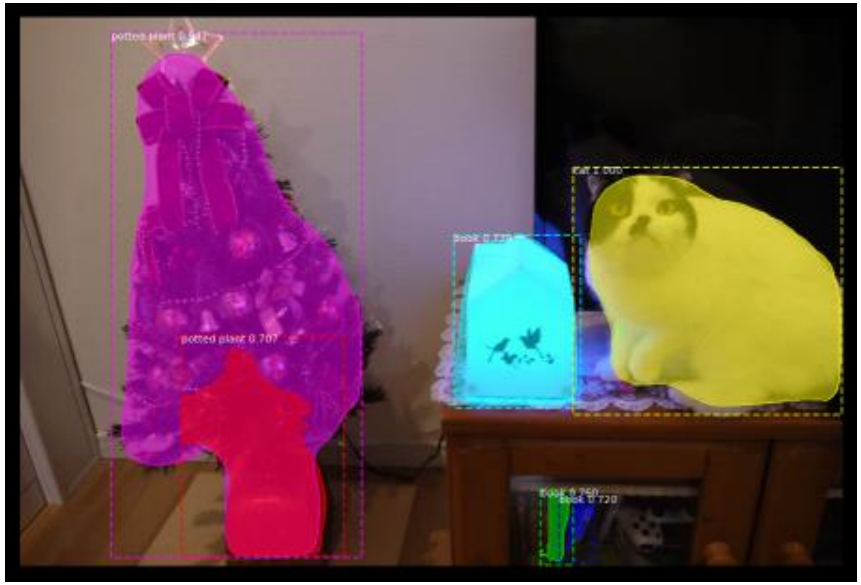
目次

第1章

AI基礎

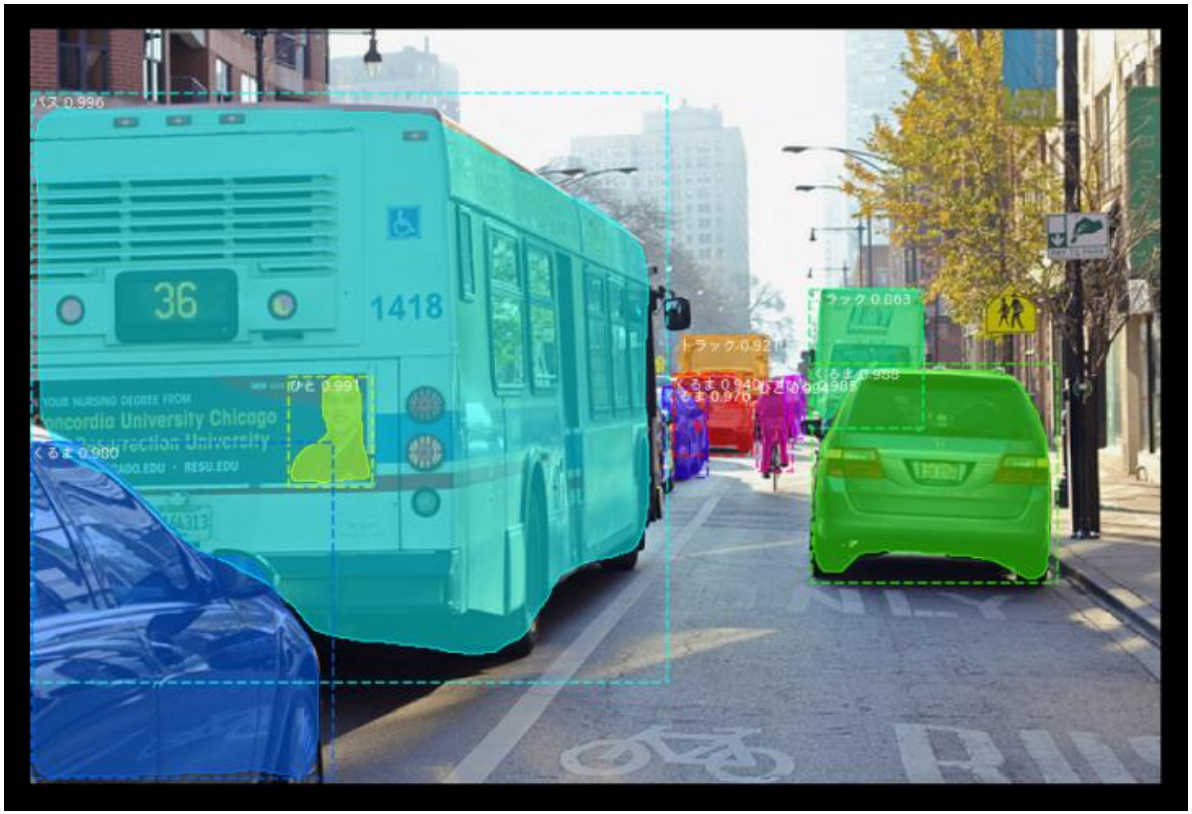
世間一般のAIのイメージ

一番よくある画像分類・判定
画像から物体を認識して領域を切り出す



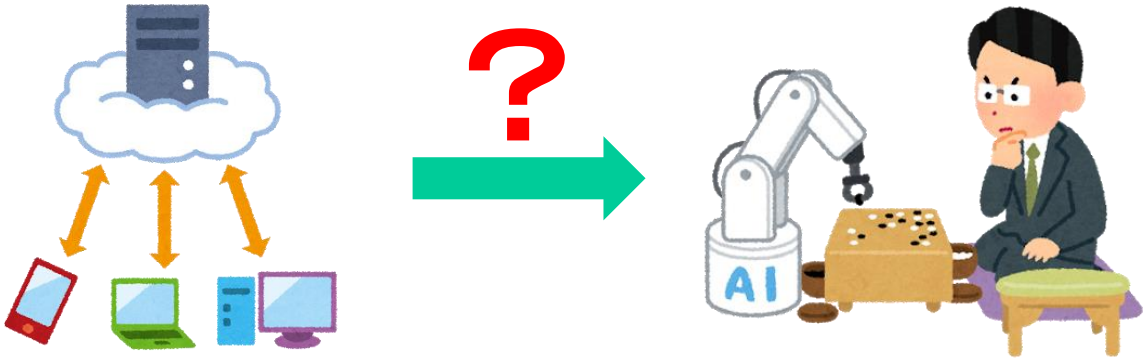
画像判定系の一番の応用といえば……

自動運転はこれをリアルタイムで実施



AIってバズワード？

10年前はクラウドがバズワード
今はAIが一番のバズワード
しかし、クラウドはすっかり根付いてもう欠かせなくなった
AIもバズワードになりつつも、この先欠かせなくなる？



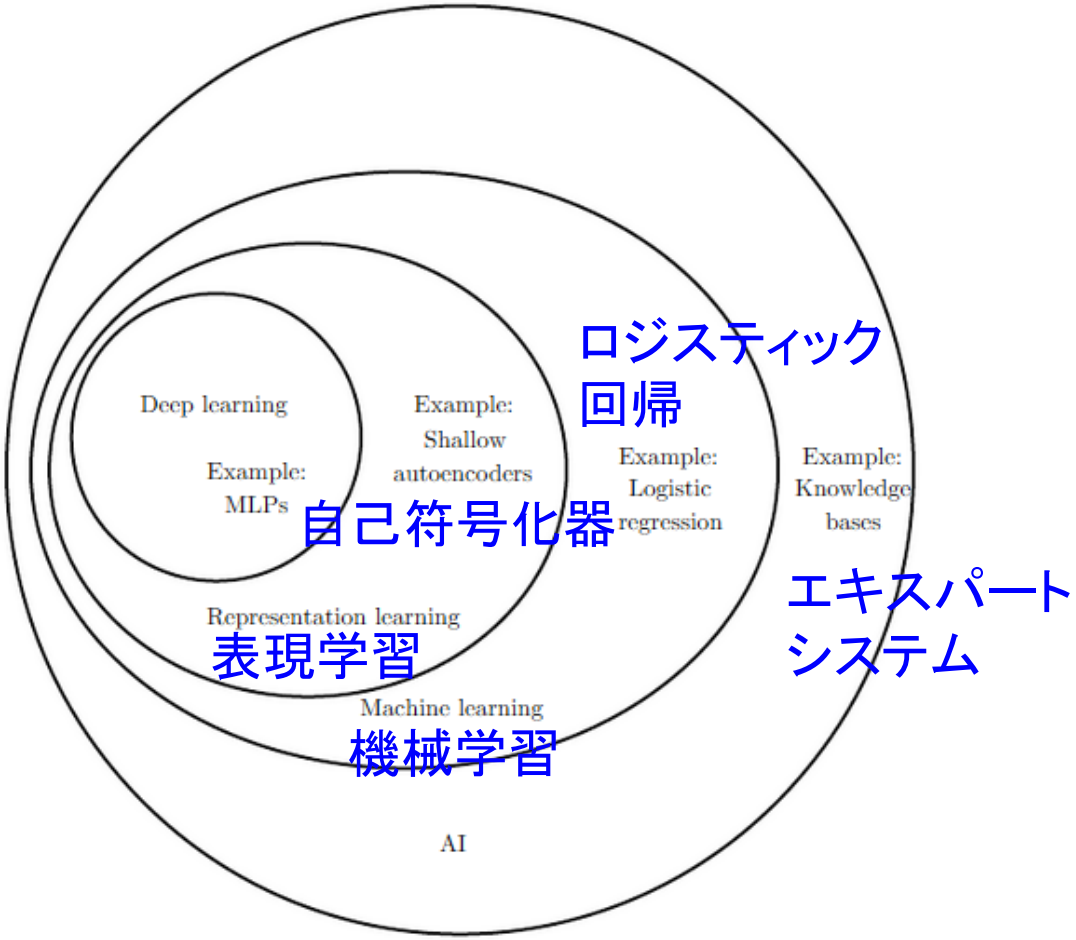
今日のテーマ

午前　大阪から：AIと機械学習と深層学習(Deep Learning)の区別をつける
AIは怖くない

午後　東京から：ビジネスへの応用(1)
まずざっくりとAIをどのようにビジネスへ活かせるか考える

AIの歴史と概念（再掲）

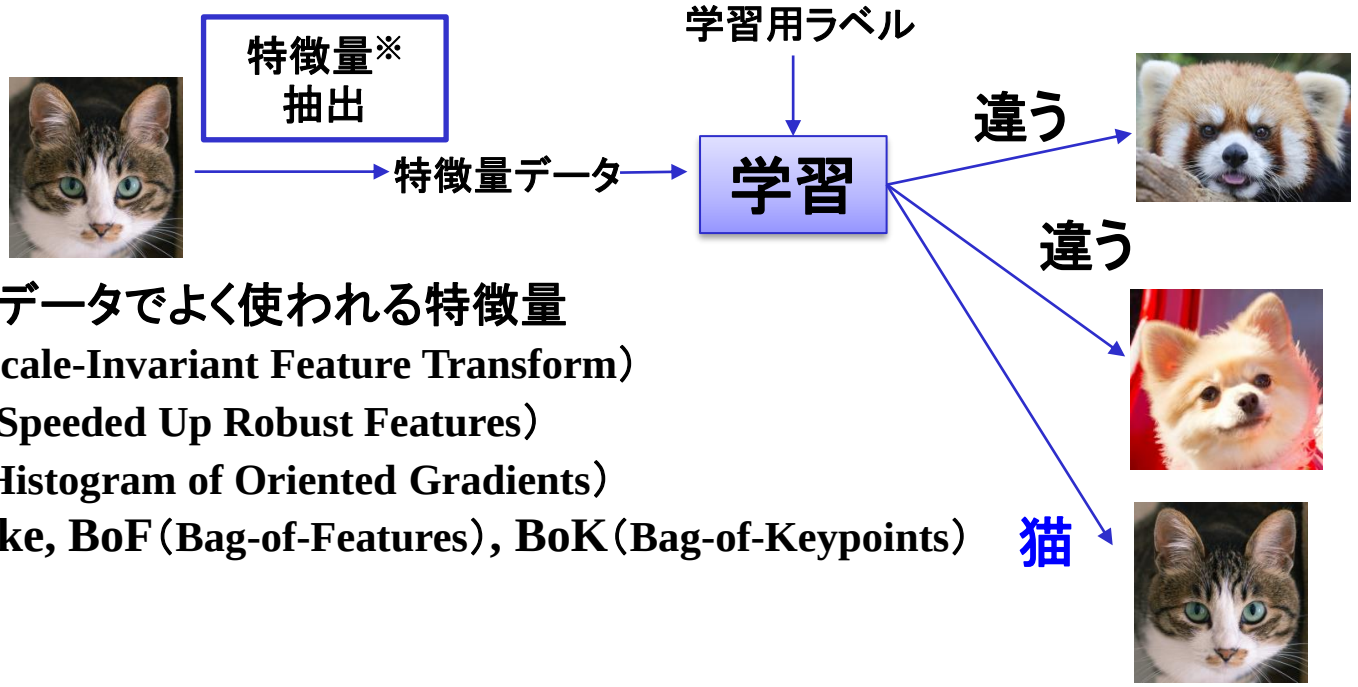
Goodfellow(Apple)
による概念図



Ian Goodfellow, Yoshua Bengio and Aaron Courville, "Deep Learning", The MIT Press, 2016

従来の機械学習

例）画像の物体認識 (OpenCVなど)



※画像データでよく使われる特徴量
SIFT（Scale-Invariant Feature Transform）
SURF（Speeded Up Robust Features）
HOG（Histogram of Oriented Gradients）
Haar-like, BoF（Bag-of-Features）, BoK（Bag-of-Keypoints）
etc.

機械学習では、認識すべき対象の着目点（＝入力データ）は人間が考える

例) OpenCVによる画像識別

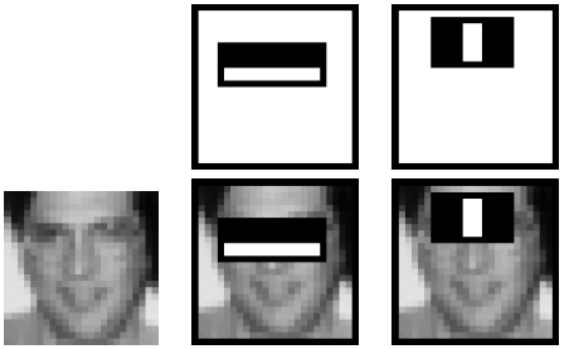
ハール(Haar)特徴量にもとづくCascade 識別器を使用

P. Viola (2001)によるオブジェクト検出の研究とR. Lienhart (2002)による改良がベースになっている

白と黒で構成される矩形を画像に適用し、対象画像の特徴量を作成する

例えば顔画像は、目の領域の画素は周辺よりも暗い、口の領域の画素が周辺より明るい、などの特徴が得られる

顔、目、正面、上半身、下半身、笑顔、などの識別器がOpenCVで使用できる

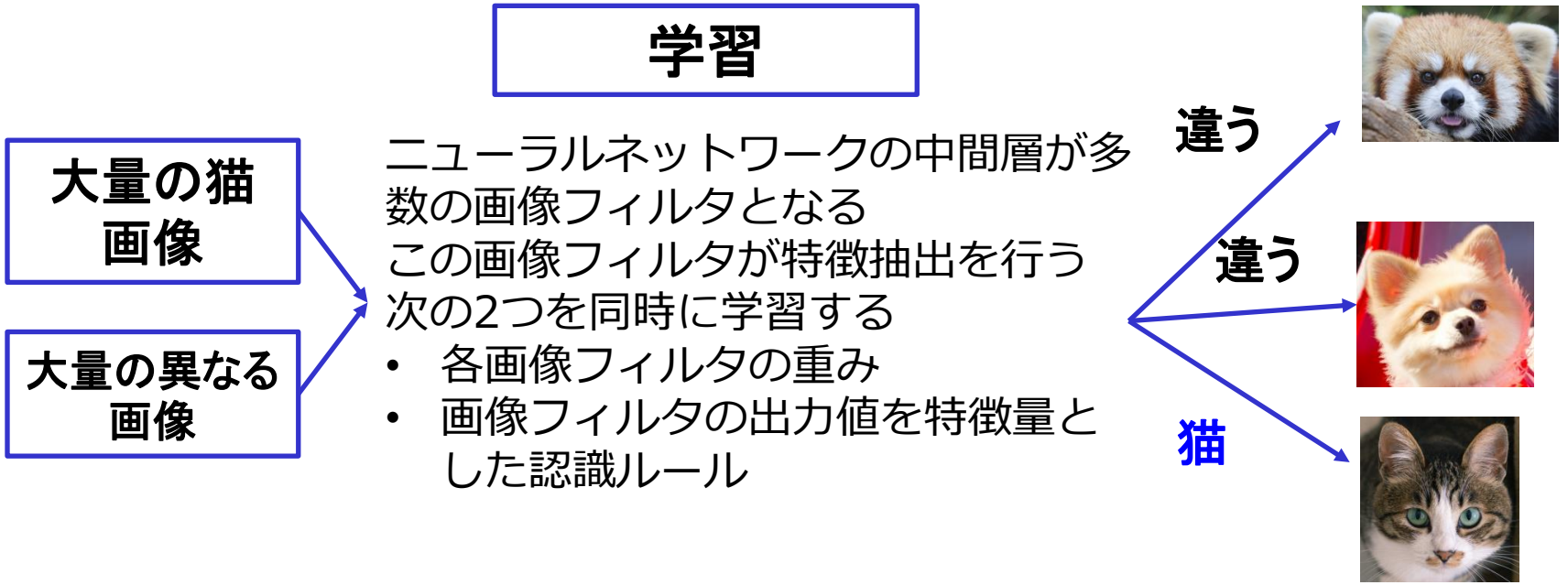


Sample face detection from, Paul Viola and Michael J. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features", IEEE CVPR, 2001.

Deep Learningの学習

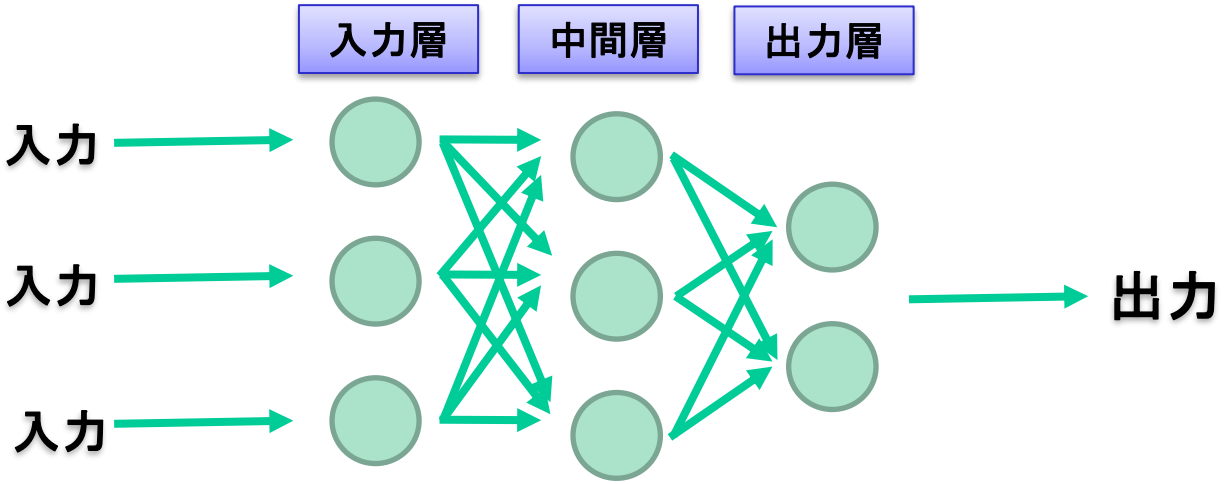
例）画像の物体認識

大量の正解画像と不正解画像（の画素値）をニューラルネットワークの入力とし、特徴と認識ルールを自動的に学習させる



従来のニューラルネットワーク

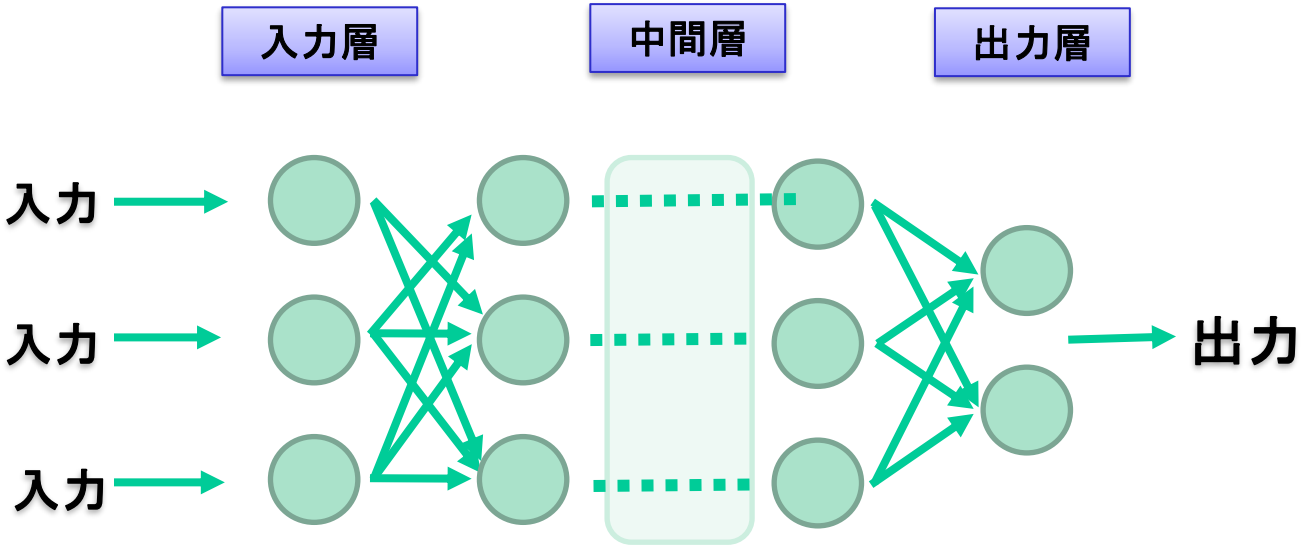
処理能力の限界から、入力層、中間層、出力層の3層構造が多く精度の問題があった



精度の問題

Deep Learning（深層学習）

中間層を多層化することで精度の問題を解決
ただし、GPUを使用してもやはり処理時間はそれなりにかかってしまう



中間層を多層化

従来手法と深層学習は何が違うのか

従来手法

人間が、どのような特徴量に着目するか、あらかじめ特徴を定義する

深層学習

コンピュータによって、特徴量を自動的に抽出する

ただし、学習のためのネットワークは人間が用意する

具体例紹介

きのこの山とたけのこの里の分類　（Interface 2017年8月号より）

従来の機械学習（SVM）

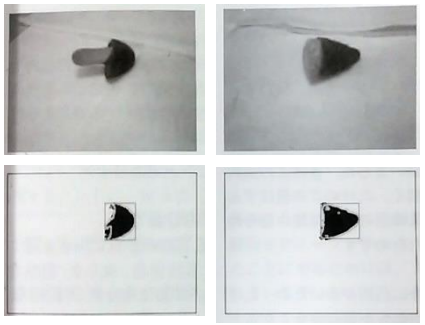
- 入力対象の何を抽出し、どうやって解析するかを人間が考える

1. きのこの山は柄の部分が細いので白い部分が多そうだ
2. きのこの山は傘の部分に凹凸が多そうだ

- 画像取得→2値化（チョコの部分抽出）

1. バウンディングボックス内の白と黒の面積比の計算
2. 黒い部分の頂点を検出し頂点数を求める

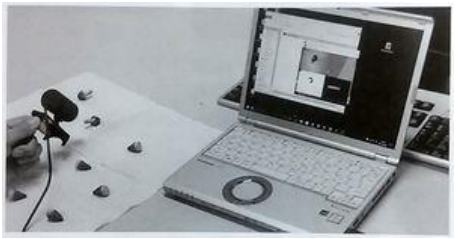
- 分類対象を1．と2．の数値データで表している



（Interface 2017年8月号 p.33）

Deep Learning

- 取得した画像を入力データとする
  - 画像の画素値が入力データ



（Interface 2017年8月号 p.33）

機械学習とは

Arthur Samuel (1959)

"Field of study that gives computers the ability to learn without being explicitly programmed“

明示的にプログラムしなくても学習する能力をコンピュータに与える研究分野

Photo by https://en.wikipedia.org/wiki/Arthur_Samuel

機械学習の例

k-近傍法（教師あり）
決定木（教師あり）
ランダムフォレスト（教師あり）
自己組織化マップ（教師なし）
サポートベクターマシン（教師あり）
ニューラルネットワーク（教師あり）
遺伝的アルゴリズム
ベイジアンネットワーク

人工知能とは

人工知能のFAQ（人工知能学会のサイトより抜粋）

Q. 人工知能とは何でしょうか？

- A. 知的な機械、特に、知的なコンピュータプログラムを作る科学と技術です。人の知能を理解するためにコンピュータを使うことと関係がありますが、自然界の生物が行っている知的手段だけに研究対象を限定することはありません。

Q. では、知能とは何でしょうか？

- A. 知能とは、実際の目標を達成する能力の計算的な部分です。人間、動物、そして機械には、種類や水準がさまざまな知能があります。

Q. AIは人の知能をまねようとしているのではないのですか？

- A. ときにはそうしますが、いつもというわけではありません。あるときは、機械に問題を解決させることについて、他人や自分自身がどうするかを調べます。一方、AIのほとんどの研究は、人間や動物について研究するよりも、知的に解決しなければならない問題そのものについて研究しています。AI研究者は、人間がやらないような方法や、人間ができるよりも多くの計算を伴う方法を用いることもできます。

AIは何の略ですか？

Artificial Intelligence :人工的な 知性

教師あり学習

- 人間（教師）があらかじめ用意したデータとパターンを学習し、ここから新しいデータに対して何らかの結果を出力する

教師なし学習

- 与えられたデータやパターンから、何らかの規則性をAI自らが発見し、見いだす

強化学習

- どのような行動を取れば最大限の利益が得られるかを学習する（囲碁やチェスなど）

ニューラルネットワークとは

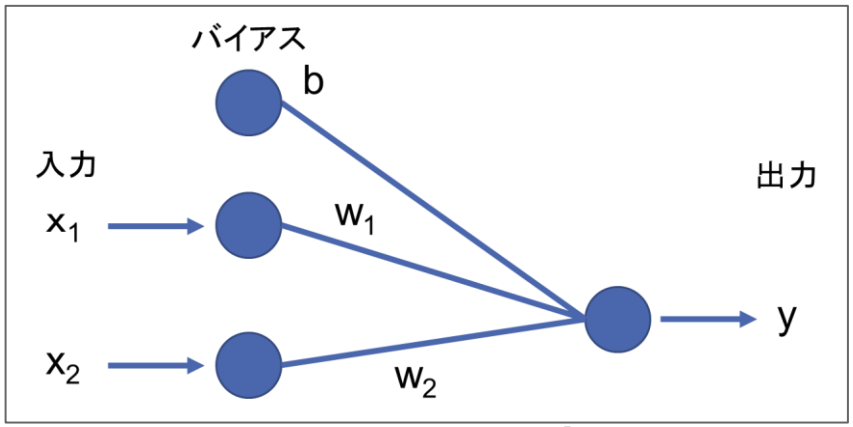
シナプスの結合によりネットワークを形成した人工ニューロン（ノード）が、学習によってシナプスの結合強度を変化させ、問題解決能力を持つようなモデル（Wikipediaより）

教師あり学習、教師なし学習の両方がある

単純パーセプトロン

入力層と出力層の2層のみの
ニューラルネットワーク

入力Xに対して重みとバイアスを用いた計算により得られた値と教師データYの誤差を計算し、誤差が小さくなるように重みとバイアスを更新して学習する

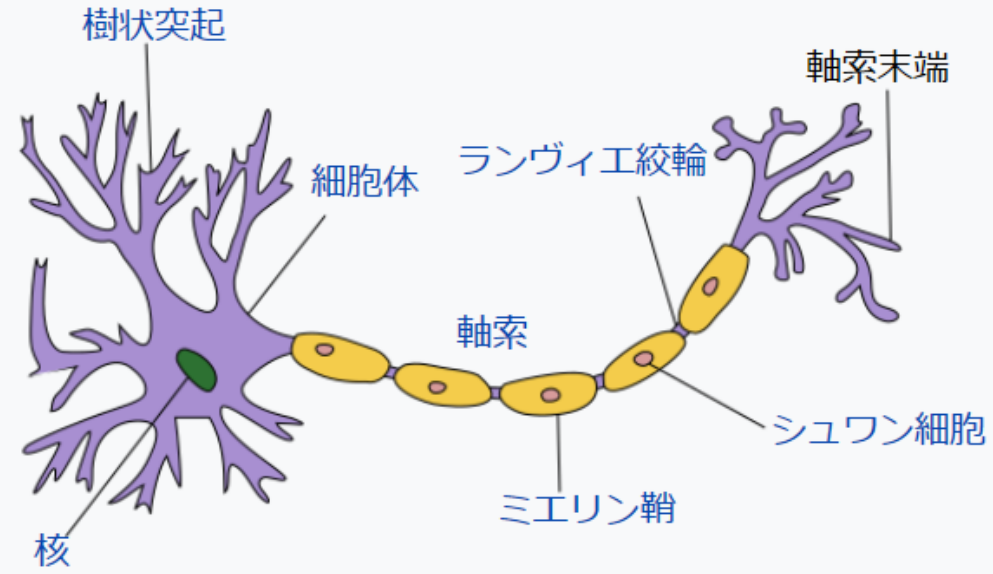


$$y = w_1 x_1 + w_2 x_2 + b$$

単純パーセプトロンの例

ニューロン

情報処理と情報伝達に特化した神経組織
電気信号を樹状突起から軸索末端に伝達する
昆虫で10万、人間にはおよそ1000億のニューロンが存在している

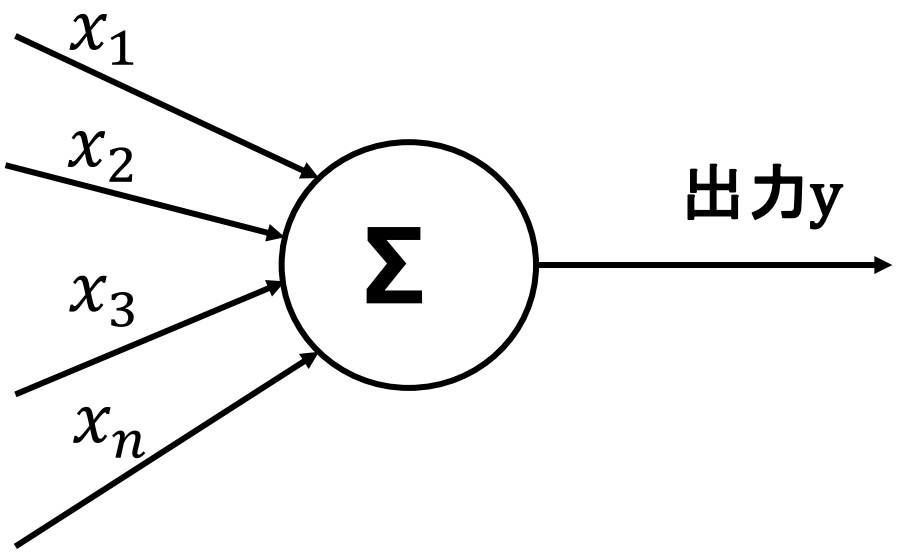


髄鞘をもつ神経細胞の構造図(Wikipedia 神経細胞 より)

形式ニューロン

1943年に発表された人工ニューロン。マカロックとピッツによって発表されたので、マカロック・ピッツモデルとも呼ばれる

ニューロンに対する入力を足し合わせ、ある閾値を超えると発火する

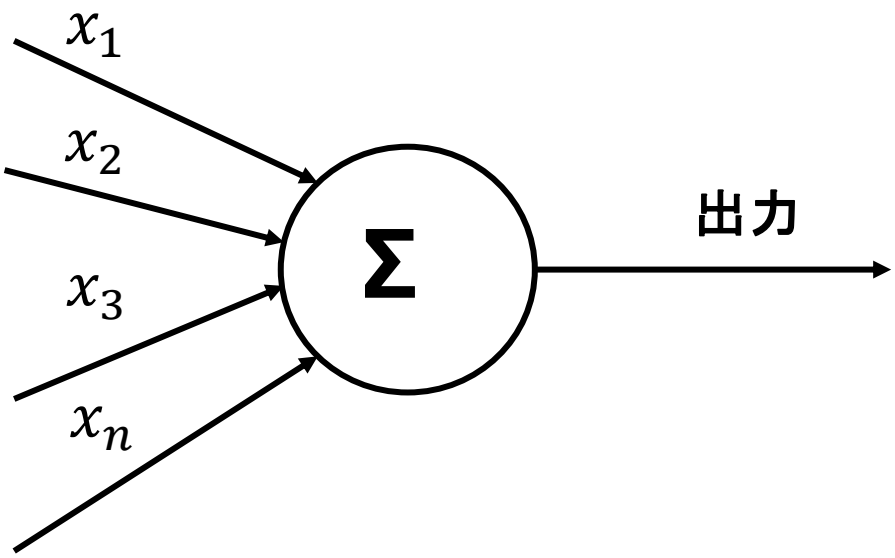


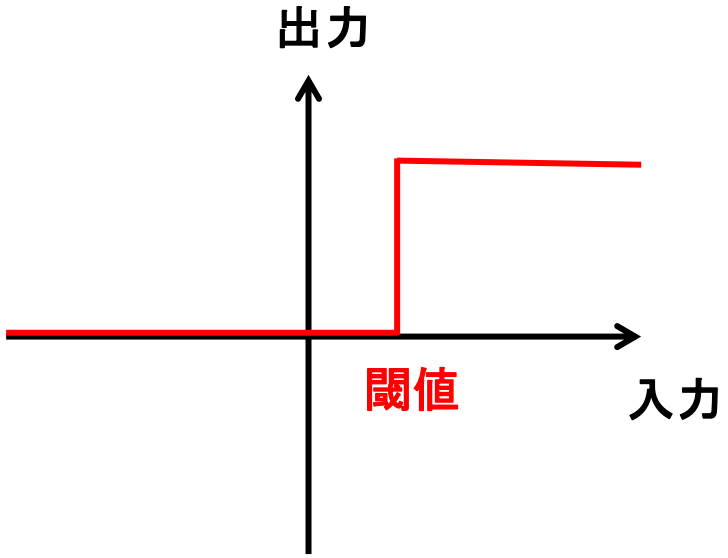
Warren S. McCulloch; Walter Pitts. “A logical calculus of the ideas immanent in nervous activity”. *The bulletin of mathematical biophysics* (Kluwer Academic Publishers) 5 (4): pp.115-133, 1943.

ニューロンの発火

設定された閾値を超えて、初めてニューロンは値を出力する

閾値はコップのようなもの。コップの容量内なら水を入れてもこぼれないが、容量を超えると水があふれる





AIの歴史

1960年代：最初期のAI。推論・探索の研究が盛んに行われたが、まったく実用的ではなく、急速にブームが収束

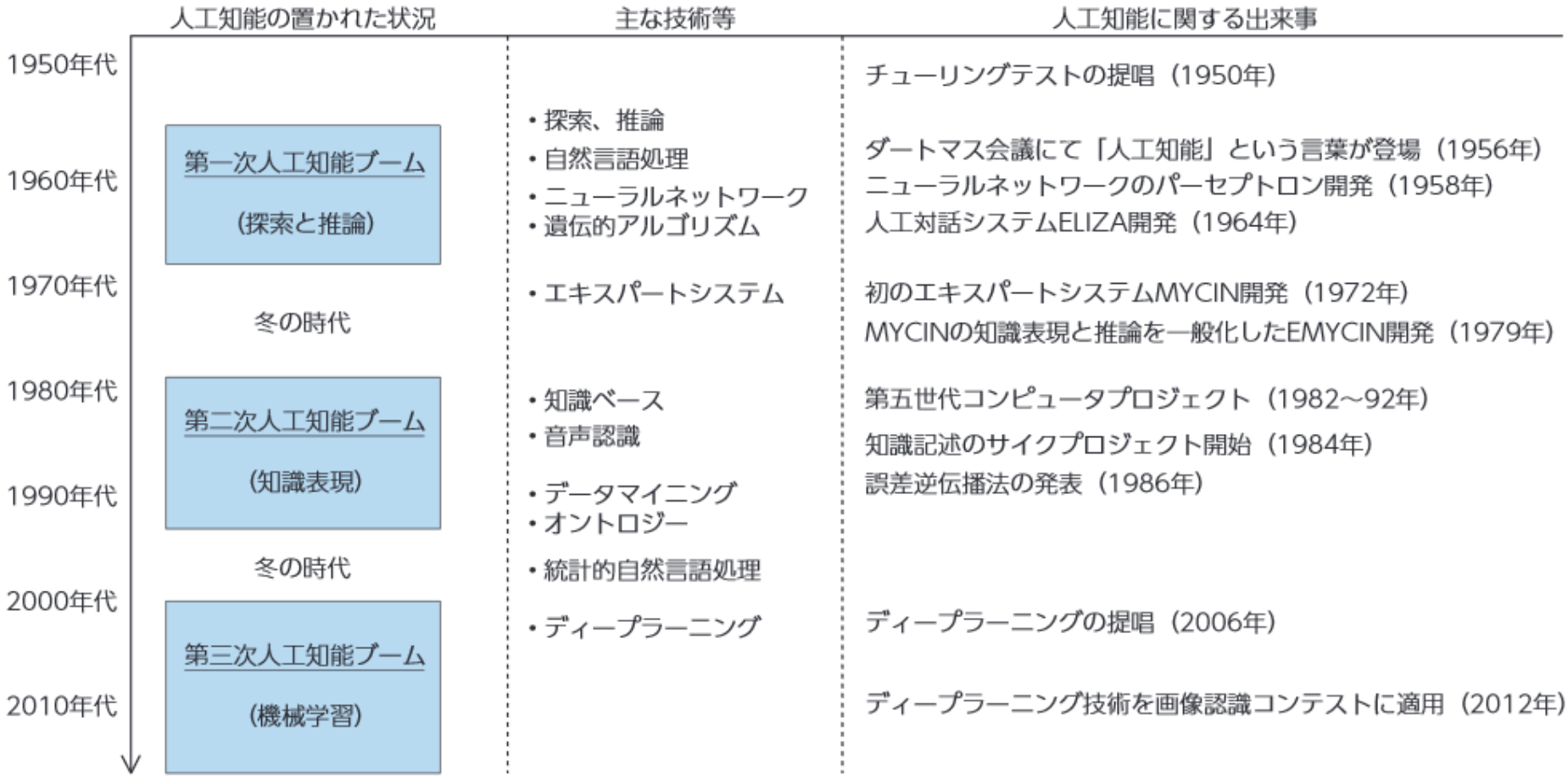
1980-90年代：エキスパートシステム、ファジー

専門分野の知識を取り込み、推論を行うことでその分野のエキスパートのように振る舞うシステム

If-Thenルール：条件を定義し、条件を満たせばその条件における判断を下す

2010年代：Deep Learning

図表4-2-1-5 人工知能（AI）の歴史



（出典）総務省「ICTの進化が雇用と働き方に及ぼす影響に関する調査研究」（平成28年）

ELIZA

最初期のAI。基本的にルールベース

ELIZA自身は精神科医という立場を取り、人間と対話する

今の技術からすると稚拙だが、当時はブームに。一部のユーザーはワイゼンバウムがその仕組みを説明しても納得せず、ELIZAの応答を真剣に受け止めた

日本でも類したものに、人工無脳、チャットボットがある

例）
人間：I have a terrible headache.
ELIZA：Maybe you should consult a doctor of medicine, I am psychiatrist.

画像をパターンで扱う

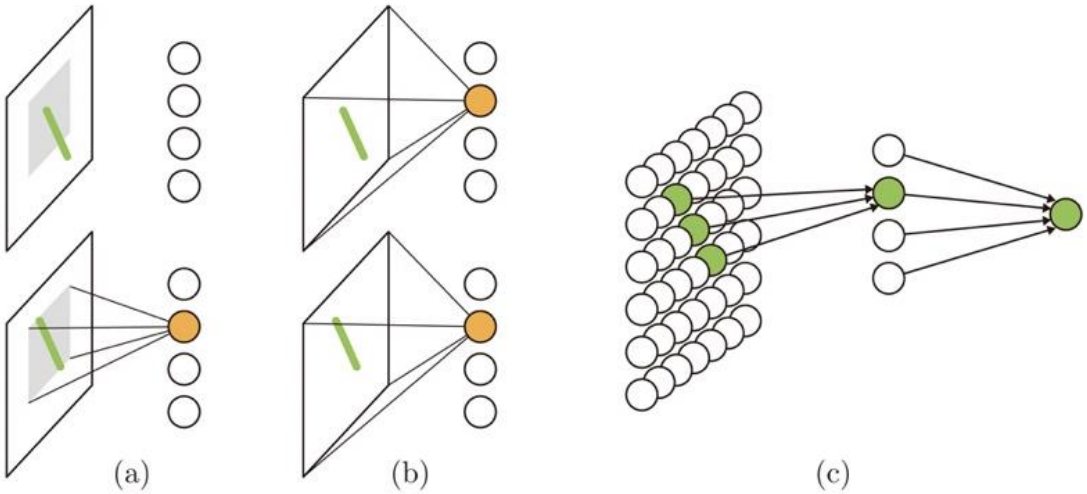
ヒューベル・ウィーゼルの階層仮設（1958年）

猫の視覚野にある線分を見せた時に反応する細胞があることを発見

さらに、単純型細胞(a)と複雑型細胞(b)の2種類に大別

単純型：パターン（線分）がある領域内に存在するときのみ反応

複雑型：パターン（線分）は領域からずれても反応



「これならわかる深層学習入門」瀧雅人, 講談社 より

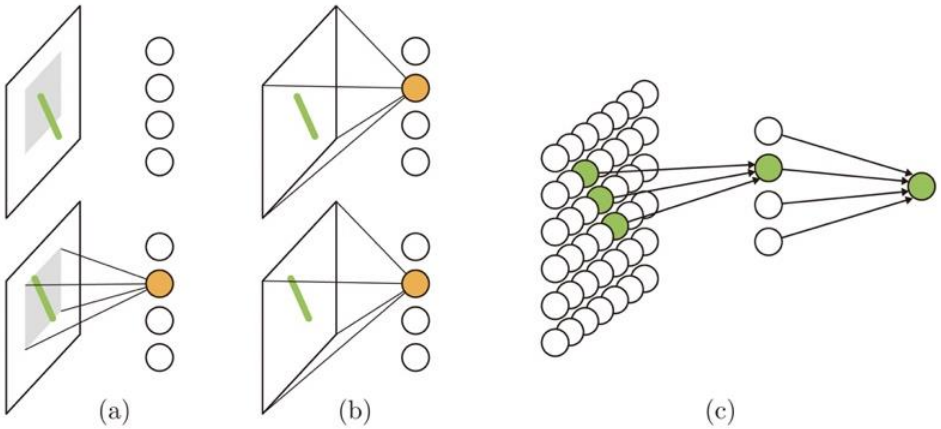
パターンの抽出

単純型細胞(a)が反応する部分を1つのニューロンに集める (c)

これを繰り返すことで、単純型細胞を複雑型細胞として束ねることができる

この考え方を元にして、1979年に福島(NHK)が単純型と複雑型を交互に重ねた多層ネットワーク、ネオコグニトロン*を提唱

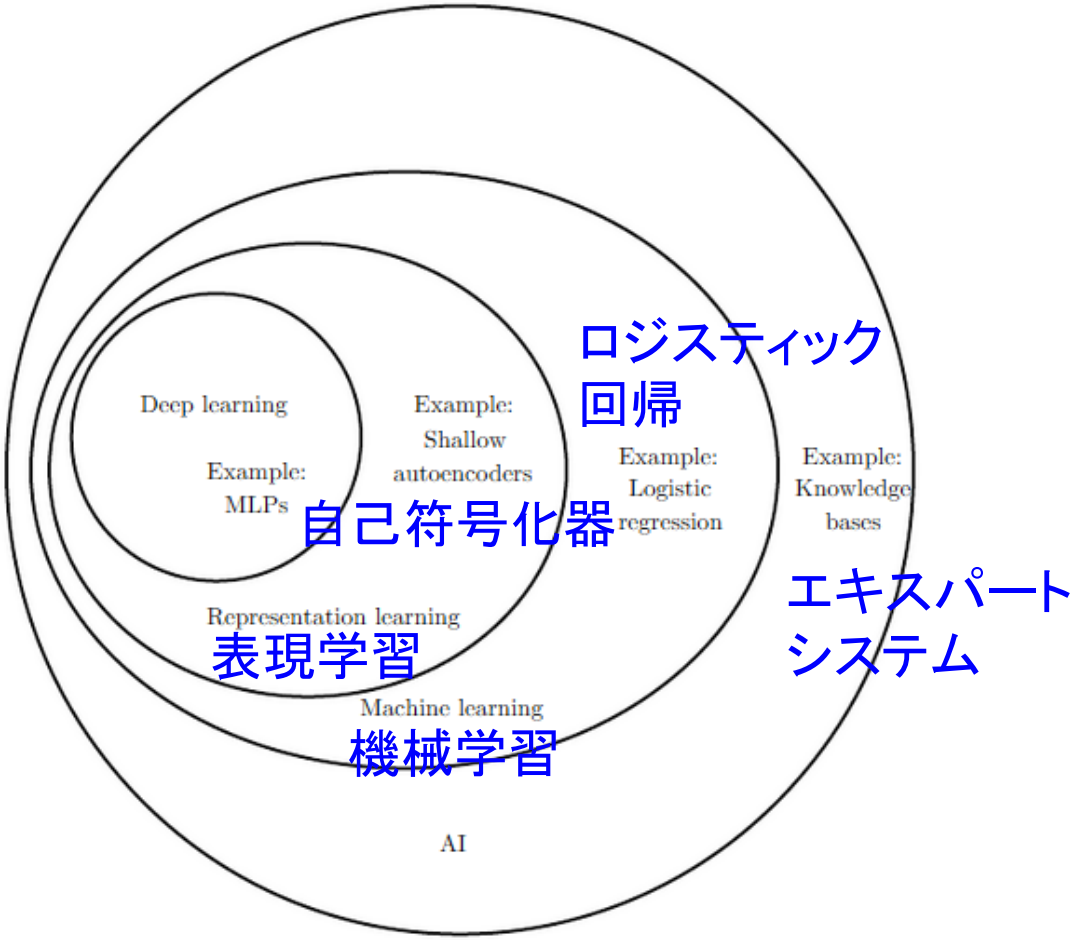
さらにこれを発展させ、ヤン・ルカンが1989年に5層、1998年に8層のCNNの構築・学習に成功している(LeNet) ……しかしこの後しばらくニューラルネットワークは冬の時代へ



*福島邦彦、位置ずれに影響されないパターン認識機構の神経回路のモデル --- ネオコグニトロン ---、1979年、電子通信学会論文誌A, vol. J62-A, no. 10, pp. 658-665

AIの歴史と概念（再掲）

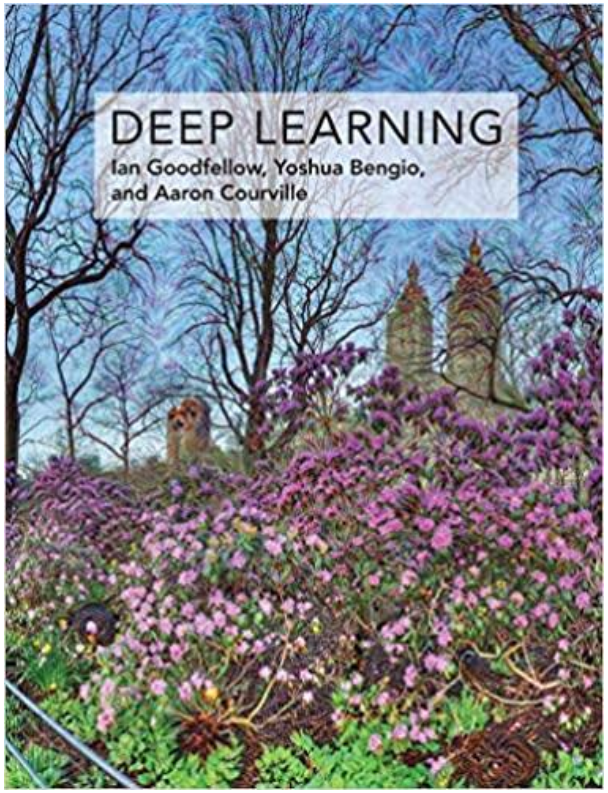
Goodfellow(Apple)
による概念図

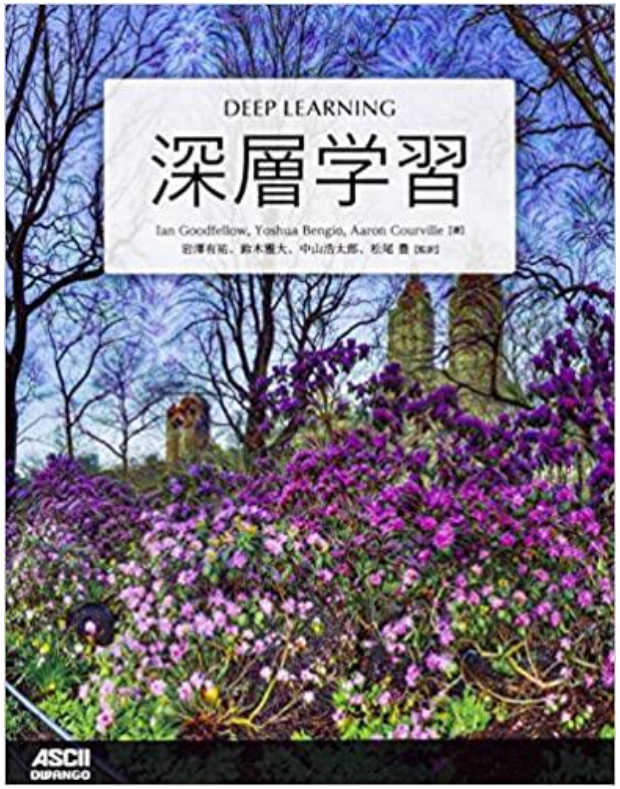


Ian Goodfellow, Yoshua Bengio and Aaron Courville, "Deep Learning", The MIT Press, 2016

書籍: DEEP LEARNIG

英語版800ページ、日本語版600ページというボリューム
英語版はWebでも公開されている： http://www.deeplearningbook.org/





ちょっとブレイク

今後の受講中サポートについて検討中です

リモート講師からのサポートがまだ決まっていません

Slack, MS Teams, Cisco Teams, Zoomブレイクアウトルームなど検討中ですが、皆さんがやりやすいものはありますか？

（MS Teamsは会社で使っているので使いやすい or 紛らわしいからちょっと……など）

流行したきっかけ: 1

G.H.ヒントン(Toronto Univ., Google)の2006年の論文

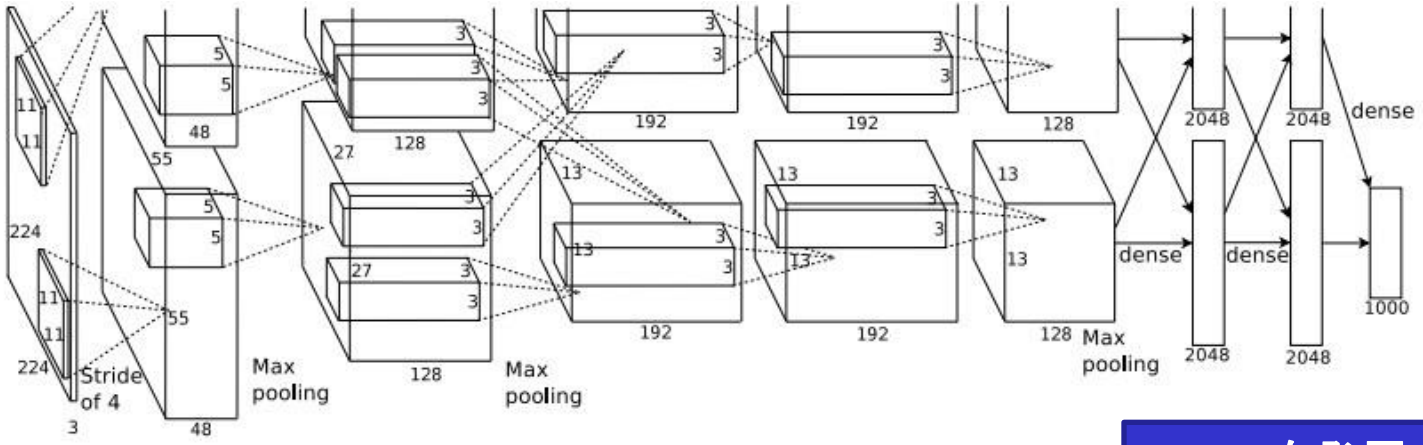
“Reducing the Dimensionality of Data with Neural Networks”, G. E. Hinton, R. R. Salakhutdinov, Science 28 Jul, Vol. 313, Issue 5786, pp. 50111-507,(2006)

要旨：小さな中心層を持つ多層ニューラルネットワークを訓練することによって高次元の入力ベクトルを再構成し、高次元のデータを低次元の符号に変換することができる。このような “Auto Encoder” ネットワークでは、グラディエント降下を使用してウェイトを微調整できるが、これは初期の重みが適切なソリューションに近い場合にのみ効果がある。

本論文では、データの次元性を低減するツールとして主成分分析よりもはるかに良く働く低次元符号をDeep Auto Encoder ネットワークが学習することを可能にする重みを初期化する効果的な方法を述べる。

流行したきっかけ: 2

物体認識率コンテストである IMAGENET Large Scale Visual Recognition Challenge 2012 (ILSVRC2012)において、G.H.ヒントンらのチームがDeep Learning手法で従来26%のエラー率を15%まで改善させた



LeNetを発展させたCNN

AlexNet: 5つの畳み込み層、3つの全結合層、65万ニューロン

ILSVRC2012 結果

1000万枚の画像で学習しテスト用に用意された15万枚画像のエラー率を競う
HintonらのSuperVisionのエラー率は他チームを圧倒
他チームは従来の特徴量抽出技術の延長
ディープラーニング手法はSuperVisionのみ

チーム名	エラー率 (%)	備考
SuperVision	15, 16	Hintonら
ISI	26, 27	牛久ら東大チーム
OXFORD_VGG	27	Simonyanら Oxford Univ.
University of Amsterdam	29	アムステルダム大学
XRCE/INRIA	33	イエーナ大学

http://www.image-net.org/challenges/LSVRC/2012/results.html

ILSVRCでは1000カテゴリの画像分類を行う
http://image-net.org/challenges/LSVRC/2012/browse-synsets

その後の発展: GoogLeNet, VGG16

ILSVRC2014で1位がGoogLeNet, 2位がVGG16

GoogLeNet: Googleによる22層モデル。Inceptionモデルとも呼ばれ、Googleが開発しているTensorflowでは、最新版のInception-v3がソースレベルで公開されている。（ネットワークは優秀だが複雑）

VGG16: Oxford Univ.によるAlexNetの拡張系。16層によるCNNで、設計がシンプルなので2位に終わったがよく使われるように。 19層使用するVGG19もある

人間の認識率を超える

ILSVRC2015では人間の画像認識エラー率 4%を超える、3.5%の結果をMicrosoft Research Asiaのチームが実現

ここでは、ResNetと呼ばれる152層のニューラルネットワークが使用された

前年のILSVRC2014で使用された主要ネットワーク VGGは19層、GoogleNetは22層なので、前年よりも7倍近い規模

多層にすればある程度は性能が上がるが、多くしすぎると逆に誤差が増加したり、逆誤差伝搬の問題もあり、様々な工夫がなされている

Deep Residual LearningMSRA @ ILSVRC & COCO 2015 competitions
http://kaiminghe.com/ilsvrc15/ilsvrc2015_deep_residual_learning_kaiminghe.pdf

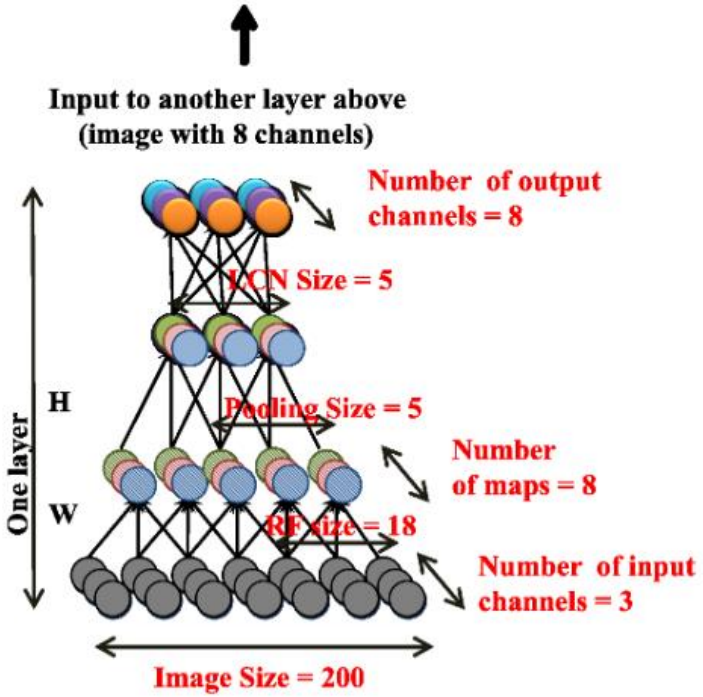
流行したきっかけ: 3

Googleによる猫の「認識」(2012年)

YouTubeにアップロードされている動画から、ランダム抽出した200x200ピクセルサイズの画像を1000万枚用意し、Deep Learningに入力した

3%前後の画像に人間の顔が含まれており、猫が含まれる画像も多数

Googleクラウドの1,000台を使用して3日間計算した

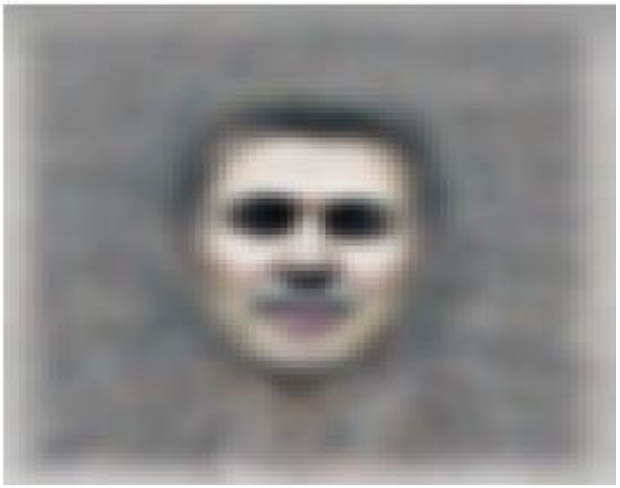


9層のネットワーク構成

人間の認識

Googleの研究では、人間の顔、背中、猫に強く反応するニューロンの選択に成功

ニューロンが反応した顔写真群



ニューロンによって生成された人間の顔の特徴

猫の認識

ニューロンが反応した猫写真群

ニューロンによって生成された猫の顔の特徴

本物の猫

https://googleblog.blogspot.com/2012/06/using-large-scale-brain-simulations-for.html

人工知能の世間的な分類

レベル1：単純な制御プログラム
- マーケティング的に「人工知能」を名乗っているだけ

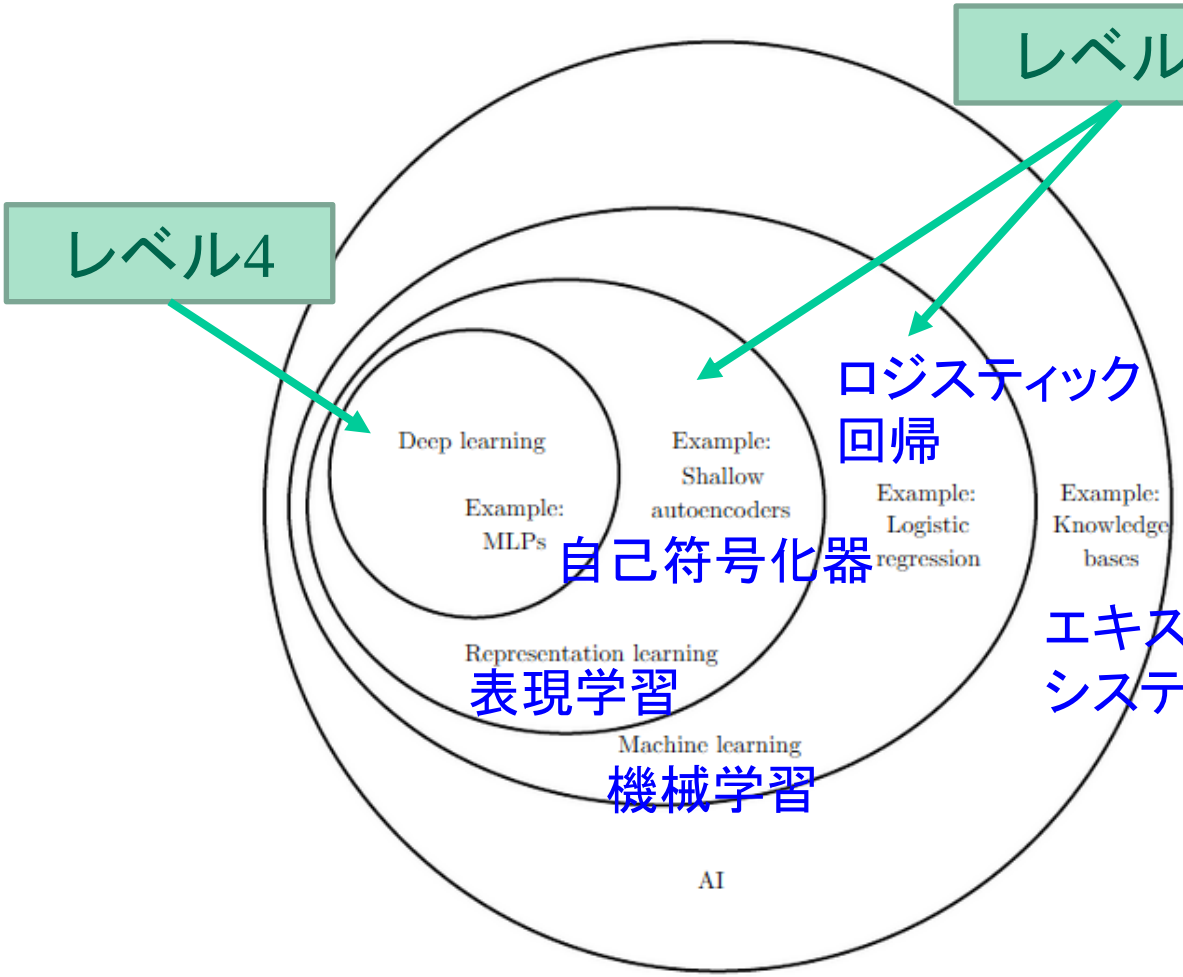
レベル2：古典的な人工知能
- 推論、探索、知識ベースなどにより、組み合わせが極端に多い入力と出力を関係付ける方法を定義する

レベル3：機械学習を取り入れた人工知能
- サンプルやデータをもとに入力と出力の関係であるルールや知識を自ら学習する
- 入力は、目的に応じて入力対象の特徴をあらわすもの（特徴量）である必要がある

レベル4：ディープラーニングを取り入れた人工知能
- 特徴量自体を学習する機械学習

（松尾豊、「人工知能は人間を超えるか」より）

Goodfellowの図と対応



Ian Goodfellow, Yoshua Bengio and Aaron Courville, "Deep Learning", The MIT Press, 2016

弱い人工知能（AI）と強い人工知能（AI）

ジョン・サール（アメリカの哲学者）の造語

- 弱い人工知能
  - 人間の全認知能力を必要としない程度の問題解決や推論を行うソフトウェアの実装や研究
  - チェスや囲碁など限定された範囲の問題について、一見知的に見える問題解決ができるもの
- 強い人工知能
  - 人間の知能に迫るようになるか、人間の仕事をこなせるようになるか、幅広い知識と何らかの自意識を持つもの
- 汎用人工知能（≒強い人工知能）AGI: Artificial General Intelligence
  - 人間レベルの知能を実現するもの
  - 限定された問題を解決する特定型人工知能ではなく、一般的な知能を実現するもの

人工知能の利用動向

- 人工知能の効果を最大化するためには、質の高い学習用データを基に付加価値を生み出す学習済みモデルを生成することが重要
- 学習済みモデルを初期状態とし再利用することで、比較的少数の学習データから優れた性能を持つ派生データを得ることができる
- 画像認識や音声認識の領域ではすでに実用性の高い技術として応用が進められている
  - 自動走行における車外走行環境認識、医用画像からの疾病等の診断支援など
  - 家庭や自動車内での音声対話や音声アシスタント、工場での異常音検知など

（AI白書2017より）

今後の展望

海外では検索サービスやSNSなどのインターネット空間での活動から得られるデータに対して適用を進めている

海外企業がすでに圧倒的なシェアを持つインターネット空間を中心とした人工知能利用に、今後対抗することは容易ではない

先行する企業はAIの機能を組み込んだ機械やロボットを普及させることで実空間における消費者との接点も押さえつつある

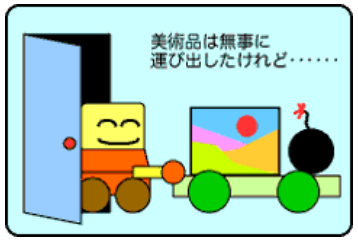
健康情報、自動車の走行データ、工場の稼働データなど、個人や企業の実世界における活動から得られる実空間データへの適用は今後の競争課題

（AI白書2017より）

AIの課題(フレーム問題)

1969年にマッカーシーとヘイズが指摘した人工知能研究の最大の難問であり、今からしようとしていることに関係のある事柄だけを選び出すことが、実は非常に難しいという問題

AI搭載のロボットは、人間の代わりに危険な作業を行う。爆弾が仕掛けられている部屋から、美術品を取り出してくる作業において、美術品の入った台車を押して作業をしていたが、爆弾が台車に仕掛けられていたので爆発に巻き込まれてしまった。そこで、改良型ロボットが作られ、再度部屋に向かった。しかし、美術品を運び出すには台車を動かせばよいと思いついたあと、台車を動かしたときの影響を、天井は落ちてこないか、部屋の壁の色は変わらないか、電気は消えないか……などと考えているうちに爆弾が作動してしまった



J.McCarthy and P.J.HayesSome philosophical problems from the standpoint of artificial intelligence, Machine Intelligence, vol.4, pp.463-502 (1969) 画像はAI学会Webサイトより

AIの課題(シンボルグラウンディング問題)

記号システム内のシンボルがどのようにして実世界の意味と結びつけられるかという問題。記号接地問題とも呼ぶ。ハルナッドが提唱

⟷ **猫**

人間は、猫の画像と、猫という概念・意味を理解している

AIがこれを理解するには、目が2つ、耳が頭の上に……などの猫の特徴を細かく画像で理解するしかない。しかし、りんごと青りんごや、シマウマと馬などがそれぞれ近い種類であることは、単なる画像認識ではわからない

Harnad, S.:"The Symbol Grounding Problem", Physica D 42: pp.335-346, 1990.

AIの実例：自動運転

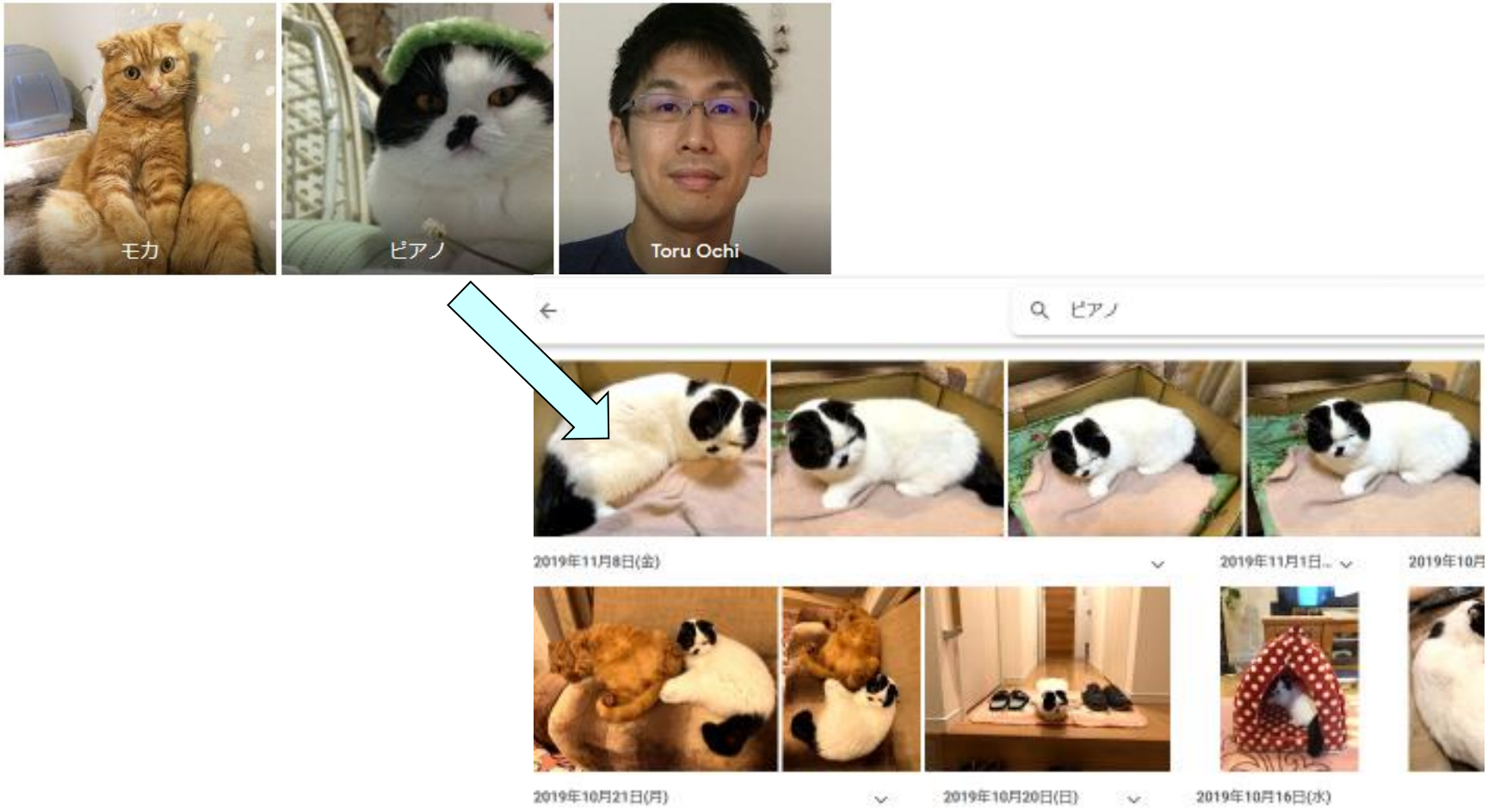
国土交通省による自動運転のレベル

レベル1：運転支援　自動ブレーキ、車線検知など
レベル2：特定条件下での自動運転機能　レベル1の組み合わせなど
レベル3：条件付自動運転　システムが全ての運転タスクを実施するが、システムの介入要求等に対してドライバーが適切に対応することが必要
レベル4：特定条件下における完全自動運転　特定条件下においてシステムが全ての運転タスクを実施
レベル5：システムによる完全自動運転

現在、米テスラがレベル3の自動運転を提供。ただし事故も報告されている

実例：Google Photo

人物や動物を登録すると、自動で抽出してくれる



パンの自動選別

株式会社ブレイン によるトレイ上のパン自動選別システムBakeryScan
手法としては従来型の画像処理で、機械学習ではあるがDeep Learningではない

きゅうりのAI

実家のきゅうり農家を継いだ元ソフトウェアエンジニアが自作

きゅうりの等級選別をなるべく自動で行いたい　→　AIによる自作開発

Webカメラ、Raspberry Pi、Tensorflowで環境構築、きゅうりを運ぶベルトコンベアも自作

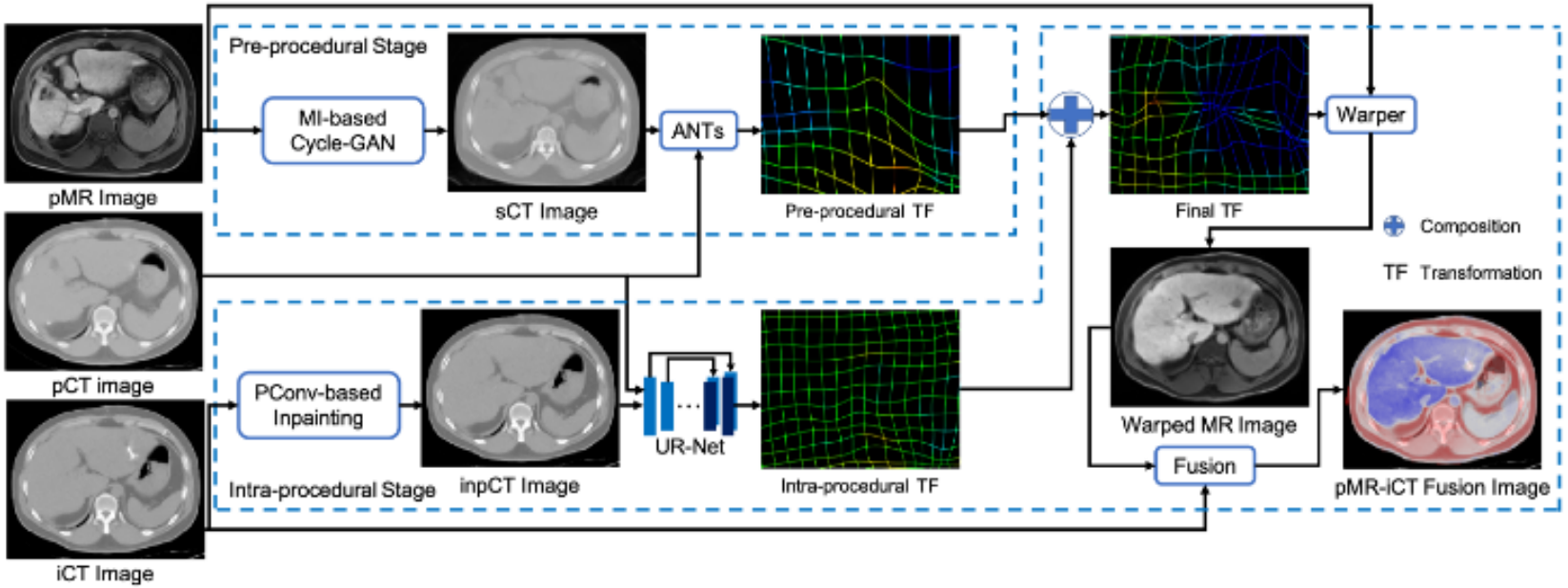
教師データ用画像2万8000枚、8000枚のテスト画像での精度は約80%

完全選別ではなく、人間のサポートとして使用すると、仕分けスピードが約40%向上

元組み込みエンジニアの農家が挑む「きゅうり選別AI」 試作機3台、2年間の軌跡
https://www.itmedia.co.jp/enterprise/articles/1803/12/news035.html

医療診断：肝臓腫瘍候補の検出

MR/CT画像を合成した画像から、肝臓腫瘍を検出
Cycle-GANをベースにしたニューラルネットワークを構築



Wei, Dongming, et al. "Synthesis and Inpainting-Based MR-CT Registration for Image-Guided Thermal Ablation of Liver Tumors." International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer, Cham, 2019.

構築したネットワーク

Layer Name	Filter Size	Number of Filter	Stride	Padding	Nonlinearity
Concat_1(Moving, Fixed)					
Conv3D_1	3 × 3 × 3	2	1	Y	LeakyReLU(0.2)
Conv3D_2	3 × 3 × 3	2	1	Y	LeakyReLU(0.2)
Conv3D_3	3 × 3 × 3	16	2	Y	LeakyReLU(0.2)
Conv3D_4	3 × 3 × 3	32	1	Y	LeakyReLU(0.2)
Conv3D_5	3 × 3 × 3	32	2	Y	LeakyReLU(0.2)
Conv3D_6	3 × 3 × 3	32	1	Y	LeakyReLU(0.2)
Conv3D_7	3 × 3 × 3	32	2	Y	LeakyReLU(0.2)
Conv3D_8	3 × 3 × 3	32	1	Y	LeakyReLU(0.2)
Conv3D_9	3 × 3 × 3	32	2	Y	LeakyReLU(0.2)
Conv3D_10	3 × 3 × 3	32	1	Y	LeakyReLU(0.2)
Conv3D_11	3 × 3 × 3	32	1	Y	LeakyReLU(0.2)
Upsampling3D_1		32	2		
Concat_2(Conv3D_8)		32+32			
Conv3D_12	3 × 3 × 3	32	1	Y	LeakyReLU(0.2)
Upsampling3D_2		32	2		
Concat_3(Conv3D_6)		32+32			
Conv3D_13	3 × 3 × 3	32	1	Y	LeakyReLU(0.2)
Upsampling3D_3		32	2		
Concat_4(Conv3D_4)		32+32			
Conv3D_14	3 × 3 × 3	32	1	Y	LeakyReLU(0.2)
Conv3D_15	3 × 3 × 3	16	1	Y	LeakyReLU(0.2)
Upsampling3D_4		16	2		
Concat_4(Conv3D_2)		2+16			
Conv3D_16	3 × 3 × 3	6	1	Y	LeakyReLU(0.2)
Conv3D_17	3 × 3 × 3	3	1	Y	Linear

診断画像からの病気予測(1)

陽電子放出断層撮影(PET)画像の結果から、アルツハイマーを予測・分類
3層の畳み込みを含む5層ネットワークを使用
提案手法(PETNet)が最も高い予測を得ている

Method	Accuracy (2-Classes)	Accuracy (3-Classes)
PETNet	93%	**77%**
PETNet (empty graph)	88%	55%
PETNet (random graph)	86%	64%
ResNet (without pre-training)	83%	58%
ResNet (with pre-training)	**95%**	65%
XGBoost	88%	62%
SVM	69%	57%

Guo, Jiaming, et al. "Predicting Alzheimer's Disease by Hierarchical Graph Convolution from Positron Emission Tomography Imaging." arXiv preprint arXiv:1910.00185 (2019).

診断画像からの病気予測(2)

富士フイルム　画像診断系プラットフォームREiLI

http://reili.fujifilm.com/ja/

書籍紹介：AI概論、動向

60分でわかる！機械学習＆ディープラーニング超入門：機械学習研究会
技術評論社、2017　本当に超入門。コンパクトにとりあえず概要をつかむ本

人工知能は人間を超えるか ディープラーニングの先にあるもの：松尾豊、KADOKAWA/中経出版、2015
日本でもトップレベルのAI研究者による紹介本。内容は若干古いが、非常にわかりやすく、かつコンパクトにまとめられている

AI白書 2020：独立行政法人情報処理推進機構 AI白書編集委員会 編、KADOKAWA、2020
国内・国外のAI動向をまとめたもの。松尾先生も委員会に加わっている

書籍紹介：事例紹介

業界別！AI活用地図：本橋 洋介，翔泳社

タイトルの通り、AIの活用事例を集めて解説したもの

2019年11月発売なので、かなり最近の事例が多い

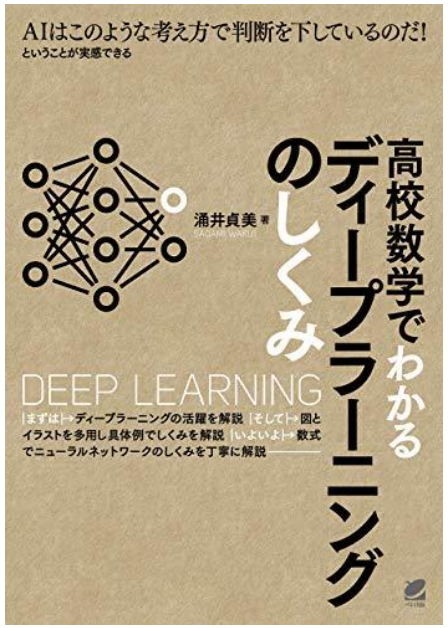
ただし、ここで公表されている事例が本当にAIなのかどうかは不明

書籍紹介：AIに必要な数学を学ぶ

高校数学でわかるディープラーニングのしくみ：涌井貞美、ベレ出版

Deep Learningに必要な数学について、高校数学ベースで説明した良書

著者の涌井氏は、この他「Excelでわかるディープラーニング超入門」など、平易でわかりやすい関連本を多く手がけている



この講座で今後使用するライブラリ

Tensorflow, Keras：機械学習ライブラリ
Matplotlib：Python標準のプロット用ライブラリ
Numpy：リスト、行列操作用ライブラリ
Pandas: データ解析用ライブラリ
OpenCV：画像・映像を扱うには欠かせない画像処理ライブラリ

Deep Learningではとにかくデータ構造を入れ替えたり操作したりが多いので、Numpyと可視化のためにMatplotlibが欠かせない。また、データ解析にはPandasも使用される

AIにライブラリは必要？

必須ではない
自分で組めばもちろん必要なし
ライブラリを使うと正直ブラックボックスも多いが、様々なネットワークが簡単に組める
主流はTensorflow/KerasとPythonによる開発
最近人気があるのがPyTorchとPython
データ操作は、Pythonを使用した方が圧倒的に楽

自作本として、例えば「ニューラルネットワーク自作入門」
著：Tariq Rashid, 監訳：新納浩幸

PC上でDeep Learningを実行するなら

CPUでは演算時間が膨大になってしまうので、NVIDIA GPUの使用がほぼ必須

ただしGPUのライブラリであるCUDA, cuDNNとPython, ライブラリのTensorflowのバージョンを合わせないといけない

※動作できるバージョンが決まっている

Python ver, 3.7, Tensorflow ver.2.1.x, CUDA 10.1.418.x, cuDNN 7.6

(最新バージョンはPython 3.8, CUDA11, cuDNN 8.0)

ローカルPCとほぼ同一ではないが、Google Colaboratoryでバージョン合わせに悩むことなく簡単に実行できる

GPU

巨大なデータをGPUで扱う場合、GPUのメモリ容量に注意
最新版 RTX 3080iは12G予定
現行品では RTX 2080 8G（8万円前後）など
nVIDIA GPUは複数使用も可能なので複数使用でメモリは拡張可能
データセンター向けのTESLAシリーズは16Gなど

Google Colaboratory

PythonをWebブラウザ上で実行し、画像を組み合わせて表示できる環境
Jupyter NotebookをGoogle Compute Engine上で実装したもの。完全無料

https://colab.research.google.com/

Googleクラウド上のGPUや、Googleが独自開発したTensorFlow用演算ハードウェアのTPUを使用して高速な演算が可能

Webブラウザからアクセスできればok

ただし、90分＆12時間ルールがある

データはGoogle Drive上に保存するか、ダウンロードする

Google ColabのGPU速度は、おおよそGTX 1060程度（2017年3万程度）

クラウドサービスのDeep Learning API

例) Amazon Rekognition
https://aws.amazon.com/jp/rekognition/

JavaScriptからAPIを呼び出す
有料だが、画像1000枚の判定につき1ドル、など安価
実際に使用するには、EC2, S3などAWSの周辺知識が必要

第2章

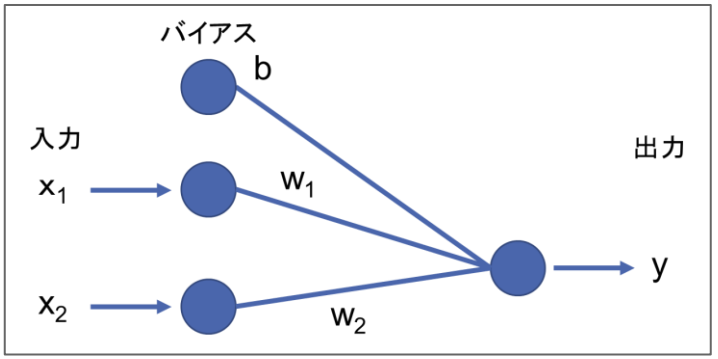
機械学習やアルゴリズムへの理解

本節の内容

e-learnin教材では、ニューラルネットワークの重みパラメータはあらかじめあたえていたが、ここではそれを自動的に求めるための基礎となる概念、手法（損失関数と勾配降下法）について説明する

（人工）ニューラルネットワーク

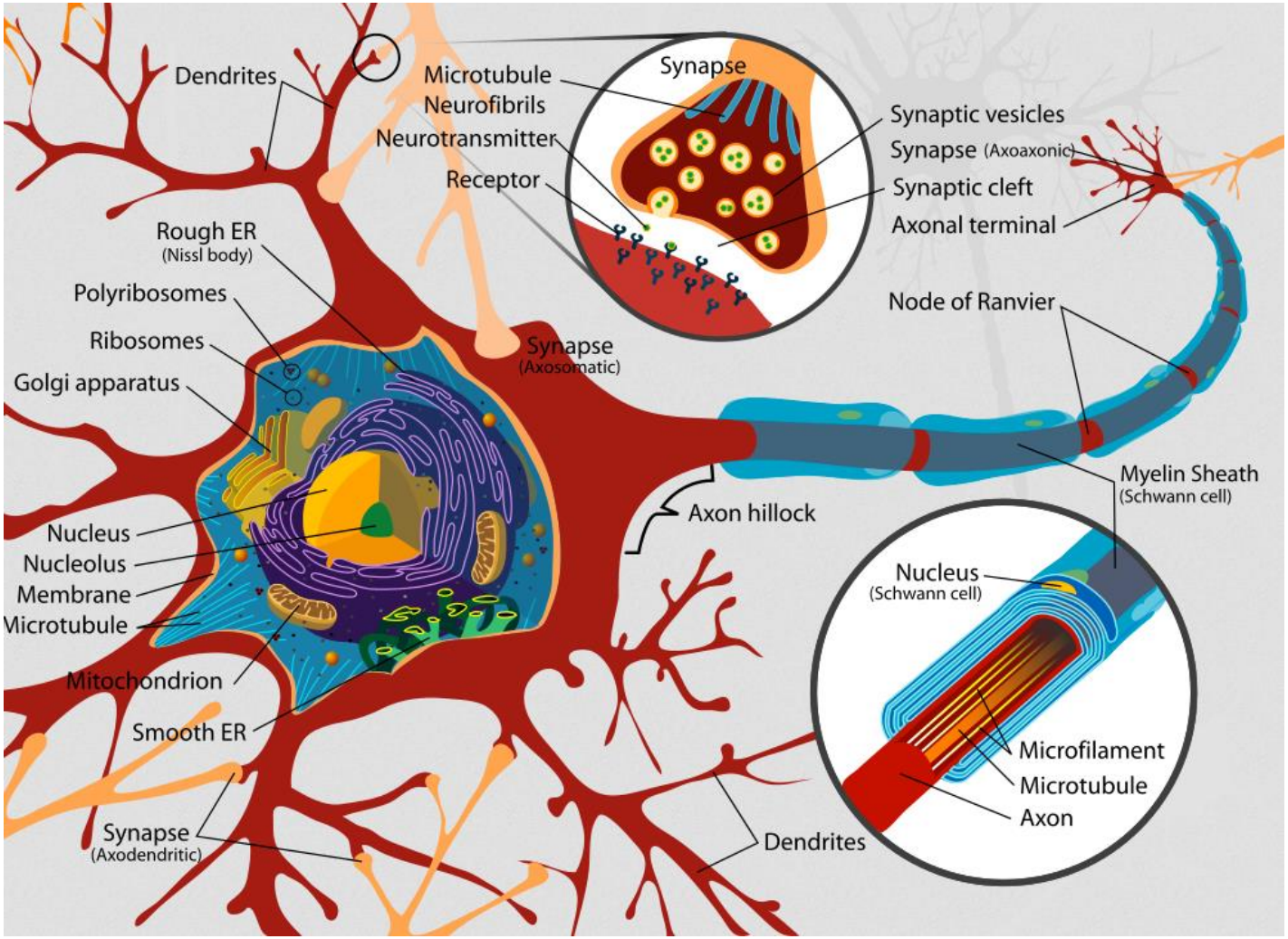
- マカロック=ピッツのモデルを源流とする人工ニューロン（という関数）を次々と組み合わせて構成された関数のこと。複数のニューロン（神経細胞）から、あるニューロンへの信号の入力を重みを持った線形和と活性化関数（後述）により数式化する。
- もっとも原始的な例（右図）
  - 入力層と出力層の2層（数え方によっては1層）のみのニューラルネットワーク
  - 右側の丸の内部では左側からの3つの重み付き和に活性化関数と呼ばれるある関数を施した結果が出力される



$$y = f(w_1 x_1 + w_2 x_2 + b)$$

- ネットワークの重みとバイアスは当初は手で（あるいは勘で）決められたが、より複雑なネットワークは人手では扱うことができず、コンピュータプログラムにより自動的に調整する必要がある。この調整過程が学習と呼ばれる

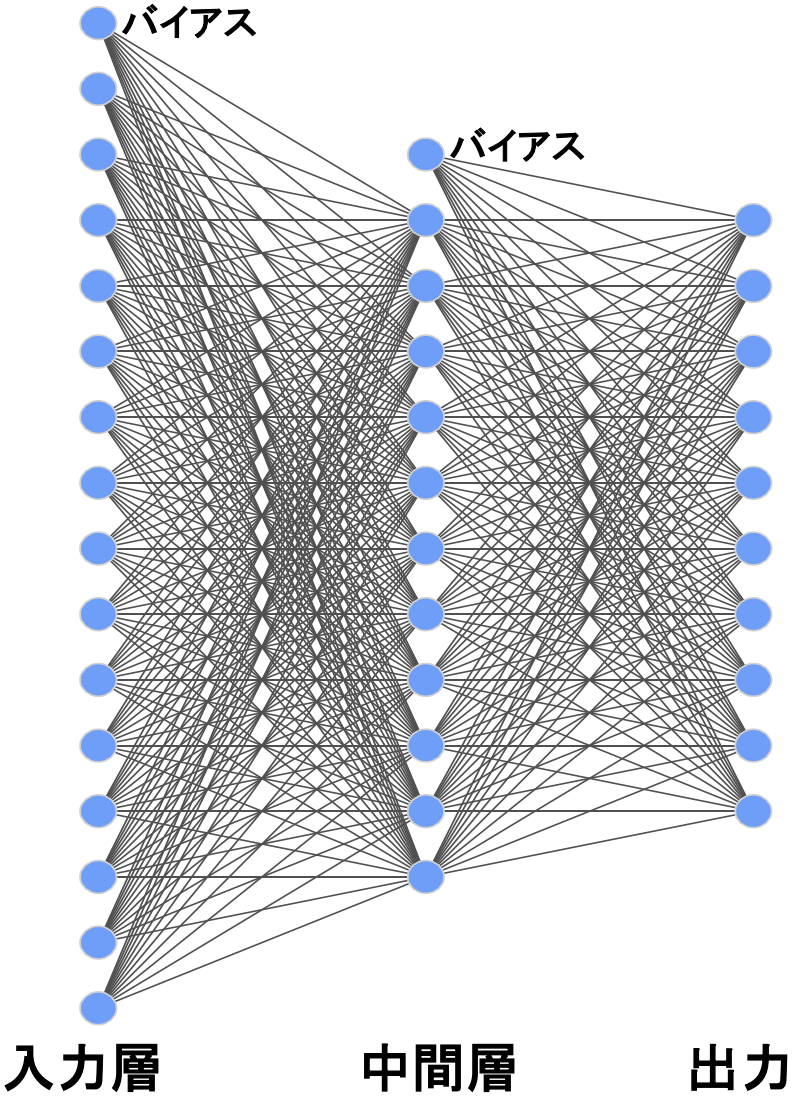
ニューロン



脳科学辞典「シナプス」より

Deep Learning（深層学習）とは

- 多層のニューラルネットワークによる機械学習手法
- 層の数に特に厳密な定義は無い
  - 入力層と出力層以外に層があればDeepと呼ぶこともある
- 層の数が増えると学習が難しく、特殊例を除きあまり実用化されていなかった
- 入力と出力層の間にある層を**中間層**、あるいは**隠れ層**と呼ぶ

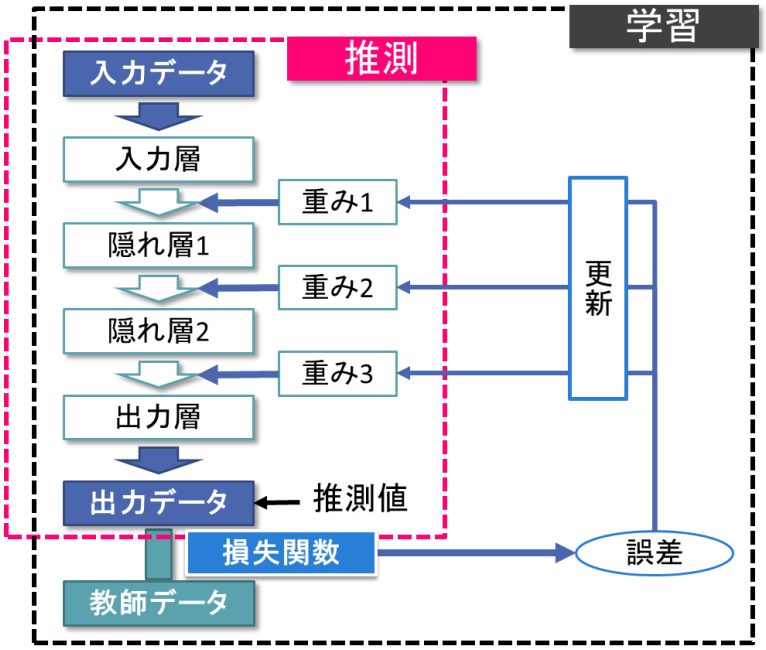


ディープラーニングの学習過程概要

入力をディープニューラルネットワークで計算し、出力を得る

出力を推測値として、教師データとの誤差を用いて各層の重みを更新する

推測と重みの更新を繰り返し、誤差が少なくなるように重みを適正な値に収束させる

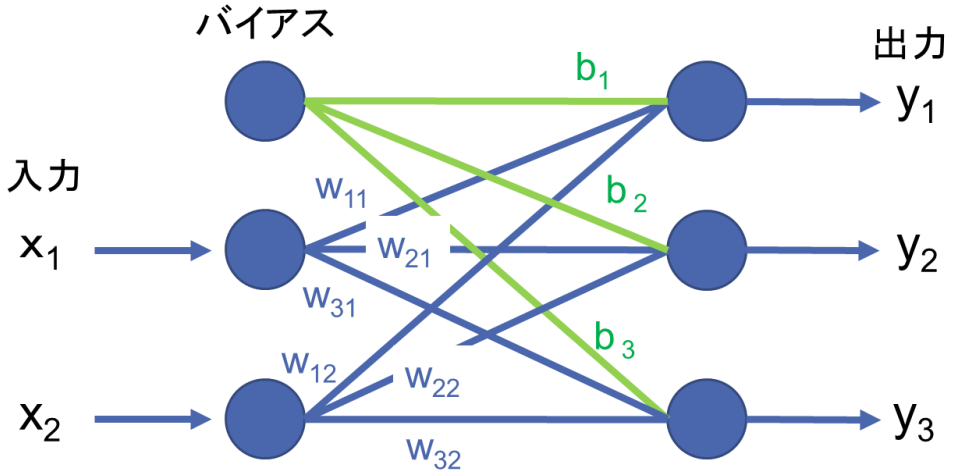


※隠れ層 = 中間層

ニューラルネットワーク内の計算

下の図は最も基本的な全結合ニューラルネットワーク
（第1層は2ユニット、第2層は3ユニット）

重み和の計算は、行列とベクトルの積として表すことができる



$$y_k = w_{k1}x_1 + w_{k2}x_2 + b_k \quad (k = 1, 2, 3)$$

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

$$= \begin{pmatrix} w_{11} & w_{12} & b_1 \\ w_{21} & w_{22} & b_2 \\ w_{31} & w_{32} & b_3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ 1 \end{pmatrix}$$

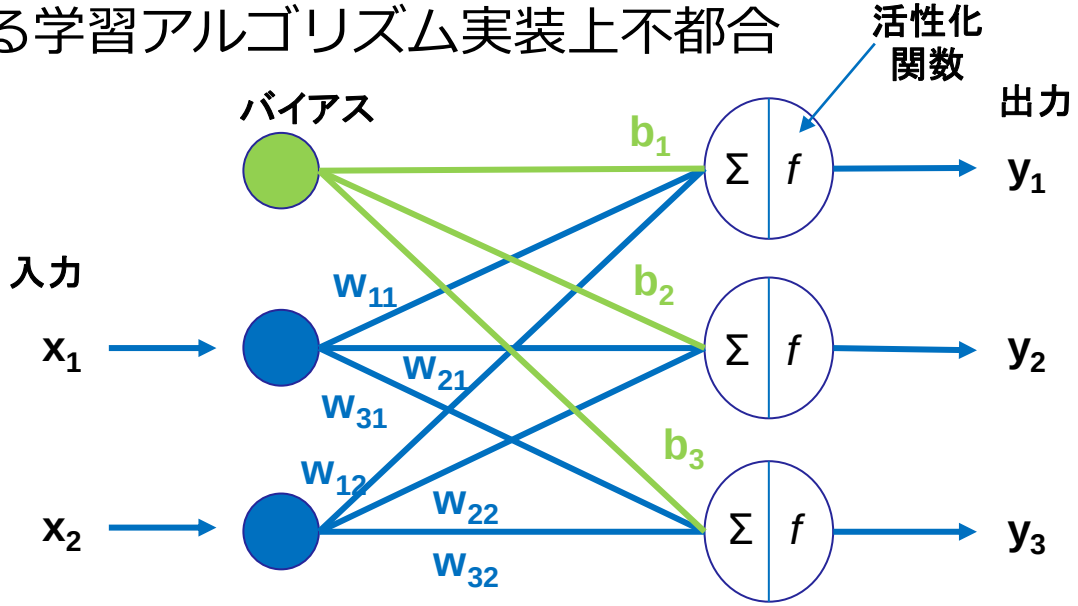
※ 実際は、次に述べる活性化関数をさらに施したものを出力とする

活性化関数（Activation function）

各ニューロン（図式内の丸印）の計算結果を活性化関数で変換して次の層へ送る

ニューロン電位がある閾値を超えると発火（シナプスから顆粒が放出）するという動きを模倣している

この発火の関数モデルは当初はステップ関数（後述）であり、現在用いられる学習アルゴリズム実装上不都合

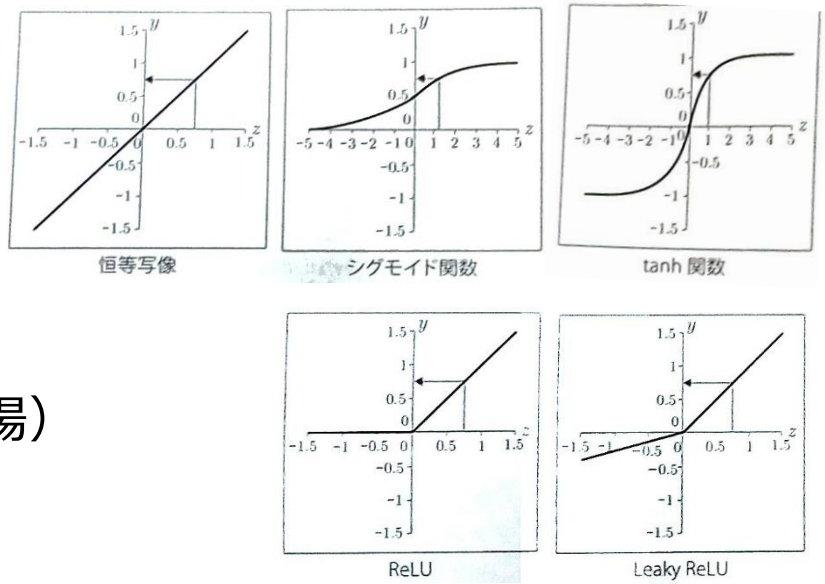


$$y_k = f(w_{k1}x_1 + w_{k2}x_2 + b_k)$$

※ここで、Σは（重み付きの）和をとるという操作を表す

活性化関数の種類（入力:x、出力:y）

- 恒等写像、線形関数
  - $y = x$
- シグモイド関数（Sigmoid）
  - $y = \frac{1}{1+e^{-x}}$
- tanh関数（e-learning教材で登場）
  - $y = \frac{e^{x}-e^{-x}}{e^{x}+e^{-1}}$
- ReLU（Rectified Linear Unit）
  - $y = \max(x,\ 0)$
- Leaky ReLU
  - $y = x \ \ if\ x \geq 0$
    $y = ax \ \ if\ x < 0$
- ソフトマックス関数（Softmax）
  - $y_i = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}}$ K：出力層のユニット数



（藤田、高原、「実装ディープラーニング」, p.52）

$0 \leq y_i \leq 1$、$\sum_{i=1}^{K} y_i = 1$ となるため、確率変数とみなせる

活性化関数は他にも種類があるが割愛

損失関数（Loss Function）

推測した出力データと教師データとの間の誤差を計算する関数

n:サンプル番号、y_n：出力データ、t_n：教師データとする

- 平均二乗誤差（Mean Squared Error）　連続値の回帰の場合

$$E_n = \frac{1}{2}(y_n - t_n)^2、\ E = \frac{1}{K}\sum_{n=1}^{K} E_n$$

- クロスエントロピー　出力が離散値の場合
  - $E_n = -\sum_{k=1}^{K} t_{nk} \log y_{nk}、\ E = \sum_{n=1}^{K} E_n$

    t_{nk}：nサンプル目のクラスkの教師データ

    y_{nk}：nサンプル目のクラスkの出力データ
- 2値（バイナリ）クロスエントロピー　2値分類の場合

$$E_n = -\{t_n \log y_n + (1 - t_n)\log(1 - y_n)\}、\ E = \sum_{n=1}^{K} E_n$$

One-hotベクトル

- ラベルデータのベクトル表現
- 総ラベル数Nの i 番目のラベルを、N次元ベクトルの i 番目の要素だけを1とし、その他は0のベクトルで表す
- 例：総ラベル数5（1～5）で3番目ラベルは [0, 0, 1, 0, 0] となる

- 分類問題の教師ラベル表現として使われる
  - 分類問題では出力層のノード数を総ラベル数にし、Softmax関数を活性化関数とする
  - 教師データをOne-hotベクトルで与えると、正解ラベルに相当するノードの出力が1、それ以外のノードの出力が 0 に近づくようにノードの重みを更新する
  - Softmax関数を使うので、それぞれの出力ノードの値は、そのラベルに属する確率とみなすことができる

誤差最小化手法

- 勾配降下法

  重みパラメータ全体ならなるベクトルwは、深層学習の場合、一般に巨大次元のベクトルである。
  このwに対する誤差Eの**勾配** ∇E というベクトルを計算し、これをもとに誤差Eが小さくなる方向にwを更新する（勾配ベクトル ∇E の向きは、Eが大きくなる向きに一致する）

  - $w = w - \varepsilon \nabla E$、　ε：学習係数（0.01や0.001など小さな数）

  - 仮に誤差を二乗誤差 $\mathrm{E} = \frac{1}{2}(Y - t)^2 = \frac{1}{2}(wX - t)^2$ とすると、

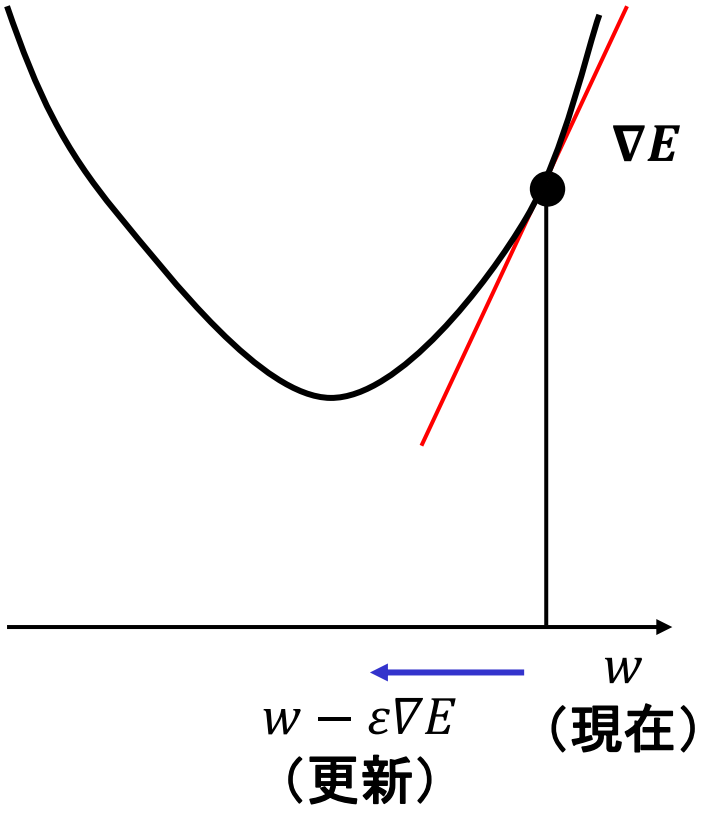
    Eのwでの微分は　$\nabla E = (wX - t)X^T$ となる

  - 誤差をクロスエントロピーとすると、
    $\nabla E = -\sum_{n=1}^{K}(y_n - t_n)\, X$

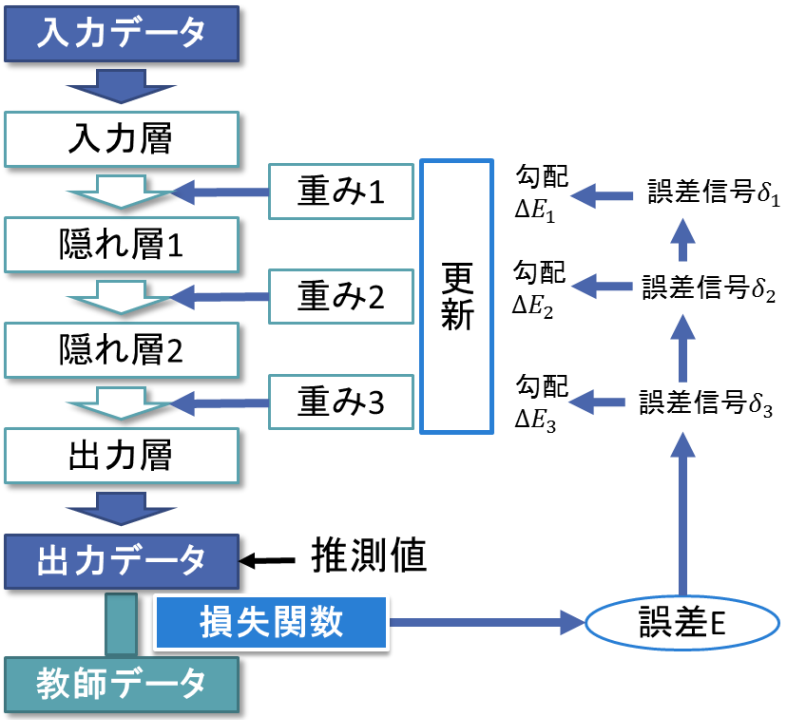
- あとで、勾配降下法をExcelで説明する

勾配降下法のイメージ

- あるウェイトwの時点での損失関数の勾配を求め、損失が減少する方向にwを更新する
- 勾配∇Eが正ならwを減少させ、負ならwを増加させる
- 学習係数 ε が大きい場合
  - 早く収束点に向かう
  - 収束点を通り過ぎることがある（収束せずに振動する）
- 学習係数 ε が小さい場合
  - 多くの学習回数が必要
  - 着実に収束点に向かう（収束しやすい）
  - 極小解で収束する場合がある（最小解ではない）
- 多次元でもあり、一概に決めることが難しい



誤差逆伝播法



∘ はアダマール積で、同じサイズの2つの行列の要素ごとの積

多層のニューラルネットワークでは誤差を伝播させて各層で重みを更新する

勾配は誤差信号δと入力から求められ、誤差信号は次のように求められる

zは隠れ層の出力

$$\delta_3 = (Y - t) \circ f_3'(Z_3)$$

f'は活性化関数の微分

$$\delta_2 = \left((W_3)^T \delta_3\right) \circ f_2'(Z_2)$$
$$\delta_1 = \left((W_2)^T \delta_2\right) \circ f_1'(Z_1)$$

$$\nabla E_3 = \delta_3 X_2^T$$
$$\nabla E_2 = \delta_2 X_1^T$$
$$\nabla E_1 = \delta_1 X_0^T$$

誤差信号が逆方向に伝播しながら勾配を求めるようになっている

勾配消失問題

- 誤差逆伝播法では、誤差が出力層側から入力層側に伝播しながら各層の重みを更新する
- 活性化関数によっては、活性化関数の微分が小さな値になり、それが層を重ねることで誤差がほぼ0になる
  - シグモイド関数の微分の最大値は0.25
  - 3層分逆伝播すると3層目は0.25^3となり、最初の誤差の約0.016倍となる
- ReLUを使うことでこの問題を軽減することができる
  - ReLUは入力が正の時の微分は常に1

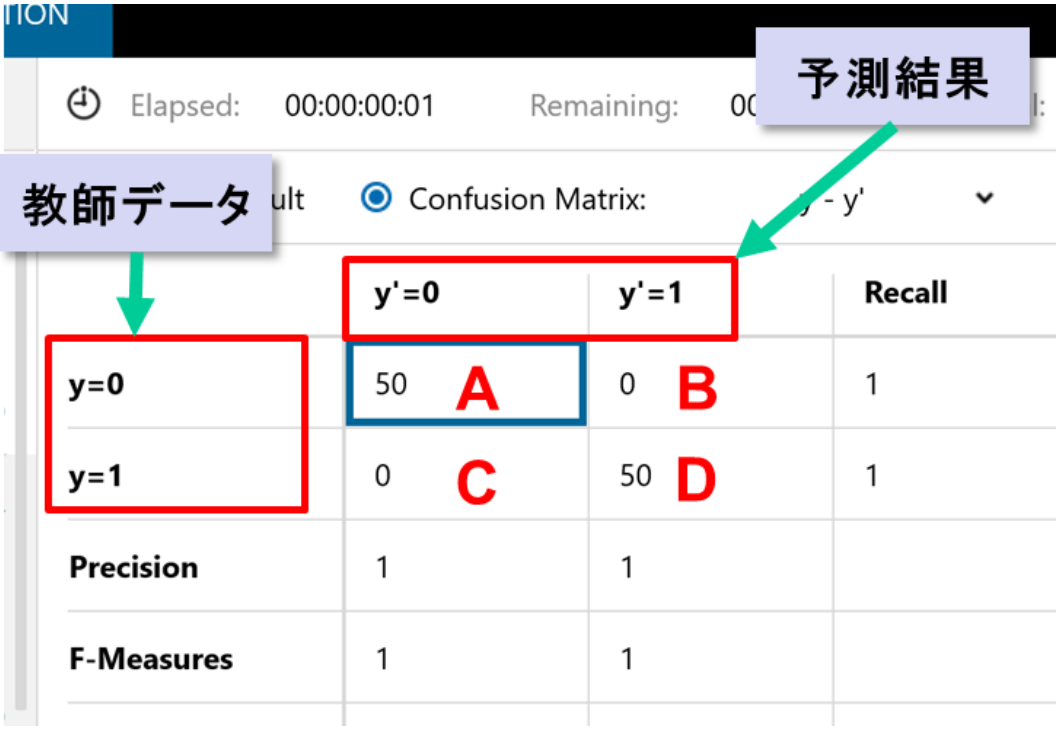
バッチ学習とミニバッチ学習

- バッチ学習
  - 用意した学習データや学習評価データを一度に使って学習を行う
- ミニバッチ学習
  - データを小分けにして学習を繰り返す
- 行列計算はデータをGPUに展開して計算を行うので、一度に計算できるデータ量はGPUのメモリに依存する
  - ミニバッチのサイズを大きくすれば学習が速くなるが、GPUのメモリ容量に応じてミニバッチのサイズを調整する必要がある
- 全サンプルの学習を1回終えることを1エポック（Epoch）と呼ぶ

過学習、過剰適合

- ディープラーニングでは、層やユニットが増えると
  パラメータ数が増え、学習データの情報を全て保持できる
  - 学習データ自体の推測はほぼ100%にできる
- しかし、学習データ以外のデータは未知のデータなので推測精度が悪くなる場合がある
  - たとえるなら、過去問は暗記して完璧だが本番が全然ダメという受験生
- 過学習を抑える方法
  - 学習評価データを使ったエポック数の設定
  - 正則化
    - ✓ 学習時のパラメータの更新幅に制約をつける
  - ドロップアウト
    - ✓ 学習時にランダムにユニットを無効化してネットワークの構造を変える

分類の評価指標



Accuracy（正解率）: $\frac{A+D}{A+B+C+D}$

Recall（再現率）: $\frac{A}{A+B}$、$\frac{D}{C+D}$

Precision（適合率）: $\frac{A}{A+C}$、$\frac{D}{B+D}$

F-Measure（F尺度）:
RecallとPrecisionの調和平均
（逆数の算術平均の逆数）

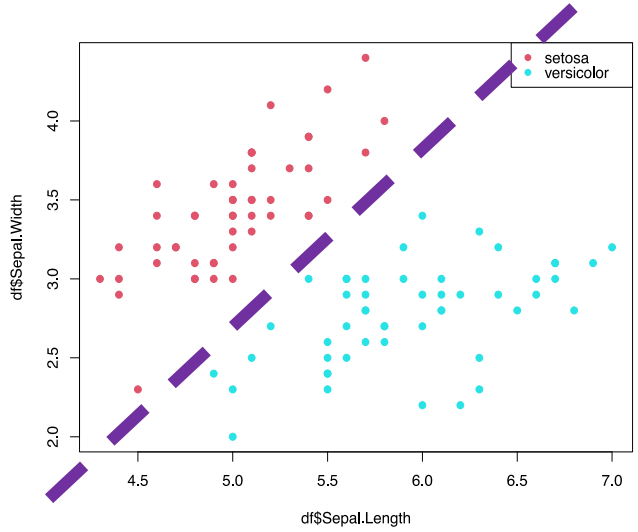
$$\frac{2 \cdot Recall \cdot Precision}{Recall + Precision}$$

損失関数と勾配降下法

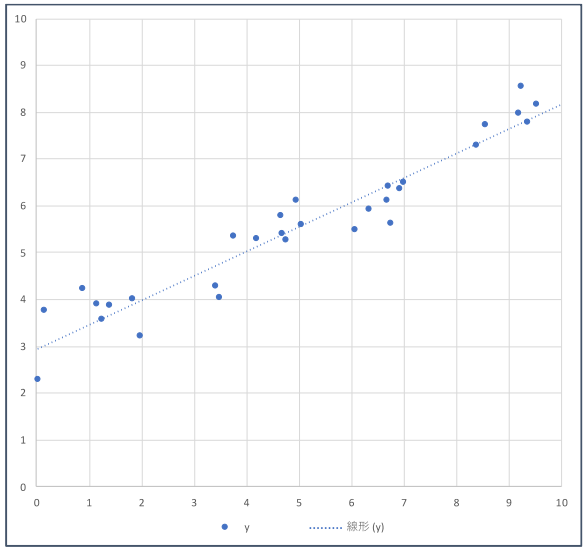
Excelによるデモ

2つのタスク：分類と回帰

- 分類
  - データをいくつかのグループに分ける
  - あるいは、個々のデータにラベルを付加する
  - 例: メールのスパム判定、画像のカテゴリ判定



- 回帰
  - 入力（説明変数）から出力（目的変数）を予測
  - 入力から出力が得られるメカニズムを関数でモデル化



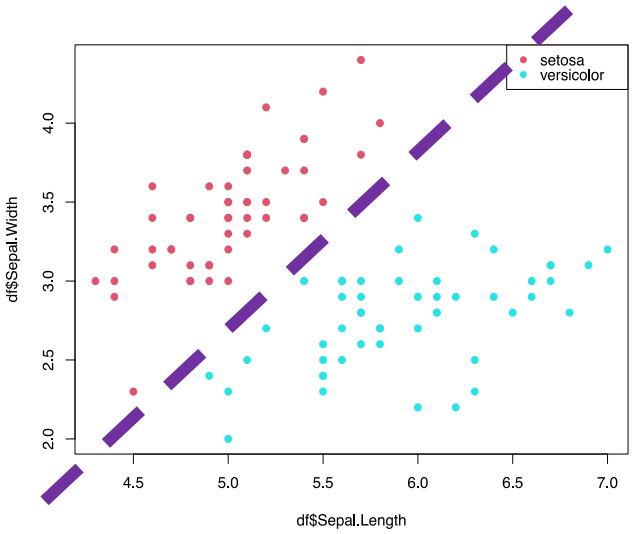
それぞれのタスクに関する手法の利用場面

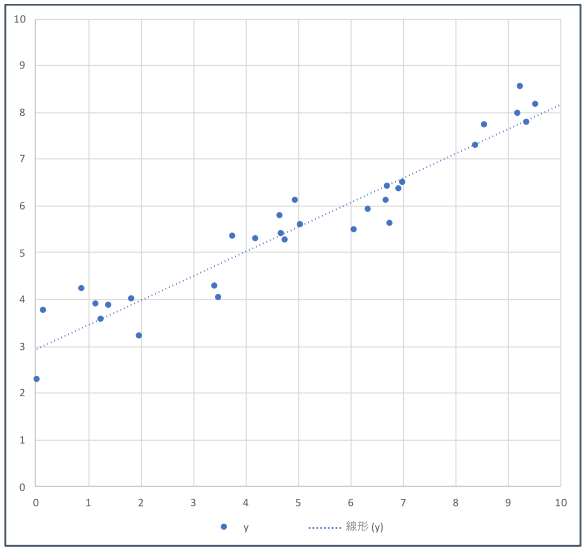
- 未知データについての推測
  - 分類タスクの場合、未知データがどのグループに属するかを推測
  - 回帰タスクの場合、未知の入力値から出力値を推測

- 既存データの説明
  - 分類の様子から知見を得る（そもそもグループがいくつあるかとか、各グループ間の関係など）
  - 変数間の関係から知見を得る

数学的視点から見た機械学習のタスク

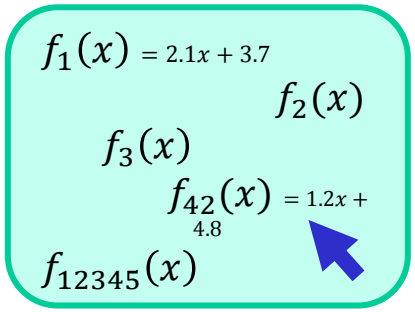
- 分類および回帰
  - 関数のあてはめ（フィッティング）
  - パラメータを持った関数族を設定し、損失関数を適切に定め（2乗誤差など）、その値が最小になるようにパラメータを決定する





損失関数と最適化

- 回帰問題の場合、当てはめに利用する関数族を設定する
  - 例： $y = ax + b$（直線の族。 a、b の組一つ一つについて、直線 $y = ax + b$ が族に含まれる）
- 関数族のパラメータを変数とする関数を適切に設定し、それを最小化することを目標とする。この関数を**損失関数**という。
  - 上の例の場合、 損失関数は $f(a, b)$ という形の関数
  - 損失関数というのはパラメータの悪さ（良さ）を測るための指標となるもの
  - 具体例は次の最小2乗法を見てください



候補となる関数族

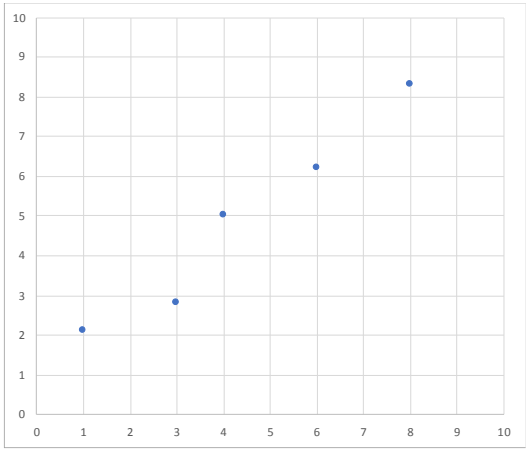
数値化（損失関数の値）

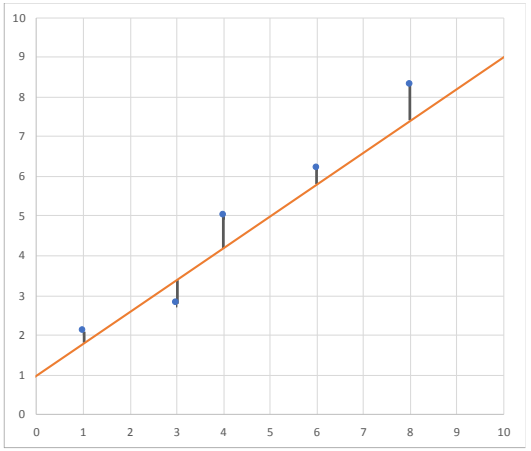
候補となる各関数を数値化し、その値の一番小さいものを選ぶ。

各関数を数値化する際に利用するのが損失関数という関数（関数の関数）

最小2乗法（あとでExcelでも説明します）

- 回帰問題の場合、損失関数として以下の2乗誤差が用いられることが多い
- 下図左のように入力と出力の組 $\{(x_i, y_i)\}_{i=1,\ldots,N}$ が与えられているとき、例えば、入力 x に対して出力 $\hat{y}$ を $\hat{y} = ax + b$ の形の関数で予測（**単回帰**）
- **2乗誤差** $E = \frac{1}{2}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 = \frac{1}{2}\left(\begin{matrix}(y_i - \hat{y}_i)\text{たち}\\ \text{の2乗の和}\end{matrix}\right) = \frac{1}{2}$（下図右の縦棒の長さの2乗の和） を最小にするように a、 b を決定する

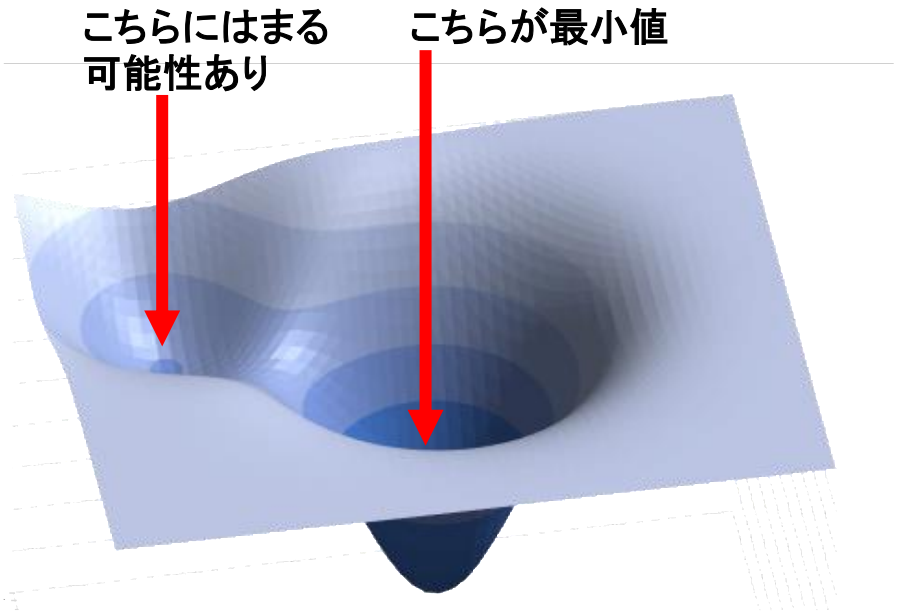




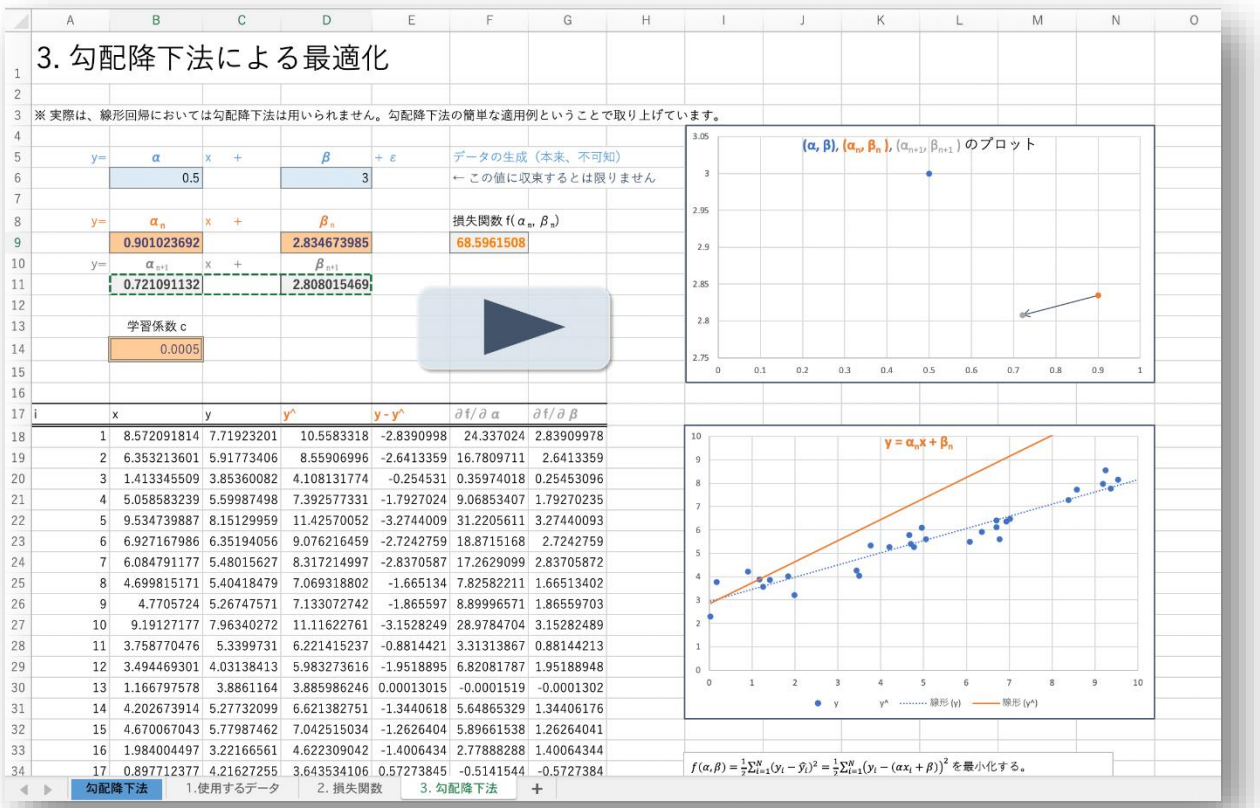
$\hat{y} = ax + b$

勾配降下法による最適化（Excelでも説明）

- 損失関数などの関数が最小値をとる点を探索する方法
- 候補点（ときにはランダムで選択）から出発し、損失関数が減少する向きへ歩を進める
- 損失関数 f が増加する向きというのは、関数の**勾配** $\nabla f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$ の向きとして計算できる（あとで説明します）
- 勾配 ∇f の向きとは逆向きに少しずつ移動することを繰り返せば、最小値に行き着くだろうという（ある意味楽観的な）手法
- 深層学習でも用いられる



Excelによる勾配降下法のデモ



i	x	y	y^	y - y^	∂f/∂α	∂f/∂β
1	8.572091814	7.71923201	10.5583318	-2.8390998	24.337024	2.83909978
2	6.353213601	5.91773406	8.55906996	-2.6413359	16.7809711	2.6413359
3	1.413345509	3.85360082	4.108131774	-0.254531	0.35974018	0.25453096
4	5.058583239	5.59987498	7.392577331	-1.7927024	9.06853407	1.79270235
5	9.534739887	8.15129959	11.42570052	-3.2744009	31.2205611	3.27440093
6	6.927167986	6.35194056	9.076216459	-2.7242759	18.8715168	2.7242759
7	6.084791177	5.48015627	8.317214997	-2.8370587	17.2629099	2.83705872
8	4.699815171	5.40418479	7.069318802	-1.665134	7.82582211	1.66513402
9	4.7705724	5.26747571	7.133072742	-1.865597	8.89996571	1.86559703
10	9.19127177	7.96340272	11.11622761	-3.1528249	28.9784704	3.15282489
11	3.758770476	5.3399731	6.221415237	-0.8814421	3.31313867	0.88144213
12	3.494469301	4.03138413	5.983273616	-1.9518895	6.82081787	1.95188948
13	1.166797578	3.8861164	3.885986246	0.00013015	-0.0001519	-0.0001302
14	4.202673914	5.27732099	6.621382751	-1.3440618	5.64865329	1.34406176
15	4.670067043	5.77987462	7.042515034	-1.2626404	5.89661538	1.26264041
16	1.984004497	3.22166561	4.622309042	-1.4006434	2.77888288	1.40064344
17	0.897712377	4.21627255	3.643534106	0.57273845	-0.5141544	-0.5727384

さまざまな工夫

デモのような簡単な場合でも、単純な実装では実用にならない可能性がある。手法の改良、他の手法の採用が必要になる。

- 勾配降下法の改良
- 局所的な最小値にはまりにくくする工夫
  - 確率的勾配降下法など
- 多層の場合にシステマティックに対応
  - 単純に合成関数の勾配を求めると面倒（→誤差逆伝播法）

第3章

教師あり・なし学習

機械学習の分類

	入力に関するデータ	出力に関するデータ（正答）	活用例
教師あり学習	有り	有り	不動産価格の予測（回帰）、検査結果に基づく陽性判定（分類）
教師なし学習	有り	無し	顧客の属性に基づく分類、情報の集約
強化学習	有り	直接的な正答は無いが、報酬が与えられる	ロボットの歩行学習、将棋、囲碁

教師あり学習

- 正解の用意された問題集（教師データ）がある
- 教師は正解の導き方を詳しく説明してくれない
- 学習者（コンピュータ）は問題と正解の関係を推測する
- これまで見たことの無い新しい類似の問題に対して正解を導けるようになることを目指す

教師あり学習の概要

- 入力とそれに対応する正解出力の組からなる教師データを用いる
- 教師データに基づいて入力と出力の関係を表現するモデルを構築する
- 新しい入力に対して精度の高い予測出力ができることを目指す
- 出力が連続値の場合「回帰」モデル、出力がカテゴリの場合「分類」モデルを構築する

ex. 不動産価格の予測（回帰）、検査結果に基づく陽性判定（分類）

教師なし学習

出力に関するデータ（正解）が与えられていない
クラスタリングか次元圧縮を目的とする

- クラスタリング
  - 入力データに基づいてサンプルをいくつかのクラスタに分類する
- 次元圧縮
  - 多種類の入力を少数種類の入力にまとめる

ex. 顧客の属性に基づいて顧客をいくつかのクラスタに分類する（クラスタリング）、学生の複数科目の試験成績データに基づいて学生の能力を説明できる少数の変数を作る（次元圧縮）

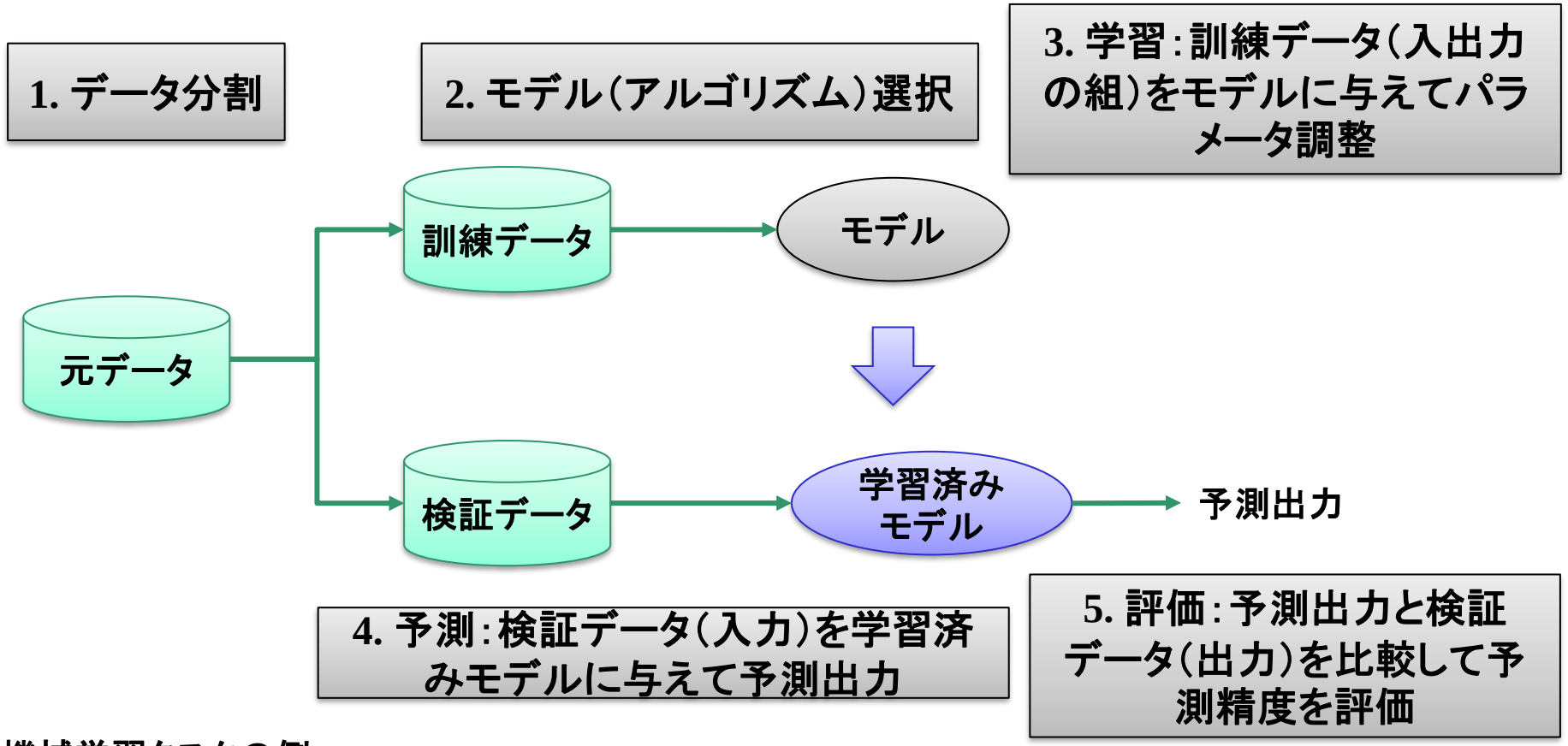
強化学習（本授業の範囲外）

試行錯誤を通じて報酬が最大となるような行動や選択を学習する

ex.

- ロボットの歩行学習
  - 歩けた距離を報酬とする。機械は手足の動かし方を試行錯誤して報酬が最大になるような動かし方を学習する
- 将棋
  - 敵の王将を取ることを報酬とする。駒の動かし方を試行錯誤して報酬が最大になる戦略を学習する

機械学習のプロセス



機械学習タスクの例:

- 腫瘍の検査値X(直径や面積、滑らかさ等)と良性・悪性の診断結果yが組になったデータ (X,y) が100サンプルある場合を考える
- 入力Xからyを予測できるモデル y=f(X,C) を構築したい(C:モデルのパラメータ)
- 80サンプルでモデルを訓練して、20サンプルでモデルの予測精度を評価する

機械学習のアルゴリズム

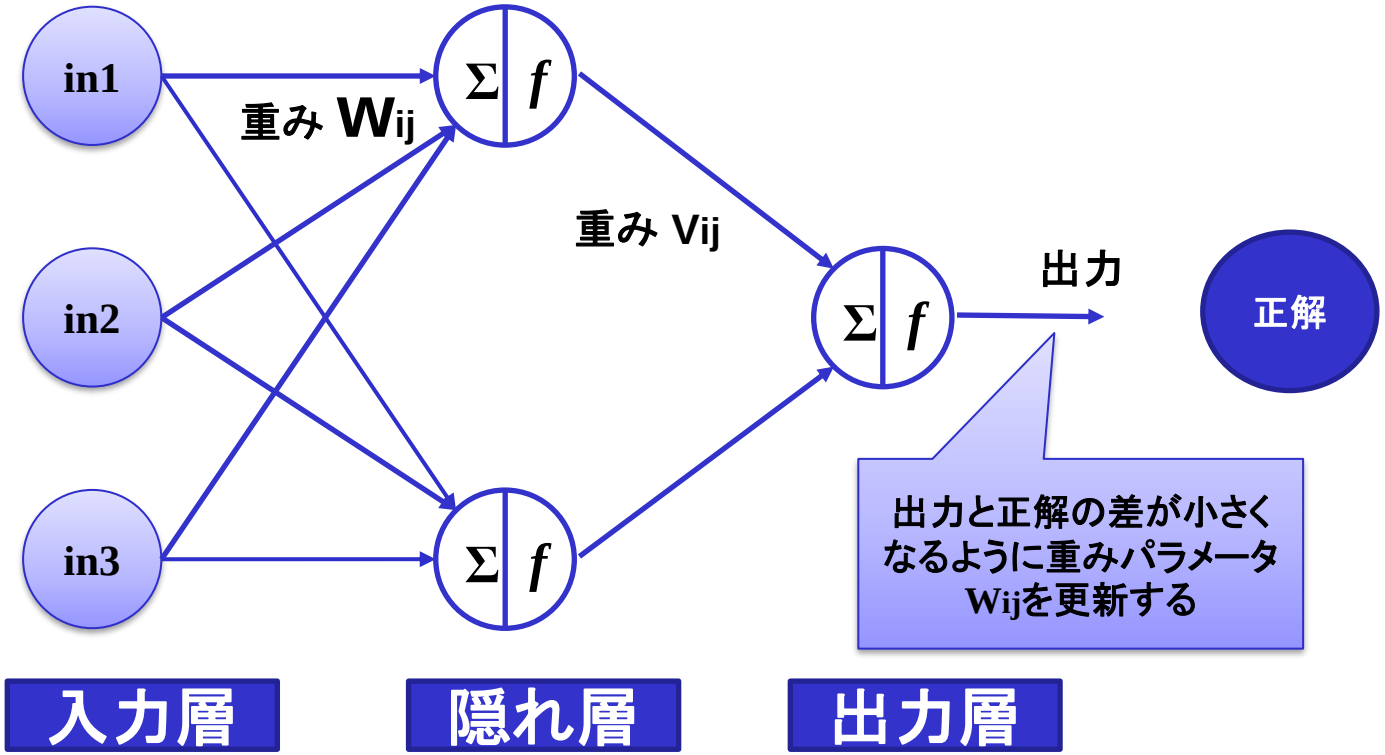
教師あり学習
- 分類
  - k近傍法、SVM、決定木、ランダムフォレスト、パーセプトロン，ロジスティック回帰，ニューラルネットワーク
- 回帰
  - 線形回帰、Ridge回帰、Lasso回帰、Elastic Net

教師なし学習
- クラスタリング
  - k平均法
- 次元削減
  - PCA、自己組織化マップ

強化学習（本授業の範囲外）
- Q-learning、SARSA、モンテカルロ法

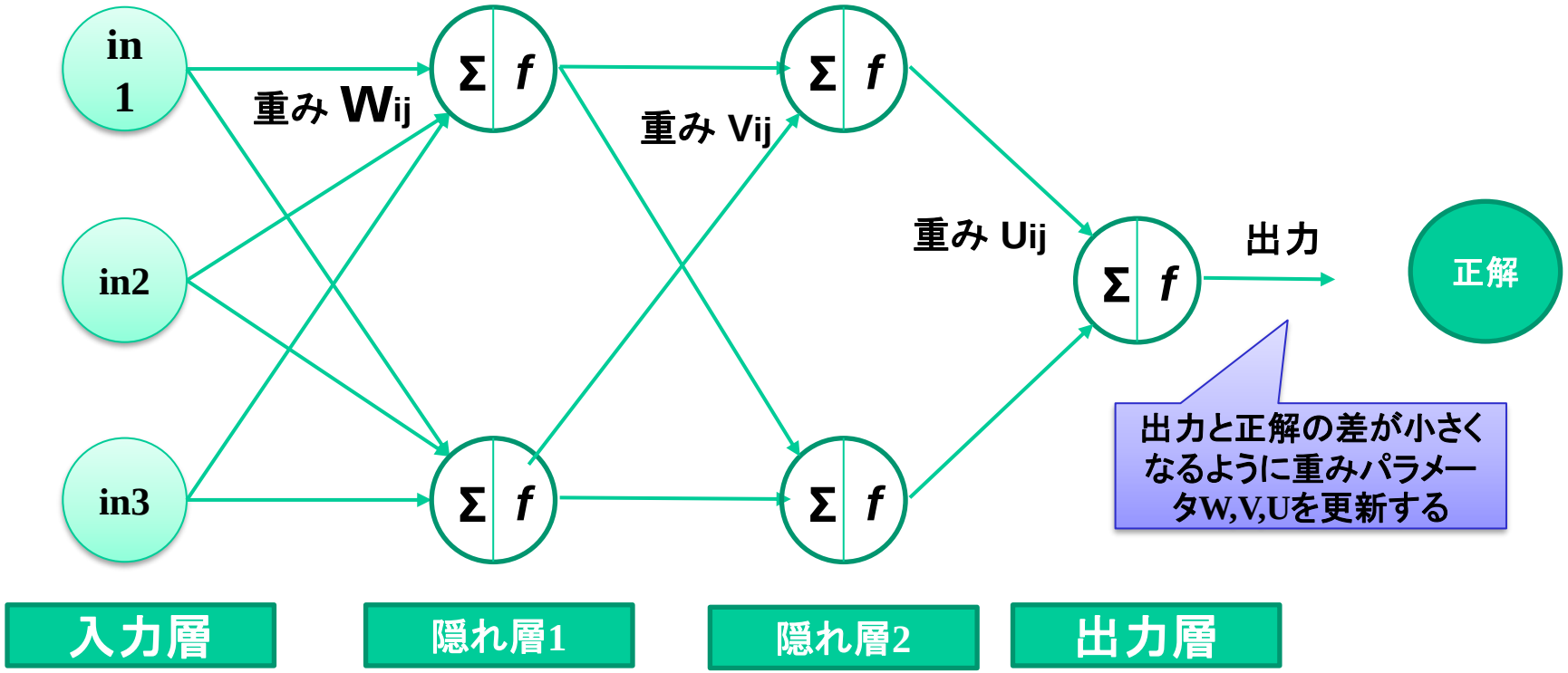
ニューラルネットワーク

脳の神経回路の仕組みを真似たモデル．機械学習アルゴリズムの一種
入力と対応する正解出力の組からなる教師データを与える
出力データと正解データの差が小さくなるように重みパラメータを更新する



深層学習

ニューラルネットワークの隠れ層を増やしたモデル
画像や音声等入力と対応する正解出力の組からなる教師データを与える
出力データと正解データの差が小さくなるように重みパラメータを更新する
機械は与えられたデータに基づいて特徴量を抽出・学習する



機械学習・深層学習のための言語

R

- ベクトル・行列やデータフレーム等データ処理機能、描画機能が標準で用意されており操作が統一されている
- 統計解析の機能が標準で用意されている
- 機械学習の各種アルゴリズムがパッケージとして豊富に提供されている
- 深層学習のライブラリはあるものの主流ではない

Python

- ベクトル・行列計算にはNumPyやSciPy、データフレーム処理にはpandas、描画にはMatplotlibのように外部ライブラリが必要で操作が統一されていない
- 統計解析の機能が標準で用意されていない（別途StatsModelsが必要）
- 機械学習のライブラリ（scikit-learn）がある
- 深層学習のライブラリ（TensorFlow, Keras, PyTorch, Caffe等）が充実しており主流となっている

演習で採用する環境

機械学習

- 言語と開発環境：RとR Studio
- ライブラリ：Rの各種パッケージ
- 実行環境：ローカル

深層学習

- 言語と開発環境：PythonとGoogle Colaboratory
- ライブラリ：Keras
- 実行環境：クラウド

教師あり学習（分類）

教師あり学習（再掲）

- 正解の用意された問題集（教師データ）がある
- 教師は正解の導き方を詳しく説明してくれない
- 学習者（コンピュータ）は問題と正解の関係を推測する
- これまで見たことの無い新しい類似の問題に対して正解を導けるようになることを目指す

教師あり学習の概要

- 入力とそれに対応する正解出力の組からなる教師データを用いる
- 教師データに基づいて入力と出力の関係を表現するモデルを構築する
- 新しい入力に対して精度の高い予測出力ができることを目指す
- 出力が連続値の場合「回帰」モデル、出力がカテゴリの場合「分類」モデルを構築する

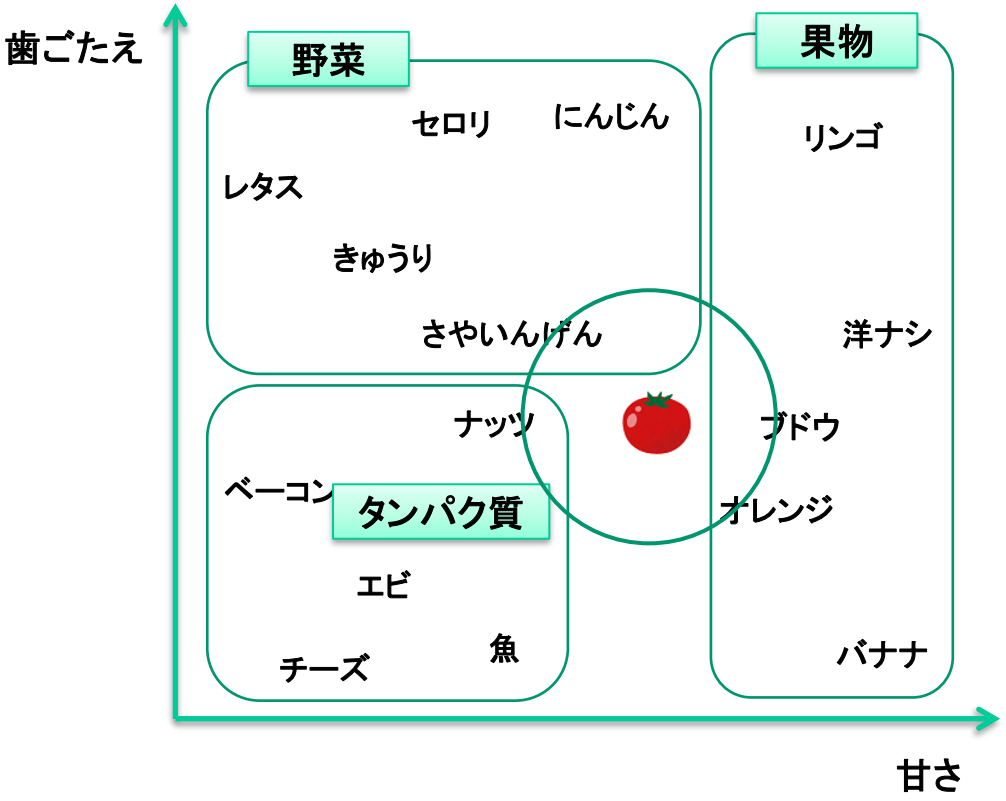
ex. 不動産価格の予測（回帰）、検査結果に基づく陽性判定（分類）

k-近傍法

教師あり学習（分類）

k-近傍法の概要と例

k-近傍法：特徴空間上で最も近い距離にある訓練データに基づいて分類を行う方法



例：トマトは野菜か果物か

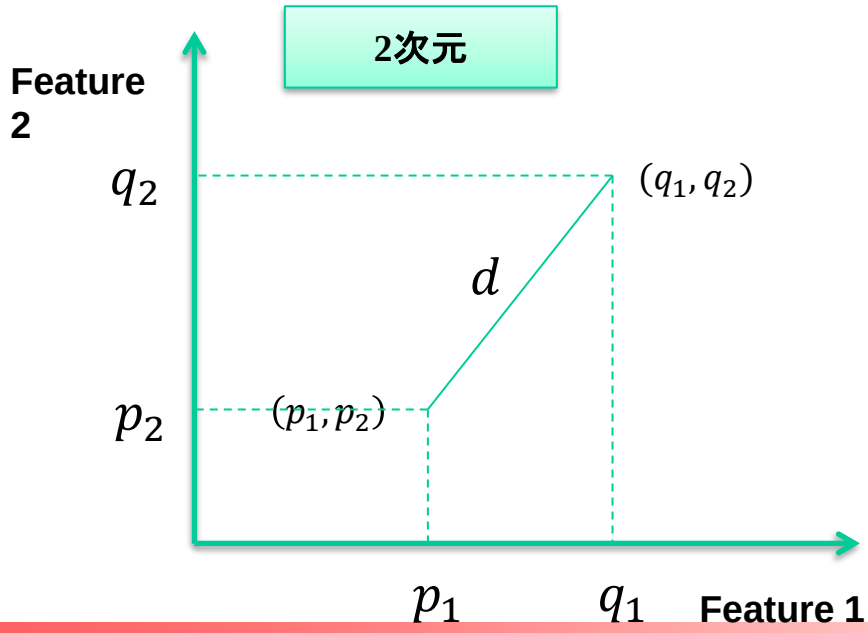
- 甘さと歯ごたえの2つの特徴量に基づいて判定すると仮定する
- 2次元特徴空間（平面）における近傍点が野菜なら野菜、果物なら果物と判定
- この例ではトマトの4つの近傍はブドウ、オレンジ、さやいんげん、ナッツだが、4つのうち2つが果物なので果物！

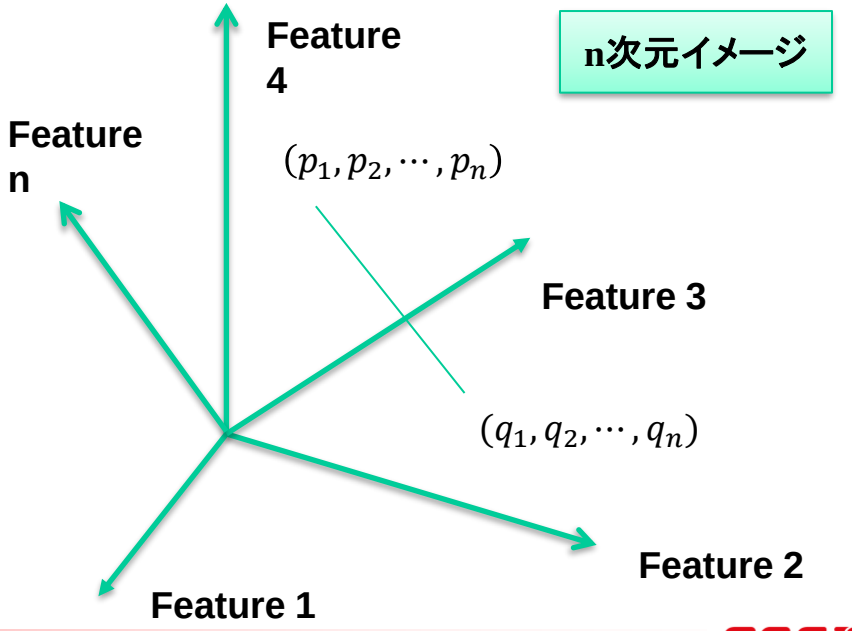
出典：Brett Lantz:「Rによる機械学習」翔泳社（2017）を基に鈴木作成

k-近傍法における次元と距離

例：トマトは野菜か果物か
- 甘さと歯ごたえの2つの特徴量に基づいて判定
- 2次元特徴空間（平面）における近傍点が野菜なら野菜、果物なら果物
- 2点$(p_1, p_2), (q_1, q_2)$間の距離
  - $d = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2}$

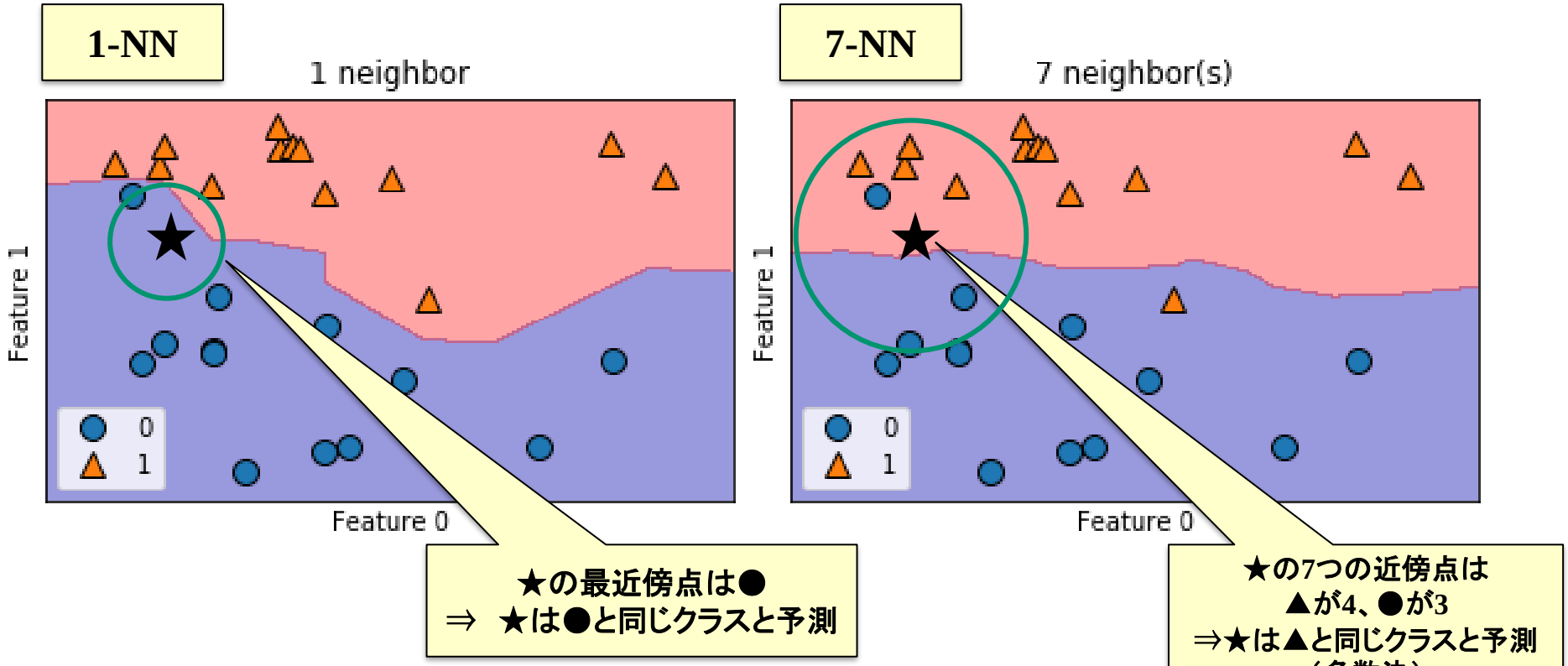
例：注目する腫瘍が良性か悪性か
- ガン細胞のサイズや滑らかさなど30の特徴量に基づいて判定
- 30次元特徴空間における近傍点が良性なら良性、悪性なら悪性
- 2点$(p_1, p_2, \cdots, p_n), (q_1, q_2, \cdots, q_n)$間の距離
  - $\sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2}$





k-近傍法による予測方法とkの値による違い

新しいデータ点は最近傍点（もしくはk個の近傍点うち多数派）と同じクラスと予測



- 線形分離不可能なデータセットでも分類可能
- kが小さい場合きめ細かく分類、大きい場合滑らかに分類される

※データ出典：A. C. Muller et al. mglearn.datasets

k-近傍法の処理手順と実装

処理手順

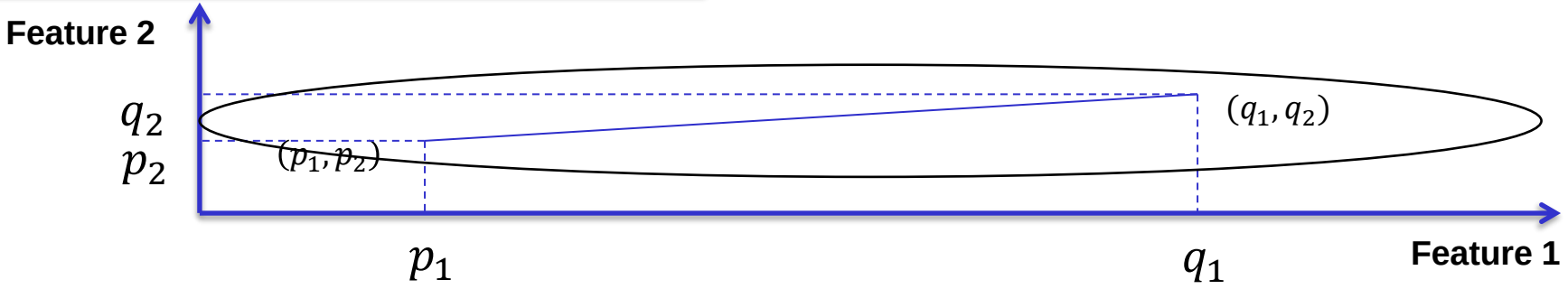
- 学習
  - 訓練データセット（特徴量・クラス）を格納するだけ（学習不要）
- 予測
  1. 未分類のデータ点1個に対して、分類済みの近傍データ点をk個見つける
  2. 未分類データ点は、分類済み近傍データ点の多数派が所属するクラスに分類する

実装

- Rのclassパッケージに含まれるknn実装など

k近傍法で用いるデータの前処理の必要性

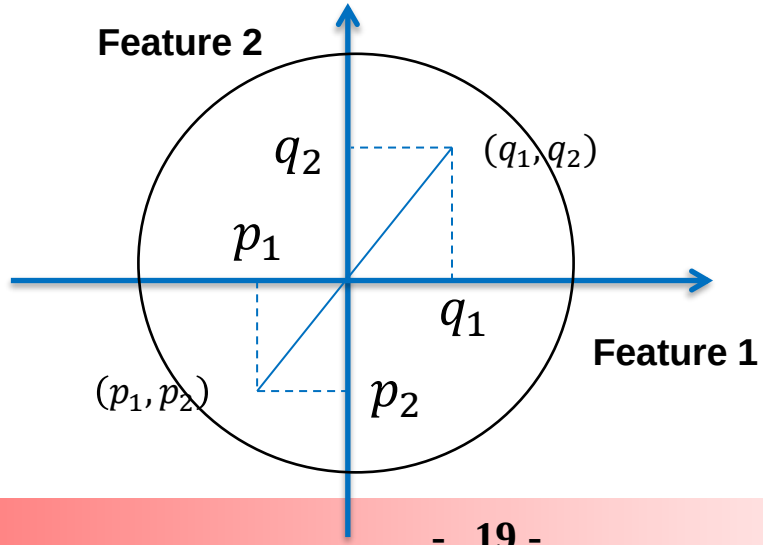
特徴量間で値の範囲が大きく異なる場合



$$\sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2} \sim \sqrt{(p_1 - q_1)^2} = |p_1 - q_1|$$

値の範囲が狭い特徴量がモデル構築において評価されない

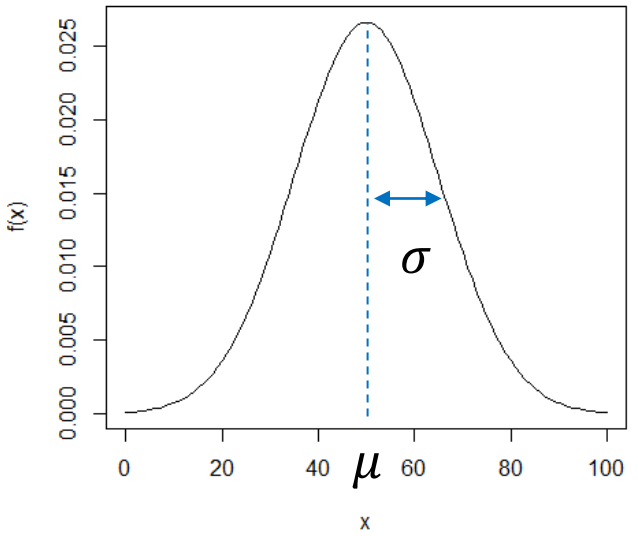
正規化・標準化



すべての特徴量について正規化・標準化を行い、どの特徴量も同等に評価されるようにする必要がある

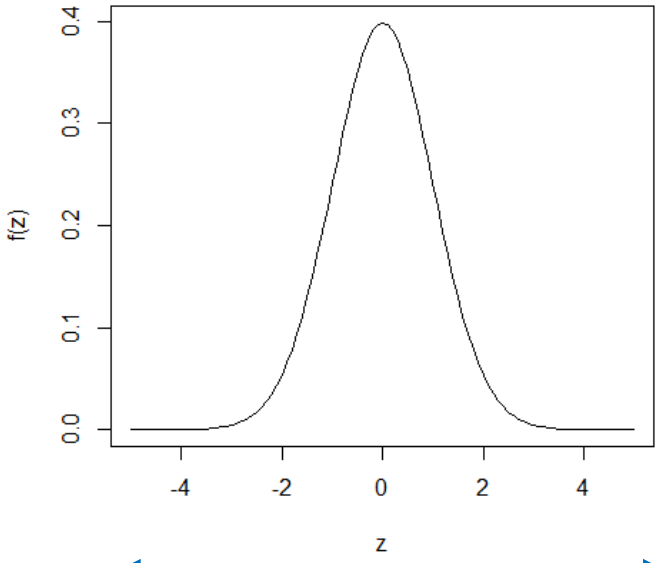
Zスコア標準化・最大最小正規化

Zスコア標準化



値の範囲が広すぎる

$$Z = \frac{X - \mu}{\sigma}$$



値の範囲が標準化された

最大最小正規化

$$X' = \frac{X - \min(X)}{\max(X) - \min(X)}$$

$$X' \in [0,1]$$

値の範囲が0以上1以下に標準化された

※参考：k-近傍法を利用した回帰

処理手順

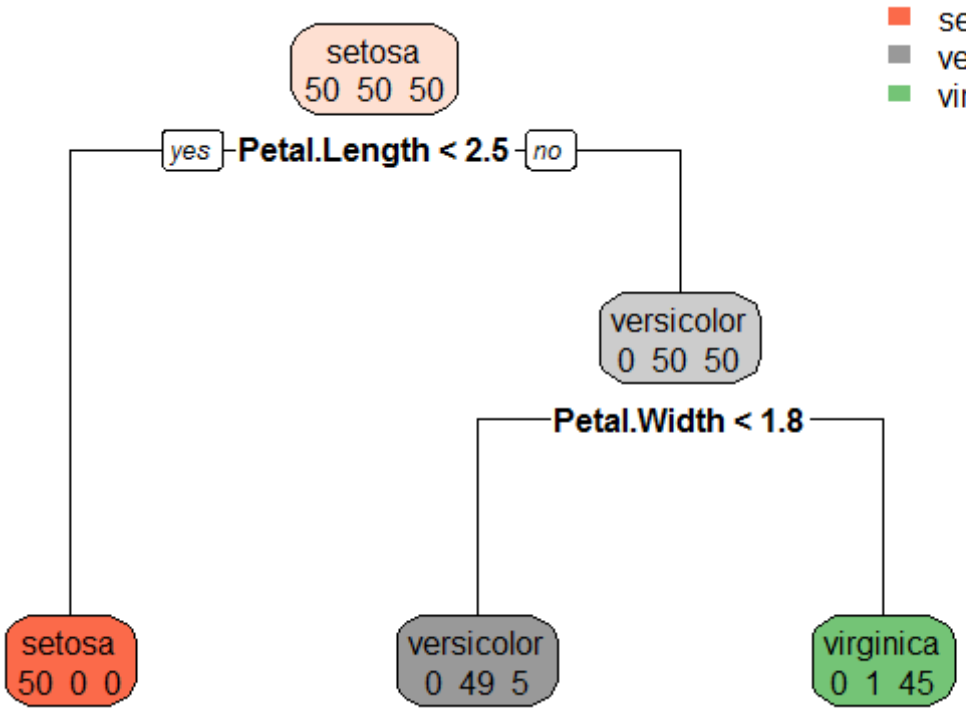
- 学習
  - 訓練データセット（特徴量・目的変数）を格納するだけ（学習不要）
- 予測
  1. 目的変数の値が未知のデータ点1個に対して、特徴空間における近傍データ点をk個見つける
  2. k個の近傍データ点の目的変数の平均を、注目データ点の目的変数の値とする

決定木

教師あり学習（分類）

決定木の概要と例

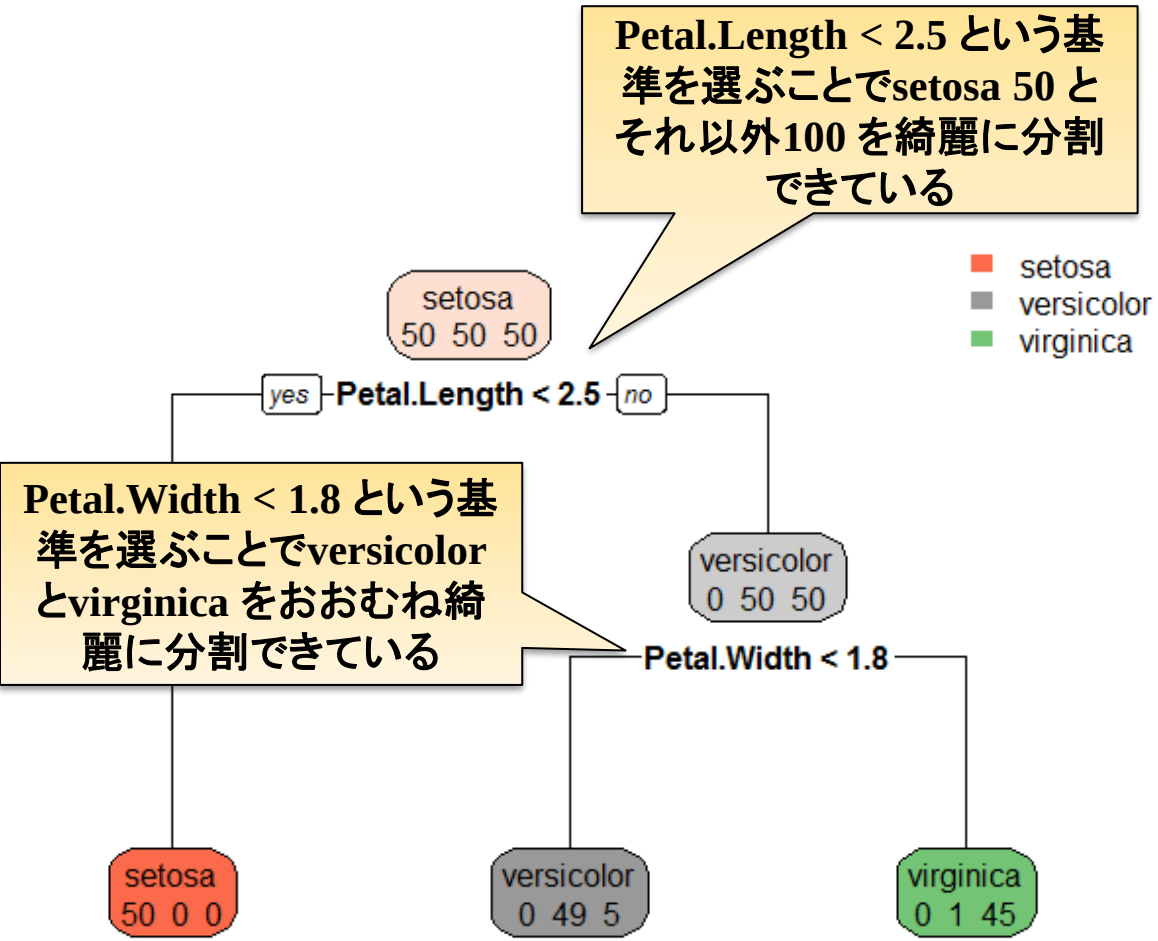
決定木：特徴量に関する条件分岐をたどると最終的に分類が決定する木構造



例：アヤメの種類を萼と花弁の長さと幅から決定する決定木

- Petal（花弁）の長さが2.5未満ならsetosa
  - 2.5以上で幅が1.8未満ならversicolor
  - 2.5以上で幅が1.8以上ならvirginica
- setosa 50個体、versicolor 50個体、virginica 50個体からなる全150個体を、本モデルによっておおむね綺麗に分けられている

決定木の学習手順



学習手順

1. 最も綺麗にデータを分割できる基準を選ぶ
   ※「情報利得」が最大となる基準
2. データを分割する
3. 終了条件を満たすまで1と2を繰り返す
   ※終了条件は、木の深さ、終端ノード数、ノードに含まれるデータ数、誤り率などを考慮する

実装

- RのrpartやC50など

決定木：
情報利得による分割基準の選択

エントロピー

エントロピー： 乱雑さ

- ノード N_tのエントロピー

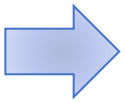
$$H(N_t) = -\sum_{i=1}^{n} p_i \log_2 p_i$$

- p_i：ノードN_tにおけるクラスi に属するインスタンスの割合
- n：クラス数

ノードN_t

クラス1: p_1
クラス2: p_2
クラス3: p_3
…
クラスn: p_n

ただし $\sum_{i=1}^{n} p_i = 1$



ノードN_tのエントロピー

$$H(N_t) = -\sum_{i=1}^{n} p_i \log_2 p_i$$

エントロピーの性質（2クラス）

$$H(N_t) = -\sum_{i=1}^{n} p_i \log_2 p_i$$

$n = 2$

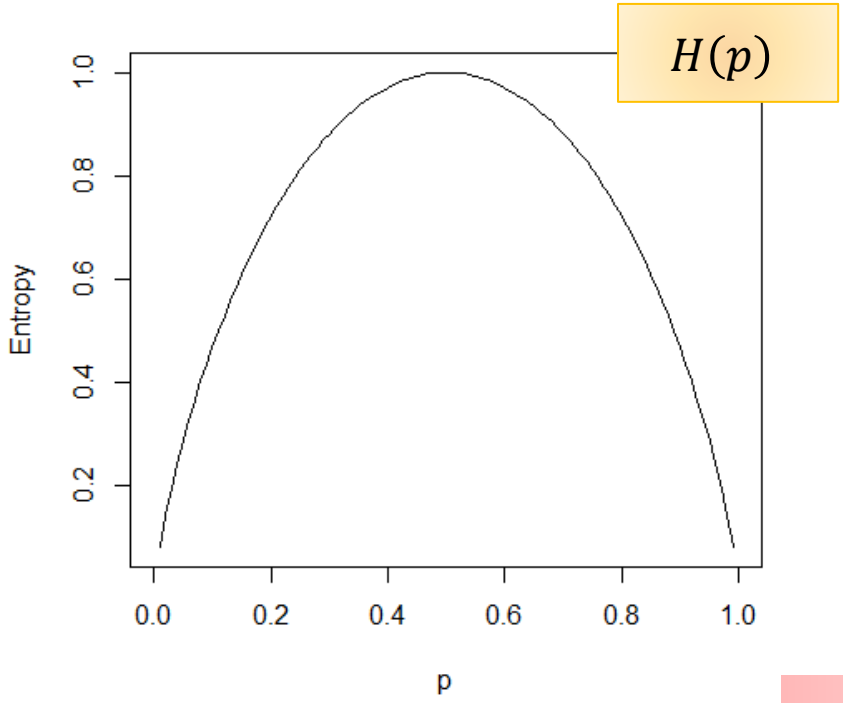
ノードN_t

クラス1: $p_1 = p$
クラス2: $p_2 = 1 - p$

※一方のクラスの割合をpとおいた

ノードN_tのエントロピー

$$H(p) = -p \log_2 p \ - (1-p) \log_2 (1-p)$$



エントロピーの性質（2クラス）

- $p = \frac{1}{2}$ で$H\left(\frac{1}{2}\right) = 1.0$（最大値）
  - データの割合が50:50 のときエントロピー最大
- $p \to 0 \ or \ p \to 1$ で$H(p) \to 0$
  - いずれか一方のクラスしかない場合エントロピーは0

クラスの混在の程度が高いほど
エントロピーが大きい

エントロピーの性質（nクラス）

ノードN_t

クラス1: p_1
クラス2: p_2
クラス3: p_3
…
クラスn: p_n

ただし $\sum_{i=1}^{n} p_i = 1$

ノードN_tのエントロピー

$$H(N_t) = -\sum_{i=1}^{n} p_i \log_2 p_i$$

エントロピーの性質（nクラス）

- データの割合が均等なとき（$p_i = \frac{1}{n}$）、エントロピー最大

$$H(N_t) = -\sum_{i=1}^{n} \frac{1}{n} \log_2 \frac{1}{n} = \log_2 n$$

- いずれか一つのクラスしか含まない場合エントロピーは0

クラスの混在の程度が高いほど
エントロピーが大きい

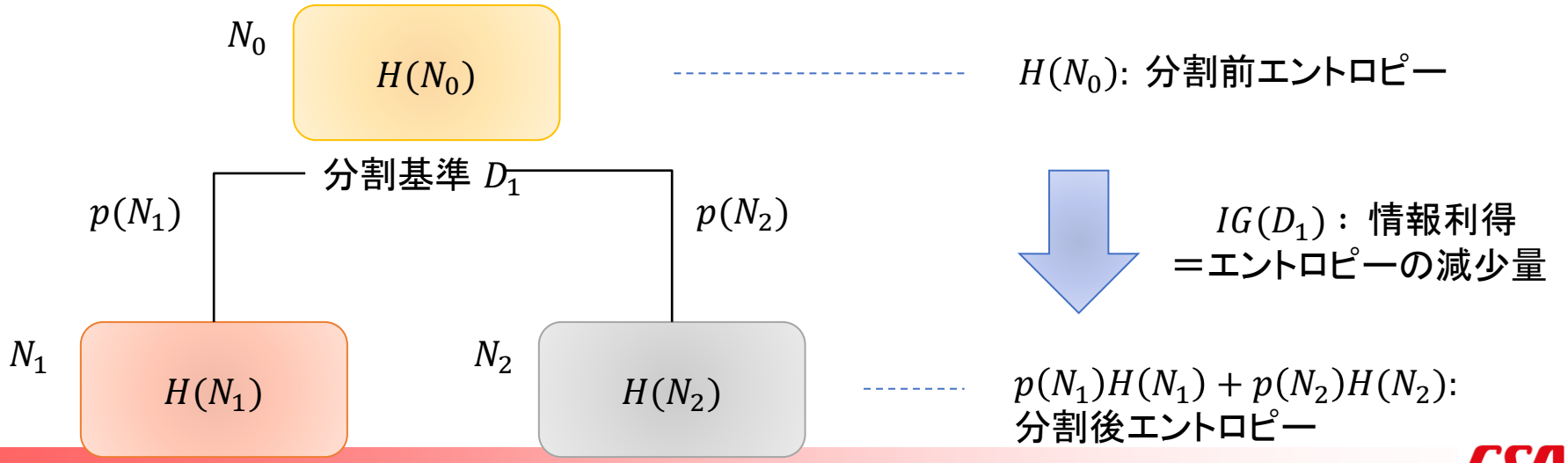
情報利得

- **情報利得**：分割によるエントロピーの減少量
  - ある分割D_1 によりノードN_0を2つのノード N_1, N_2に分割した場合の情報利得

$$IG(D_1) = H(N_0) - \{\, p\,(N_1)H(N_1) + p(N_2)H(N_2)\,\}$$

分割前エントロピー　　分割後エントロピー

  - H：エントロピー
  - p：分割比率（ $p(N_1) + p(N_2) = 1$ ）



ある分割D1による情報利得の計算例

$$H(N_0) = -\frac{1}{3}\log_2\frac{1}{3} - \frac{1}{3}\log_2\frac{1}{3} - \frac{1}{3}\log_2\frac{1}{3} = \log_2 3 = 1.584$$

分割D1:
Petal.Length < 2.5

N_0

setosa: 50
versicolor: 50
virginica: 50

Petal.Length < 2.5

$\frac{50}{150}$

$\frac{100}{150}$

N_1

setosa: 50
versicolor: 0
virginica: 0

N_2

setosa: 0
versicolor: 50
virginica: 50

$$H(N_1) = 0$$

$$H(N_2) = -\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2} = \log_2 2 = 1.0$$

情報利得

$$IG(D_1) = H(N_0) - \left\{\frac{50}{150} * H(N_1) + \frac{100}{150} * H(N_2)\right\} = 1.584 - 0.666 = 0.918 \text{ bit}$$

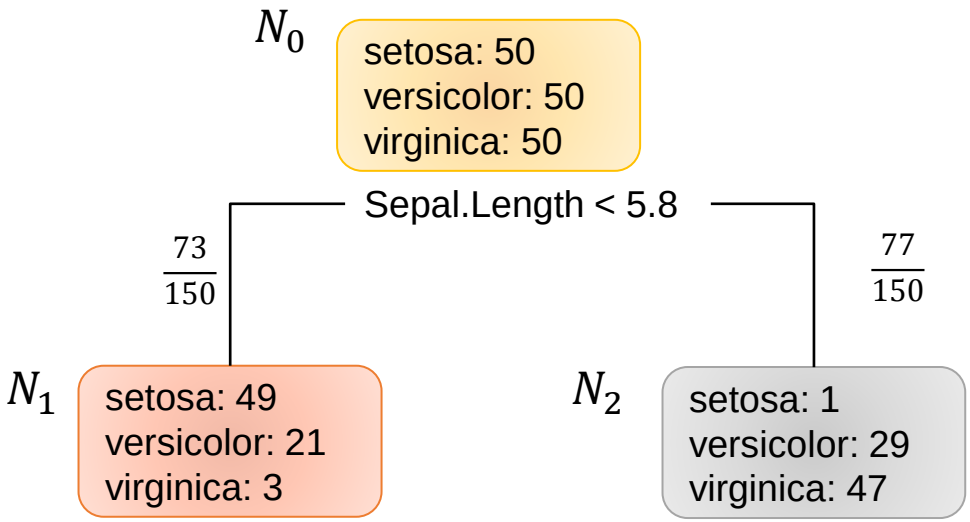
分割前エントロピー

分割後エントロピー

ある分割D2による情報利得の計算例

$$H(N_0) = -\frac{1}{3}\log_2\frac{1}{3} - \frac{1}{3}\log_2\frac{1}{3} - \frac{1}{3}\log_2\frac{1}{3} = \log_2 3 = 1.584$$

分割D2:
Sepal.Length < 5.8



$$H(N_1) = -\frac{49}{73}\log_2\frac{49}{73} - \frac{21}{73}\log_2\frac{21}{73} - \frac{3}{73}\log_2\frac{3}{73} = 1.092$$

$$H(N_2) = -\frac{1}{77}\log_2\frac{1}{77} - \frac{29}{77}\log_2\frac{29}{77} - \frac{47}{77}\log_2\frac{47}{77} = 1.046$$

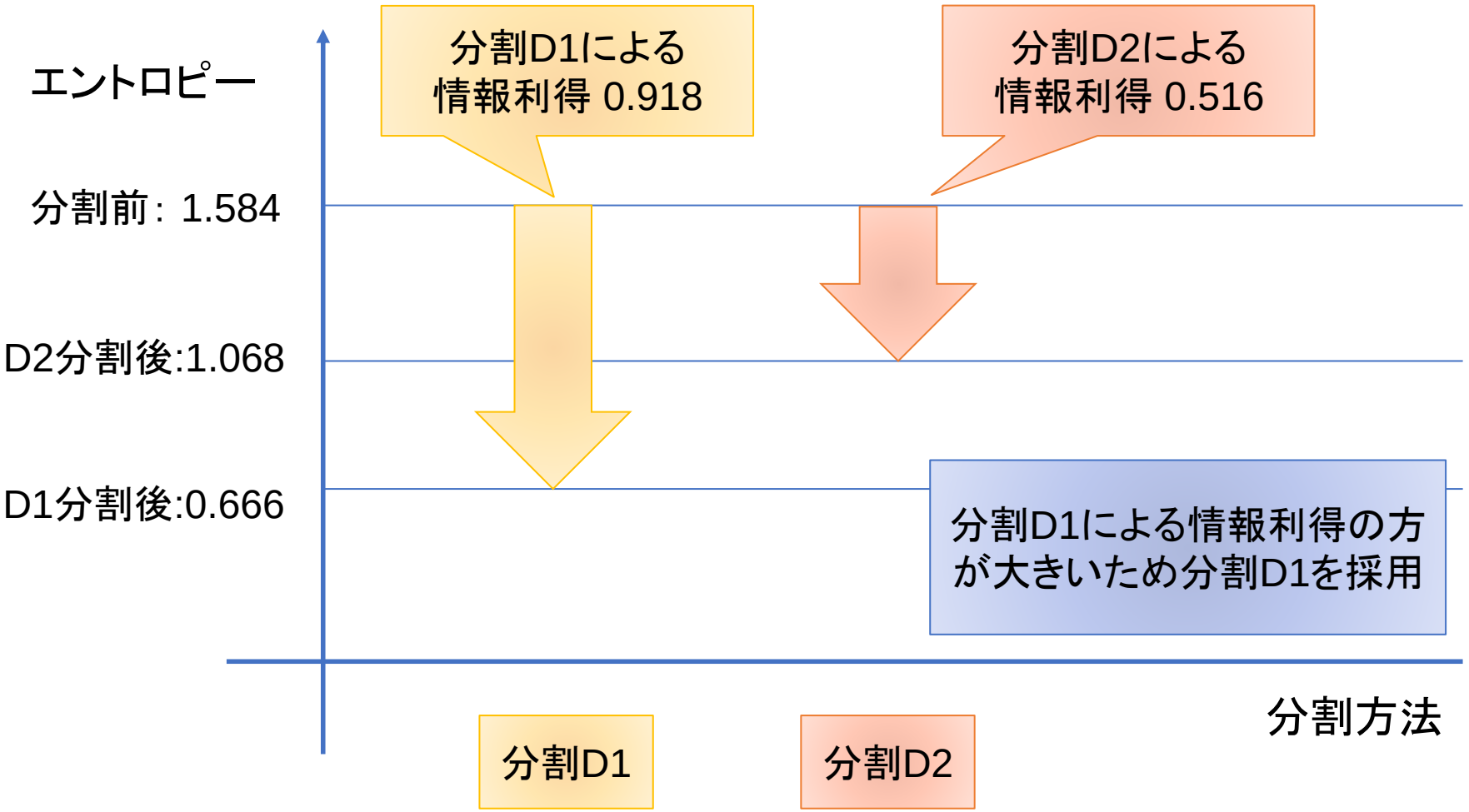
$$IG(D_2) = H(N_0) - \left\{\frac{73}{150} * H(N_1) + \frac{77}{150} * H(N_2)\right\} = 1.584 - 1.068 = 0.516\text{bit}$$

情報利得

分割前エントロピー

分割後エントロピー

分割D1と分割D2の情報利得の比較



決定木のメリット

分析結果の解釈や実装が容易

特徴量の正規化・標準化が不要

複数の決定木を組み合わせることで、ランダムフォレストなどの強力なアルゴリズムに発展させることが可能

ランダムフォレスト

ランダムサンプリングされた訓練データによって学習した多数の決定木を使用する方法

学習

1. 学習に用いるデータや特徴量をランダムに抽出する
2. 抽出したデータと特徴量で決定木を生成する
3. 1. と2. を繰り返す

予測

- 予測したい事例について、個々の決定木が予測した各クラスへの所属確率を平均し、確率が高いと判断されるクラスを予測結果とする

実装

- RのrandomForestパッケージなど

演習：教師あり学習（分類）演習ガイド参照

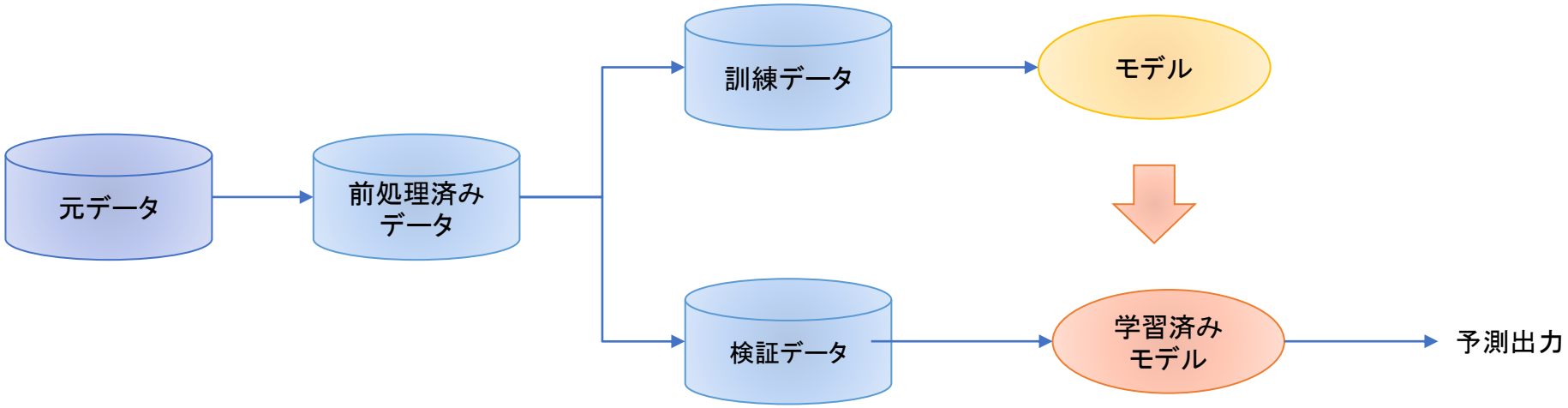
機械学習モデル構築（演習）手順

0. 前処理

1. データ分割

2. モデル（アルゴリズム）選択

3. 学習：訓練データ（入出力の組）をモデルに与えてパラメータ調整



4. 予測：検証データ（入力）を学習済みモデルに与えて予測出力

5. 評価：予測出力と検証データ（出力）を比較して予測精度を評価

演習：Wisconsin乳がんデータセットを利用

- 腫瘍細胞の30の特徴量から良性・悪性の診断を行う機械学習モデルを構築
- k近傍法か決定木を採用
- 必要なら正規化・標準化等の前処理を実施
- 全569事例のデータを訓練データ469事例、検証データ100事例に分割
- 訓練データでモデルを訓練
- 検証データを学習済みモデルに与え、その予測精度を評価

RとR Studio

RとR Studio

R

- 統計解析・可視化のための言語・開発実行環境
- オープンソース・フリーソフトウェアであり、Windows、MAC、Linux等様々なOSで利用可能
- 機械学習のためのライブラリも数多く提供されており、少ないコマンドで簡単にインストール・利用可能

R studio

- Rのための統合開発環境
- コンソールや高機能なエディタ、描画やコマンド履歴、デバッグ、ワークスペース管理のためのツールなどを備える
- クロスプラットフォームであり、各種OSで利用可能

R・Rtoolsのインストール

https://cran.r-project.org/ にアクセスし、Download and Install Rから自分のOSを選択

- Windowsの場合はbaseとRtools（recommended）をそれぞれダウンロードし、exeファイルを実行してインストール
- Rtoolsインストール中に”Add rtools to system PATH”に✓
  - Rtoolsはソースで配布されているライブラリ等をコンパイルして利用する場面で必要

R Studioのインストールと起動

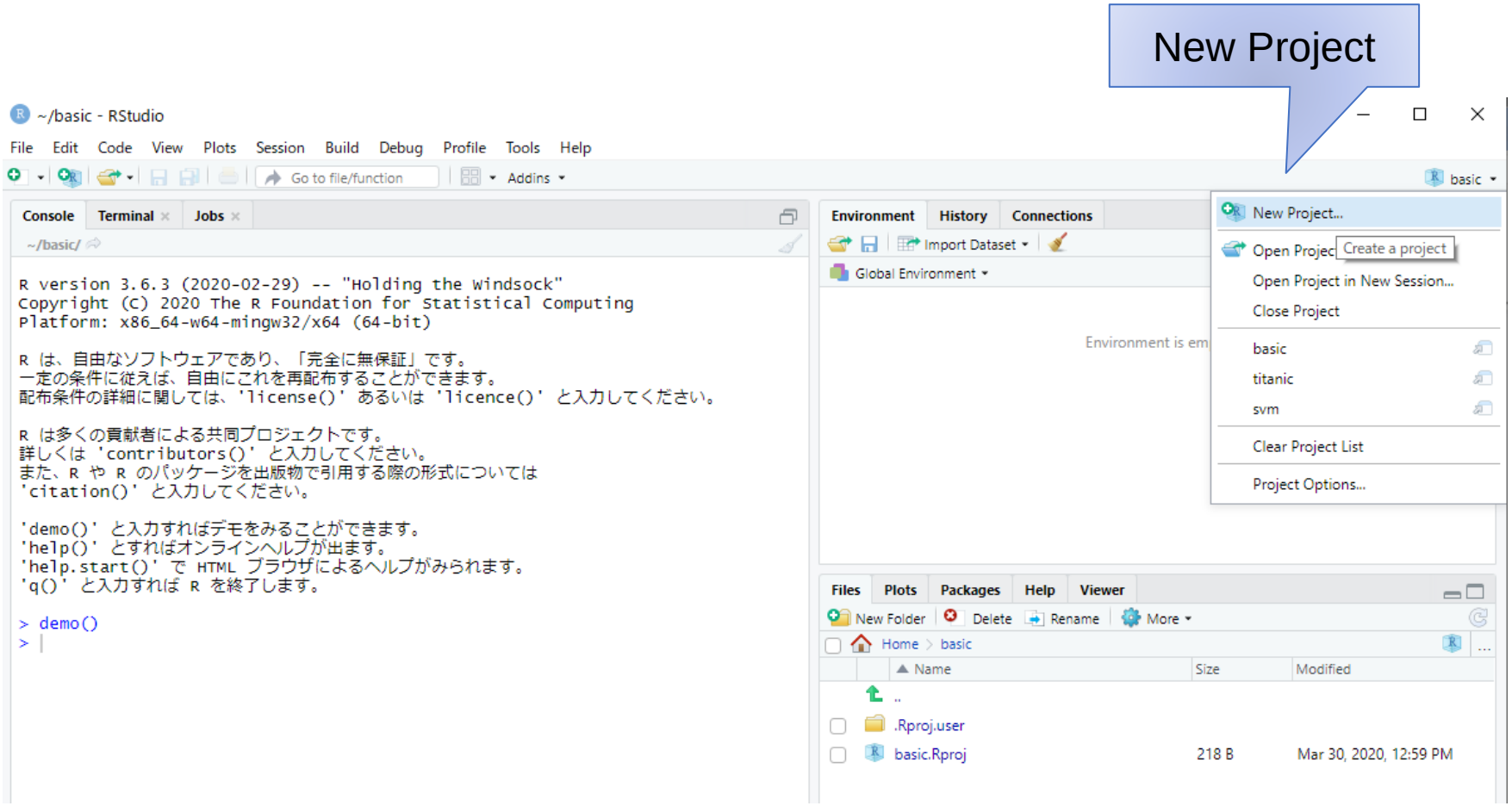
https://rstudio.com/にアクセスし、DOWNLOADからRStudio Desktop (Free)を選択

- Windowsの場合はDOWNLOAD RSTUDIO FOR WINDOWSからexeファイルをダウンロード・実行してインストール

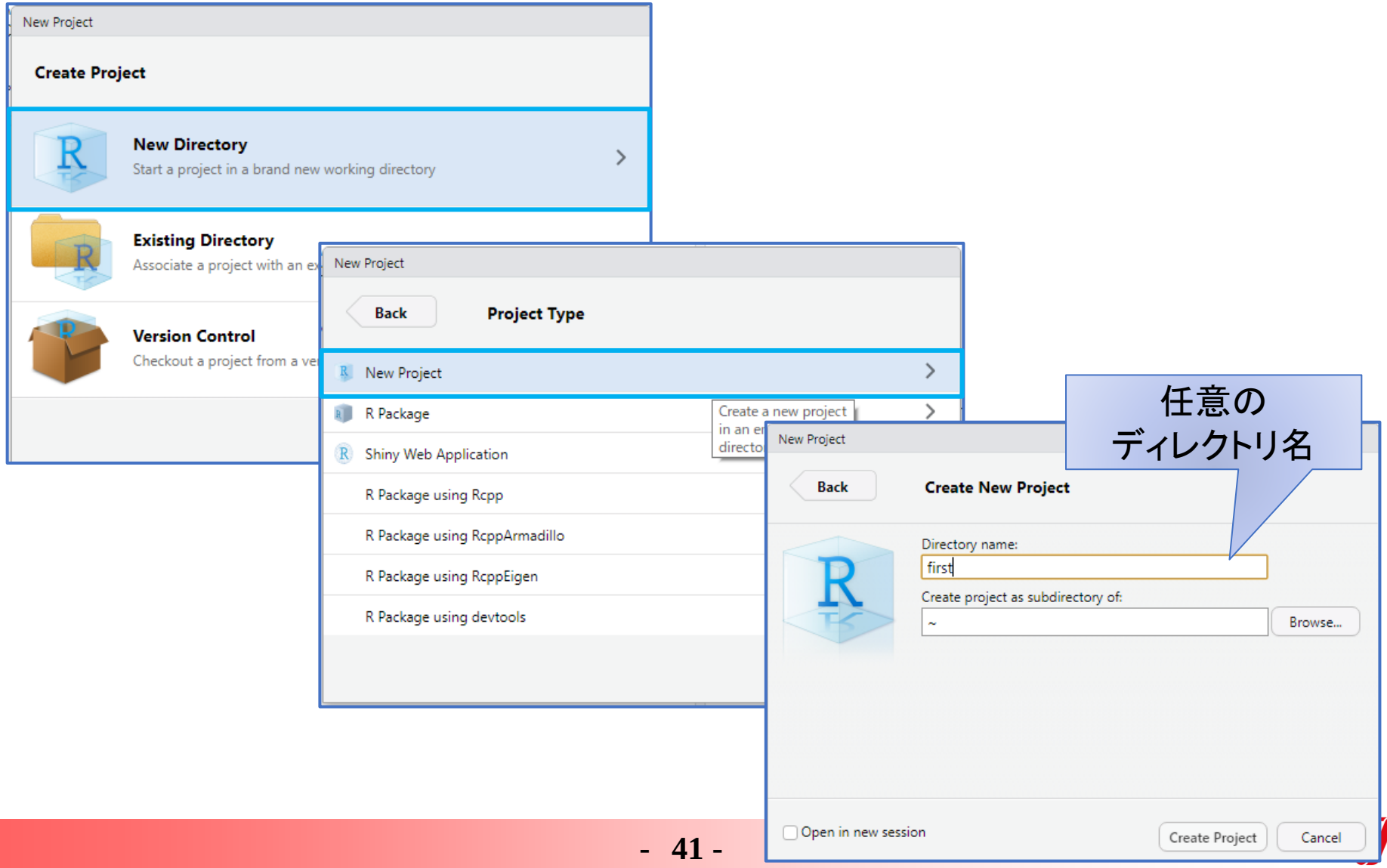
 をクリックしてR Studioを起動

プロジェクトの概念：データやRのコードをまとめて管理する

R Studio起動後、右上からNew Project



New Directory ⇒ New Project ⇒ 任意のディレクトリ名

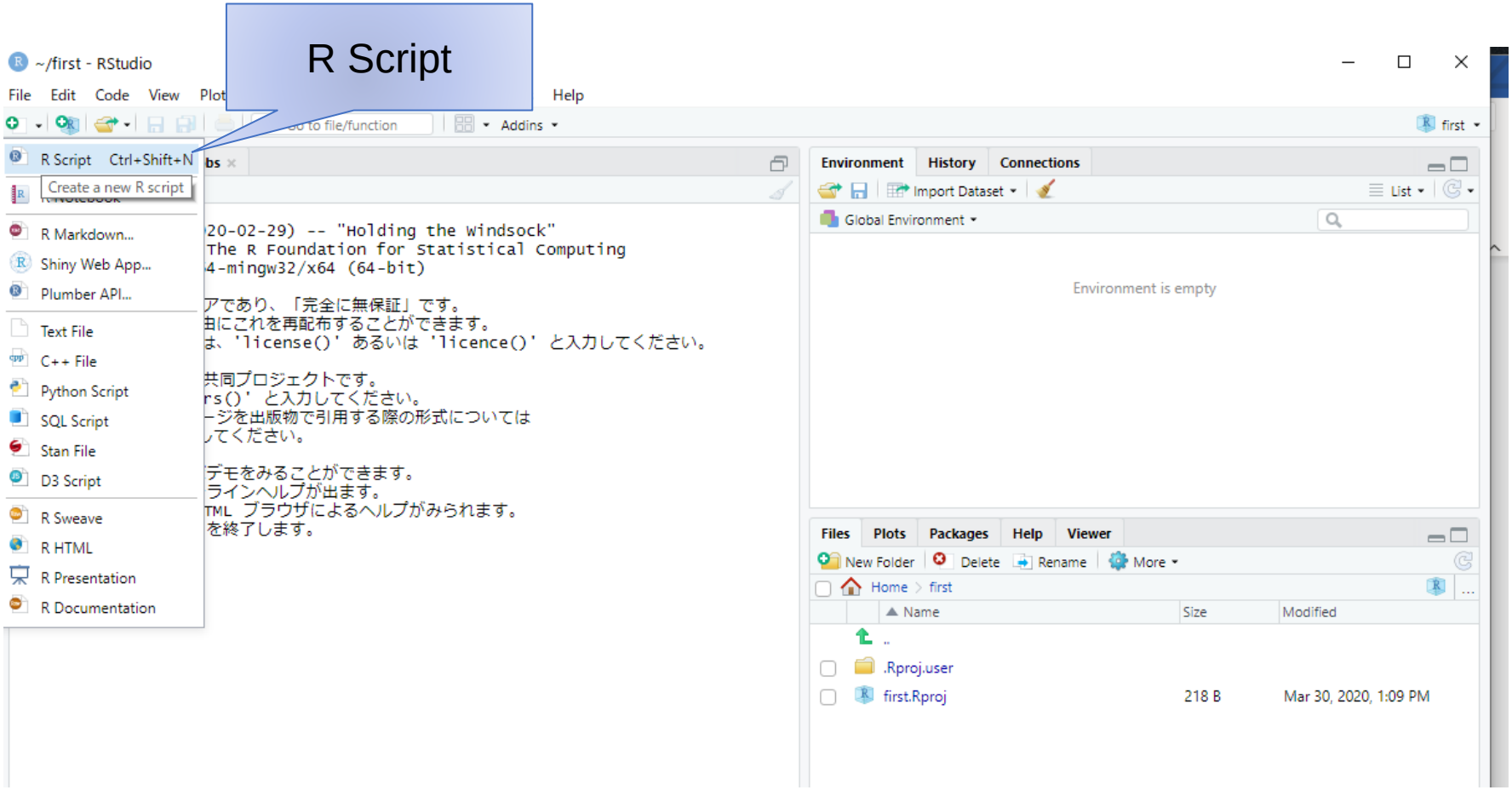


R Studioのペインと概要

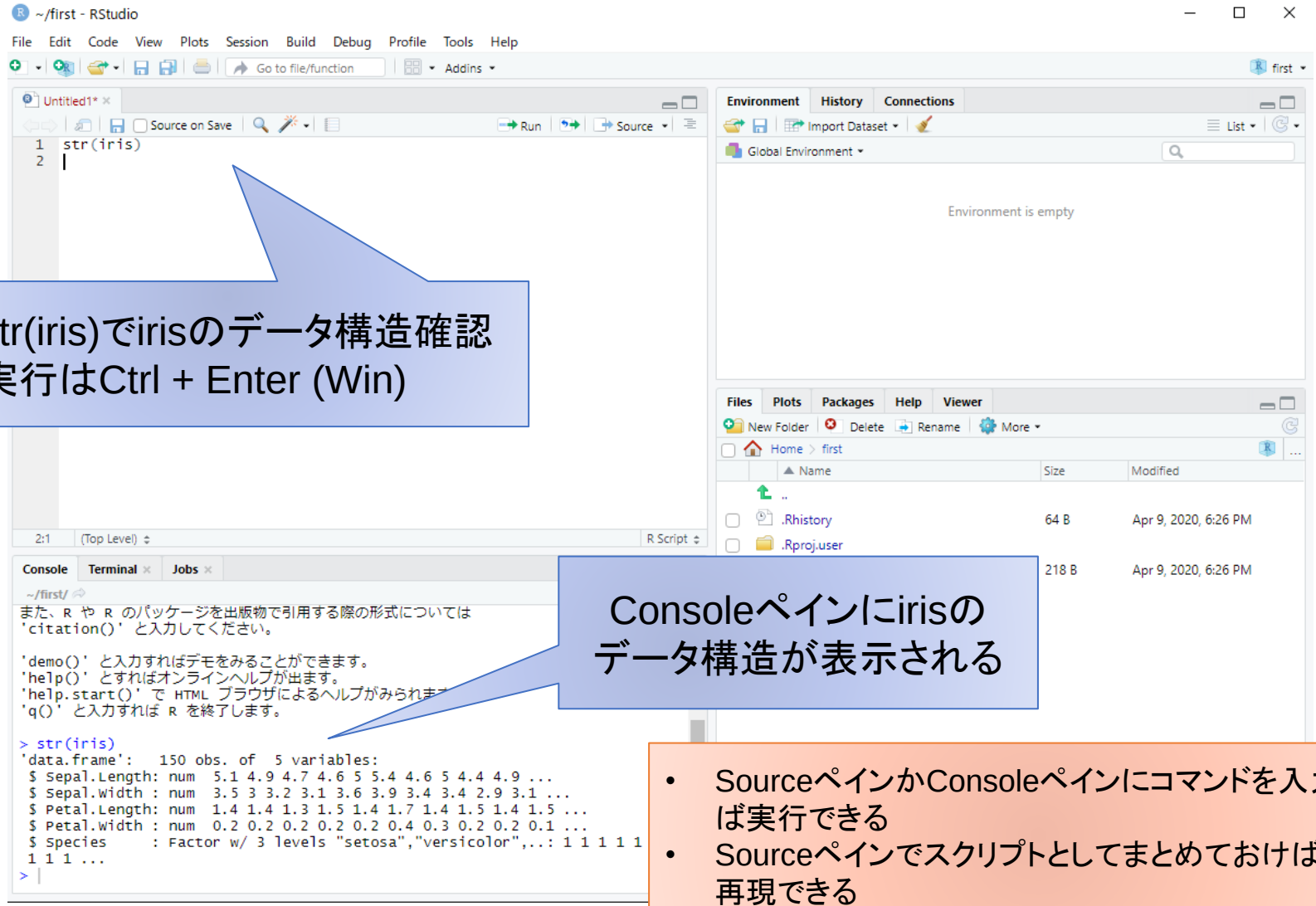
SourceペインかConsoleペインに
コマンド入力すれば実行できる

ペイン名	ショートカット	概要
Source	Ctrl + 1	スクリプトやマークダウンの編集・表示
Console	Ctrl + 2	Sourceペインに書かれたコマンドの実行。直接コマンドを入力して実行
Terminal	Shift + Alt + T	R studio内でターミナル操作するためのペイン
Help	Ctrl + 3	関数やパッケージのヘルプ
History	Ctrl + 4	コマンド履歴
Files	Ctrl + 5	ワーキングディレクトリにあるファイルを表示
Plots	Ctrl + 6	図の描画
Packages	Ctrl + 7	インストールされているパッケージの表示
Environment	Ctrl + 8	読み込んだデータセットや作成した変数など、現在のワークスペースにある環境を表示

左上からR Scriptを選択



ソースペインにコマンド入力

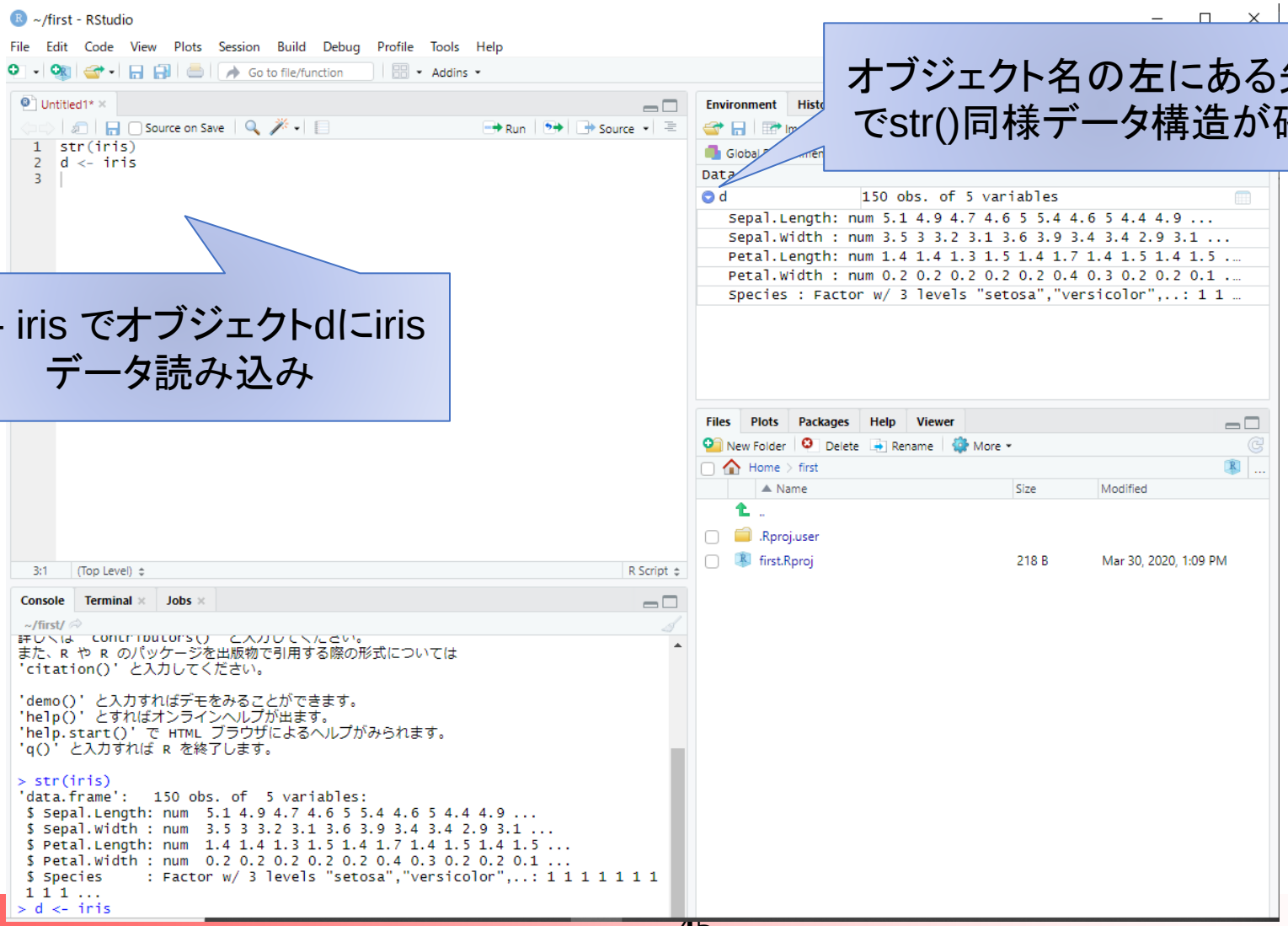


- str(iris)でirisのデータ構造確認
- 実行はCtrl + Enter (Win)

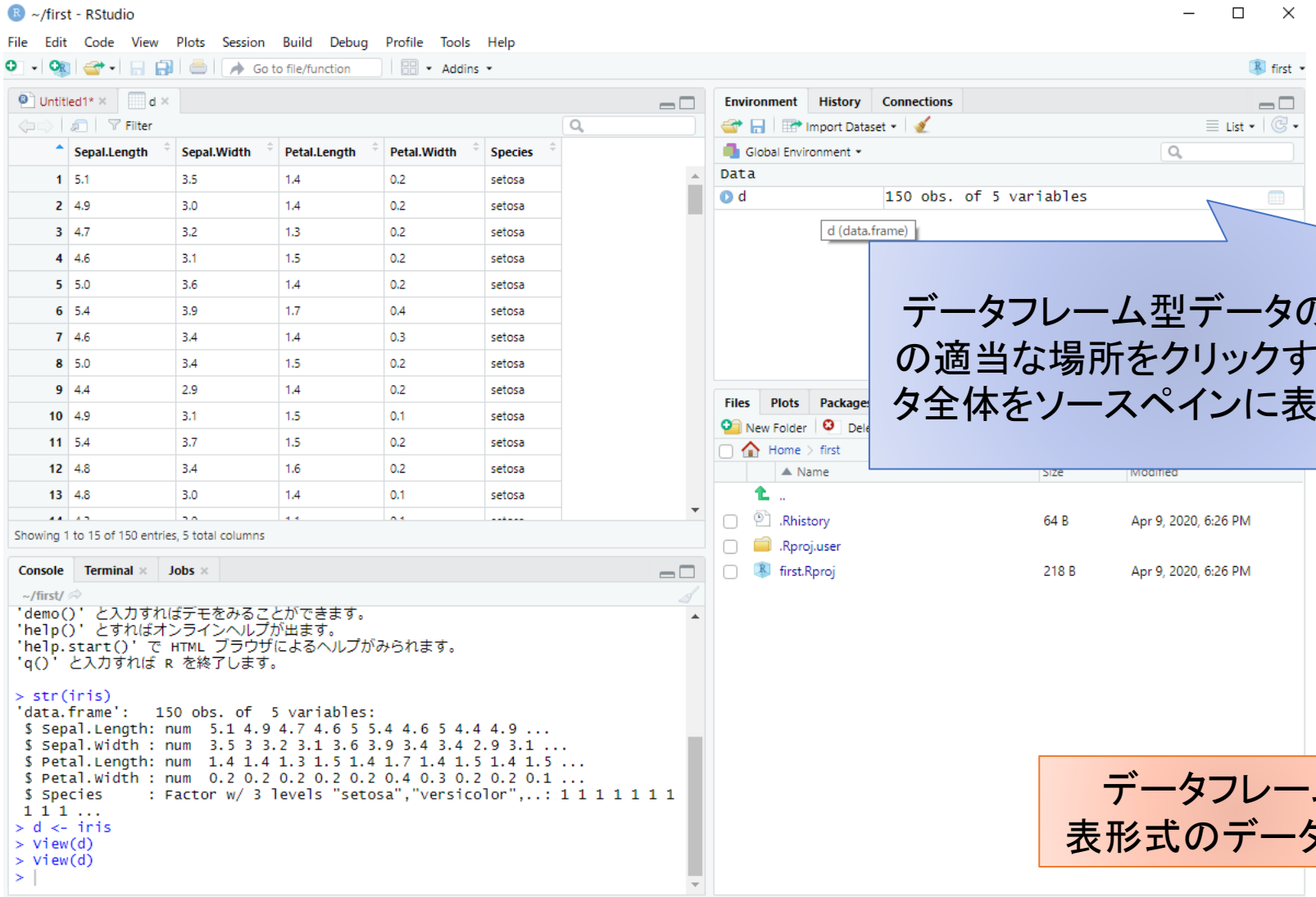
Consoleペインにirisのデータ構造が表示される

- SourceペインかConsoleペインにコマンドを入力すれば実行できる
- Sourceペインでスクリプトとしてまとめておけば保存・再現できる

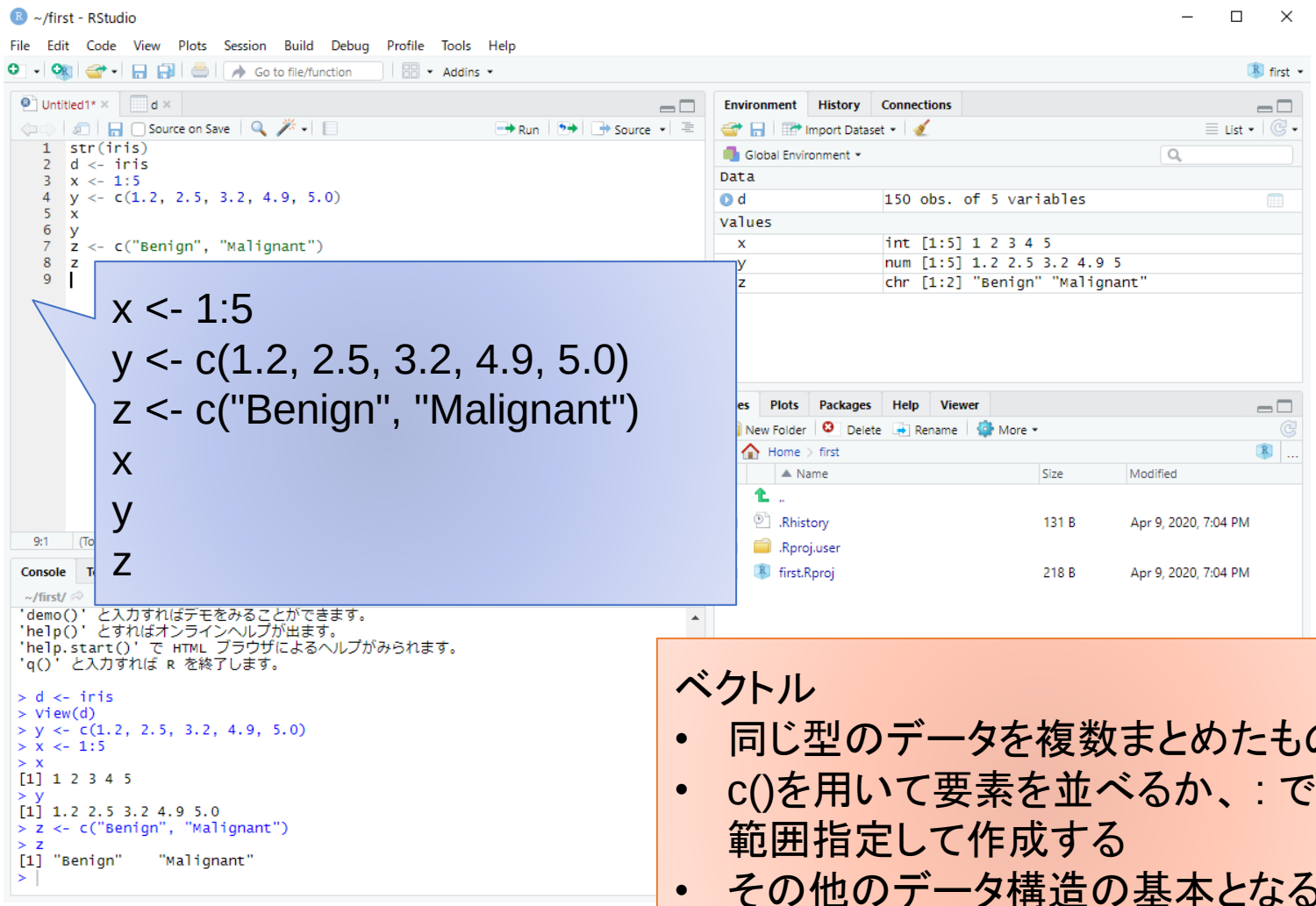
データ読込み・データ構造確認



データフレーム型データの表示



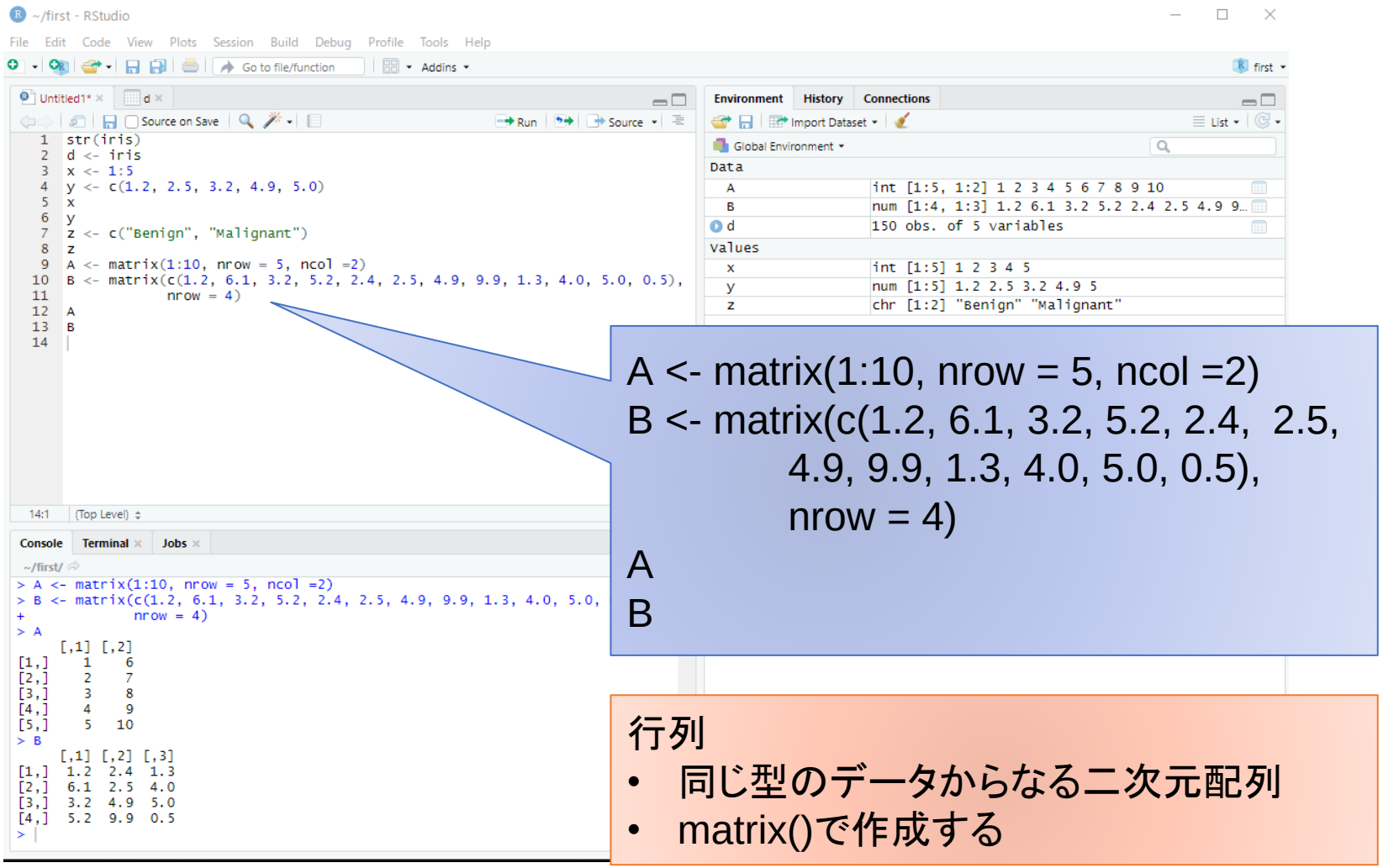
データ構造：ベクトル



ベクトル

- 同じ型のデータを複数まとめたもの
- c()を用いて要素を並べるか、：で数値範囲指定して作成する
- その他のデータ構造の基本となる

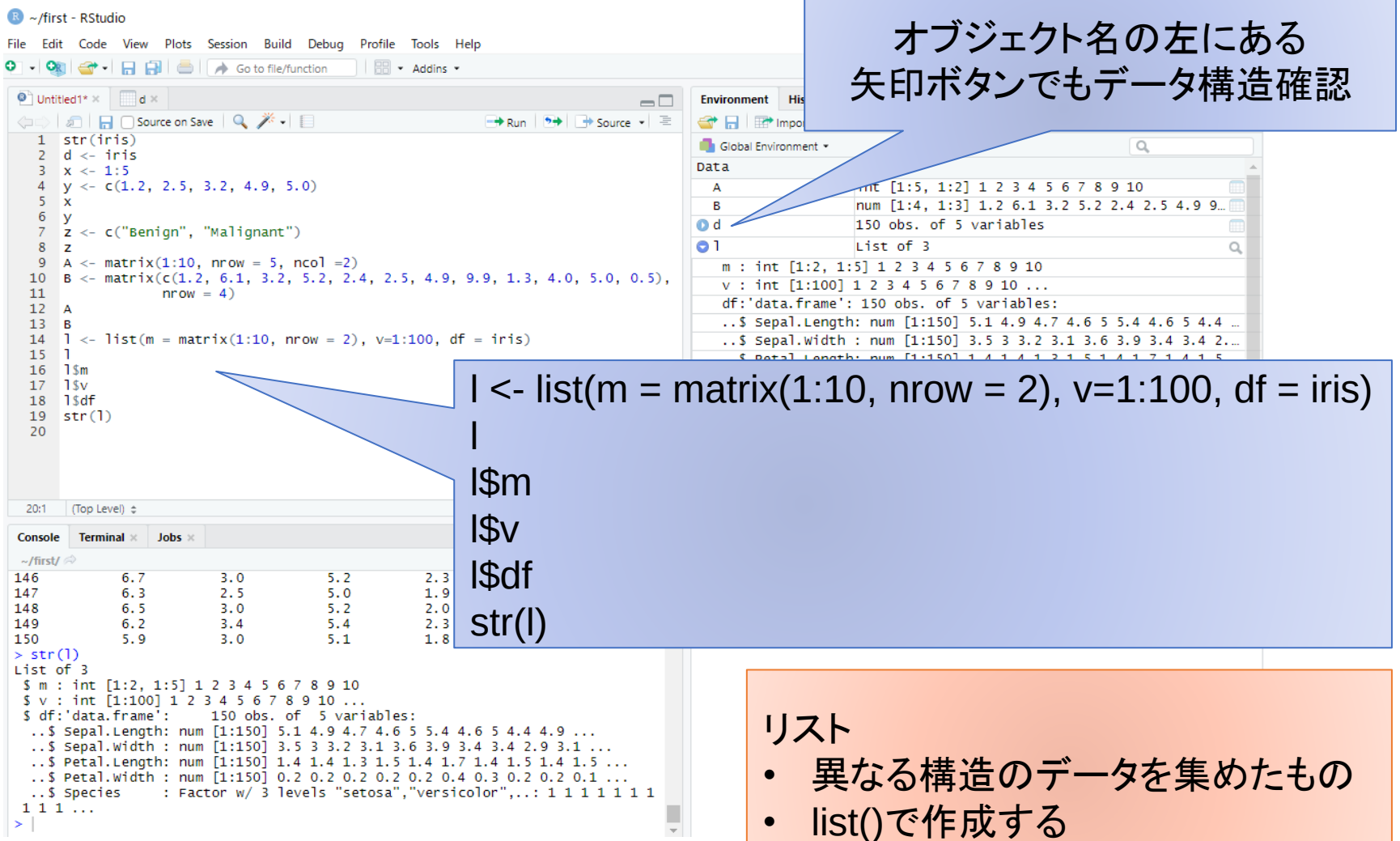
データ構造：行列




```
A <- matrix(1:10, nrow = 5, ncol =2)
B <- matrix(c(1.2, 6.1, 3.2, 5.2, 2.4,  2.5,
              4.9, 9.9, 1.3, 4.0, 5.0, 0.5),
              nrow = 4)
A
B
```

行列
- 同じ型のデータからなる二次元配列
- matrix()で作成する

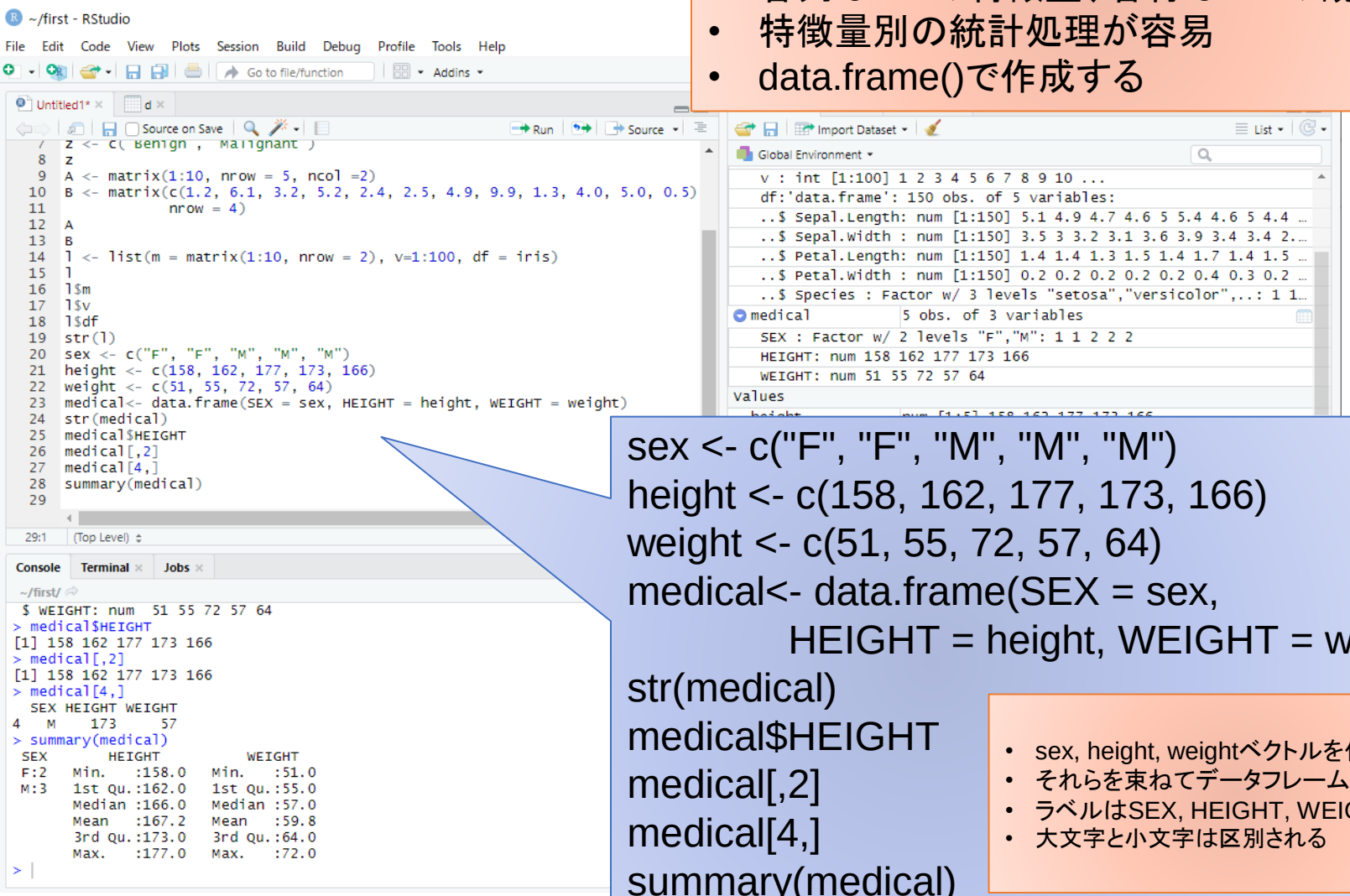
データ構造:リスト



リスト
- 異なる構造のデータを集めたもの
- list()で作成する

データフレーム

データフレーム
- リストの一種
- 各列は1つの特徴量、各行は1つの観測値
- 特徴量別の統計処理が容易
- data.frame()で作成する




```
sex <- c("F", "F", "M", "M", "M")
height <- c(158, 162, 177, 173, 166)
weight <- c(51, 55, 72, 57, 64)
medical<- data.frame(SEX = sex,
                     HEIGHT = height, WEIGHT = weight)
str(medical)
medical$HEIGHT
medical[,2]
medical[4,]
summary(medical)
```

- sex, height, weightベクトルを作成
- それらを束ねてデータフレーム作成
- ラベルはSEX, HEIGHT, WEIGHTとした
- 大文字と小文字は区別される

教師あり学習（分類）のためのデータ準備

R studioで新規プロジェクトとスクリプト作成

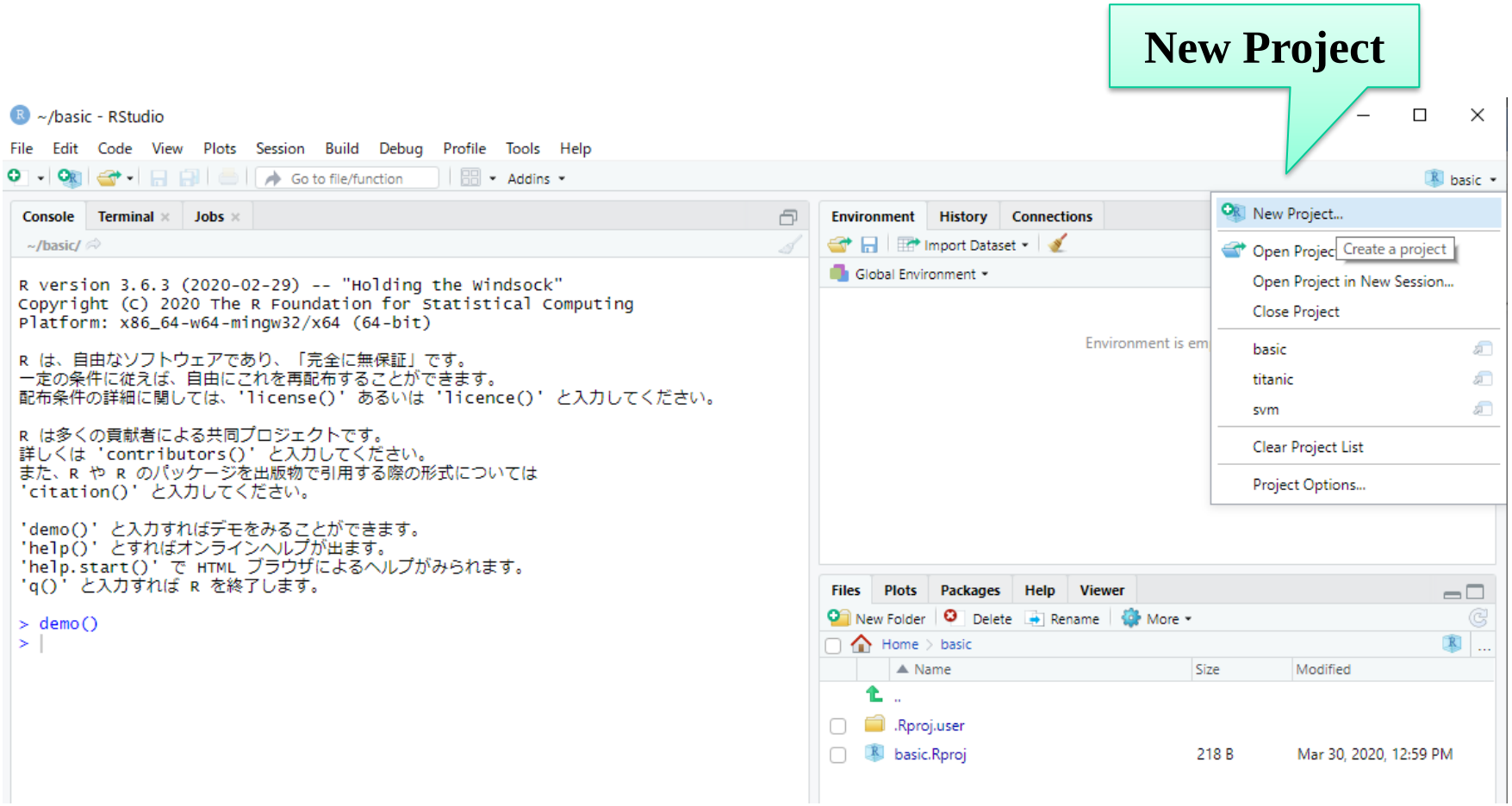
プロジェクト名は任意だが一応 wbcd とする

- Wisconsin Breast Cancer Diagnosticの意味

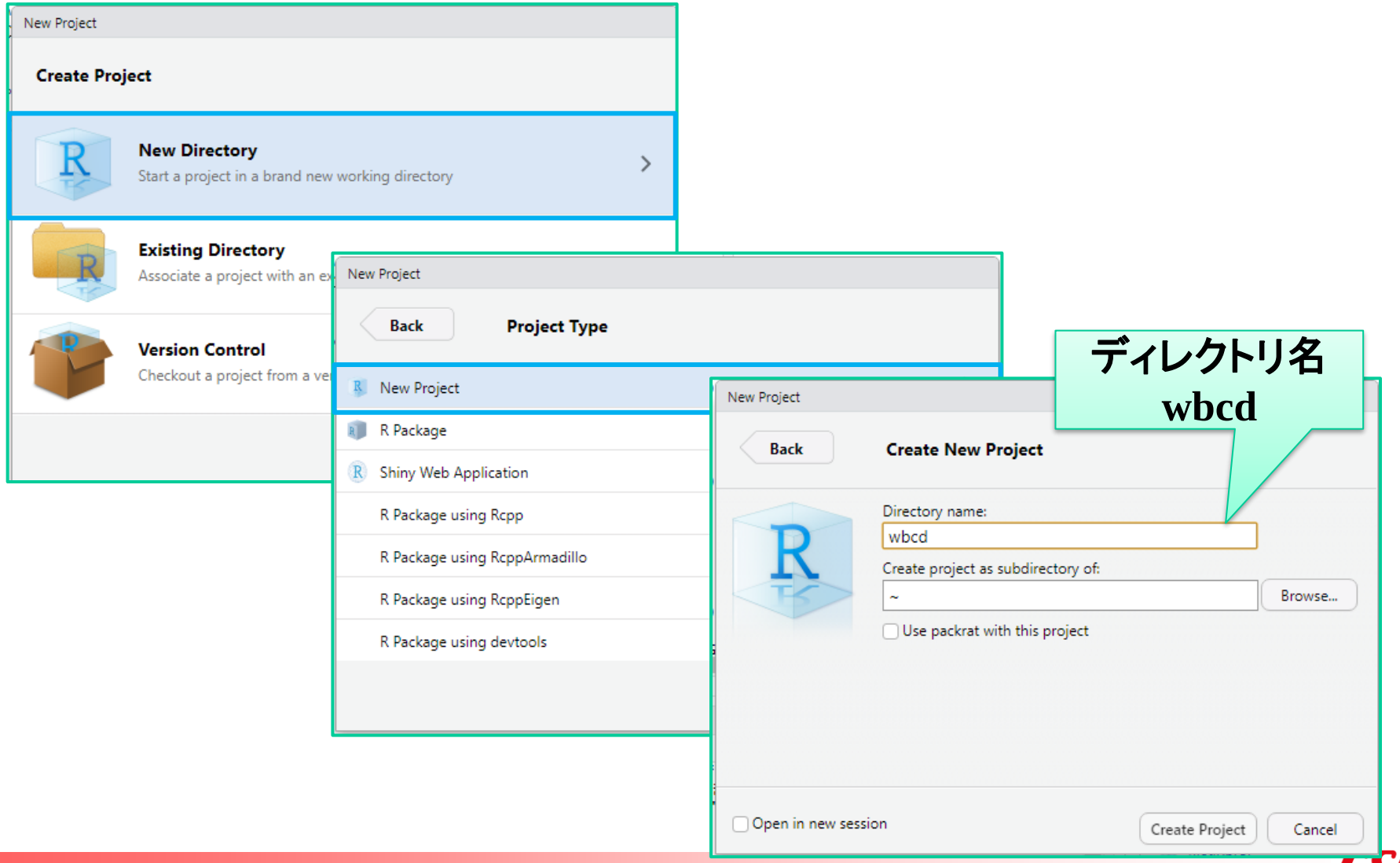
プロジェクトを立ち上げたら、新しいスクリプトを作成する

「新規プロジェクト作成」「スクリプト作成」の手順は覚えていると思うが、忘れた人は次のページを参照のこと

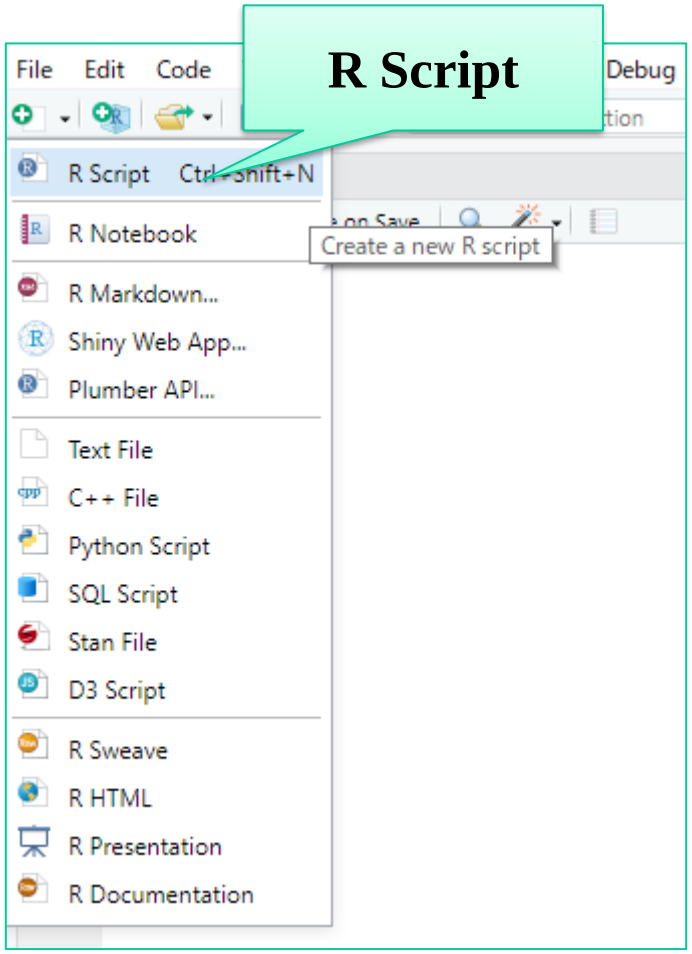
R Studio起動後、右上からNew Project



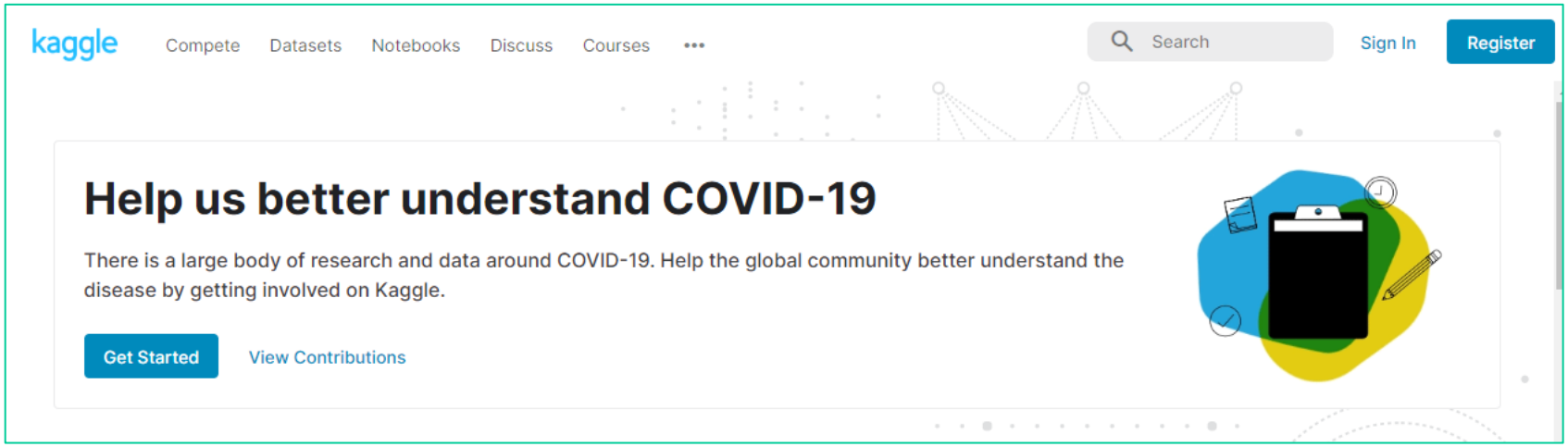
New Directory ⇒ New Project ⇒ ディレクトリ名 wbcd



左上からR Scriptを選択

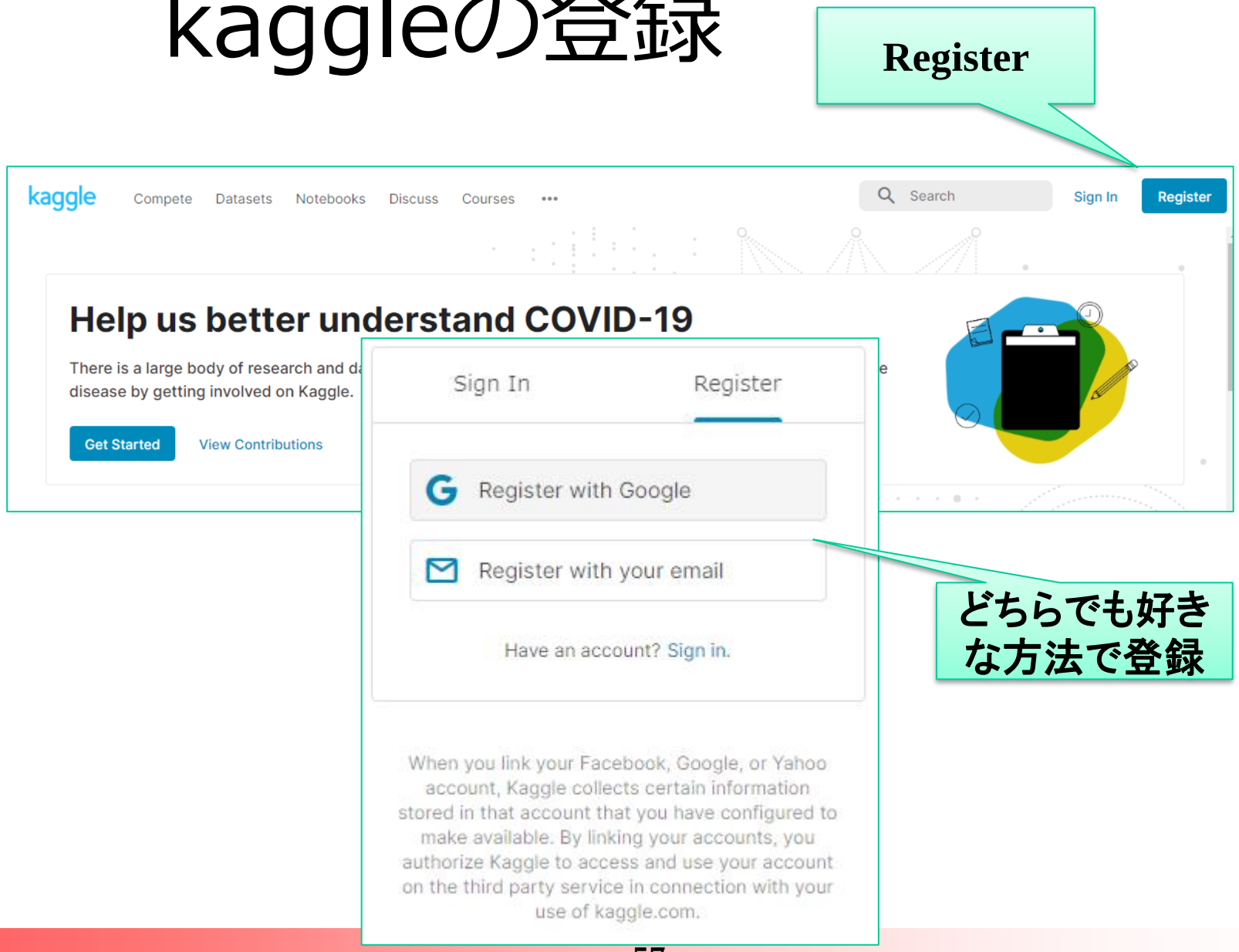


kaggleの登録

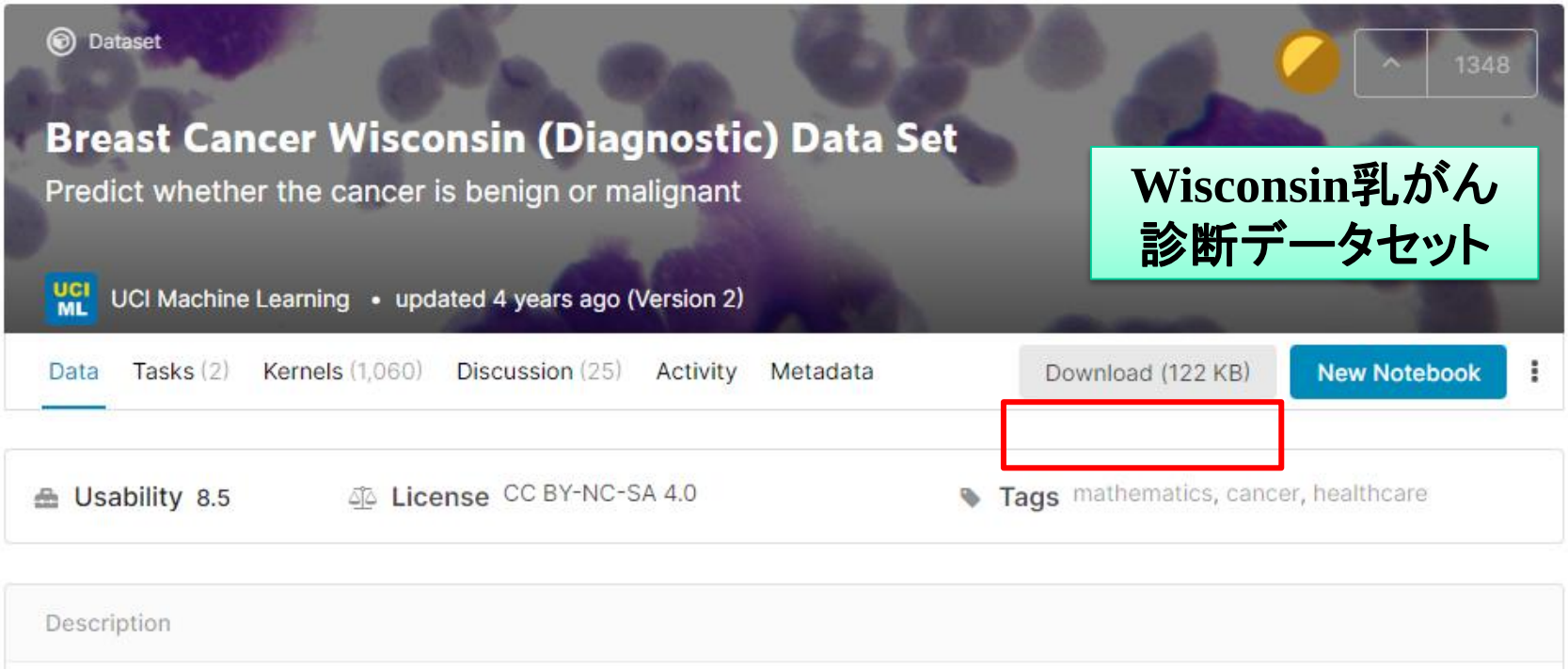


- 機械学習・データサイエンスに関わる人が集まるコミュニティ
- 企業等から実データが提供されたうえで賞金付きのコンペ（競技会）も実施される
- 提供されている実データを機械学習の学習のために利用可能
- https://www.kaggle.com/

kaggleの登録


Register
kaggle
Compete
Datasets
Notebooks
Discuss
Courses
Search
Sign In
Register
Help us better understand COVID-19
There is a large body of research and da
disease by getting involved on Kaggle.
Get Started
View Contributions
Sign In
Register
Register with Google
Register with your email
Have an account? Sign in.
When you link your Facebook, Google, or Yahoo account, Kaggle collects certain information stored in that account that you have configured to make available. By linking your accounts, you authorize Kaggle to access and use your account on the third party service in connection with your use of kaggle.com.
どちらでも好きな方法で登録

データのダウンロード



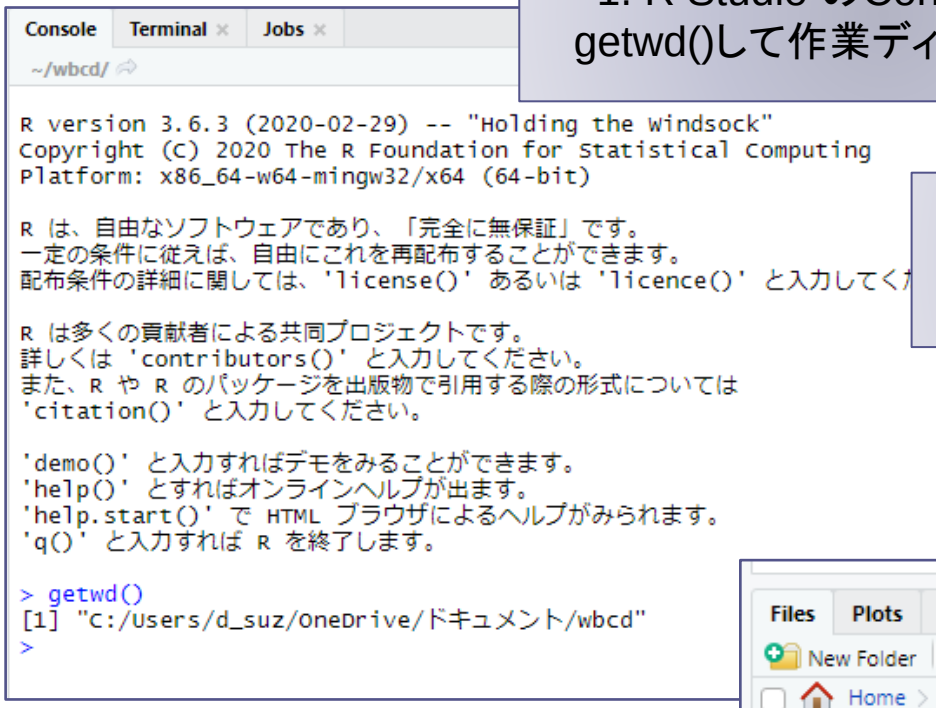
- https://www.kaggle.com/uciml/breast-cancer-wisconsin-data のDownloadから圧縮ファイル（.zip）をダウンロード
- ダウンロードしたzipファイルを展開する（zipファイルを右クリックして「すべて展開」を選択）
- 展開後のフォルダの中に"data.csv"があることを確認

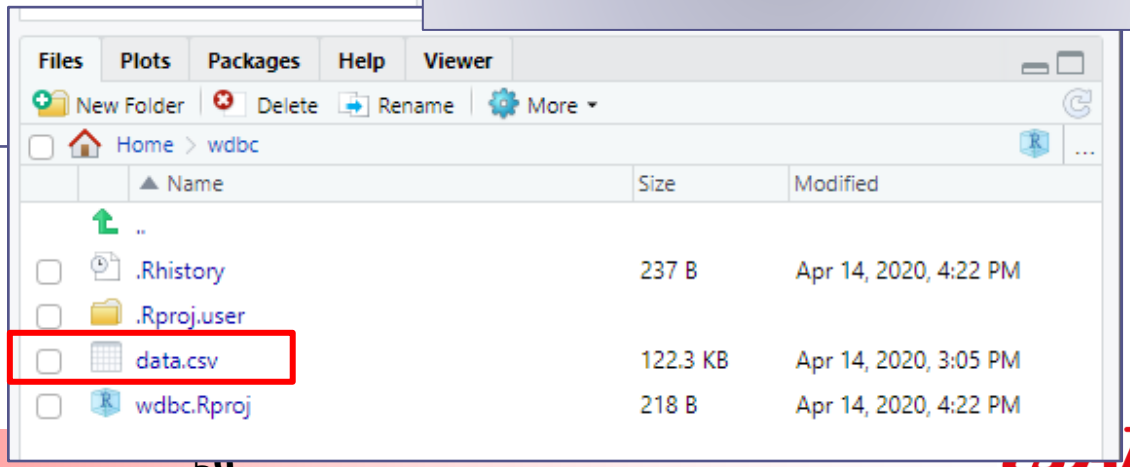
データの準備

1. R Studio のConsoleペインで getwd()して作業ディレクトリを確認

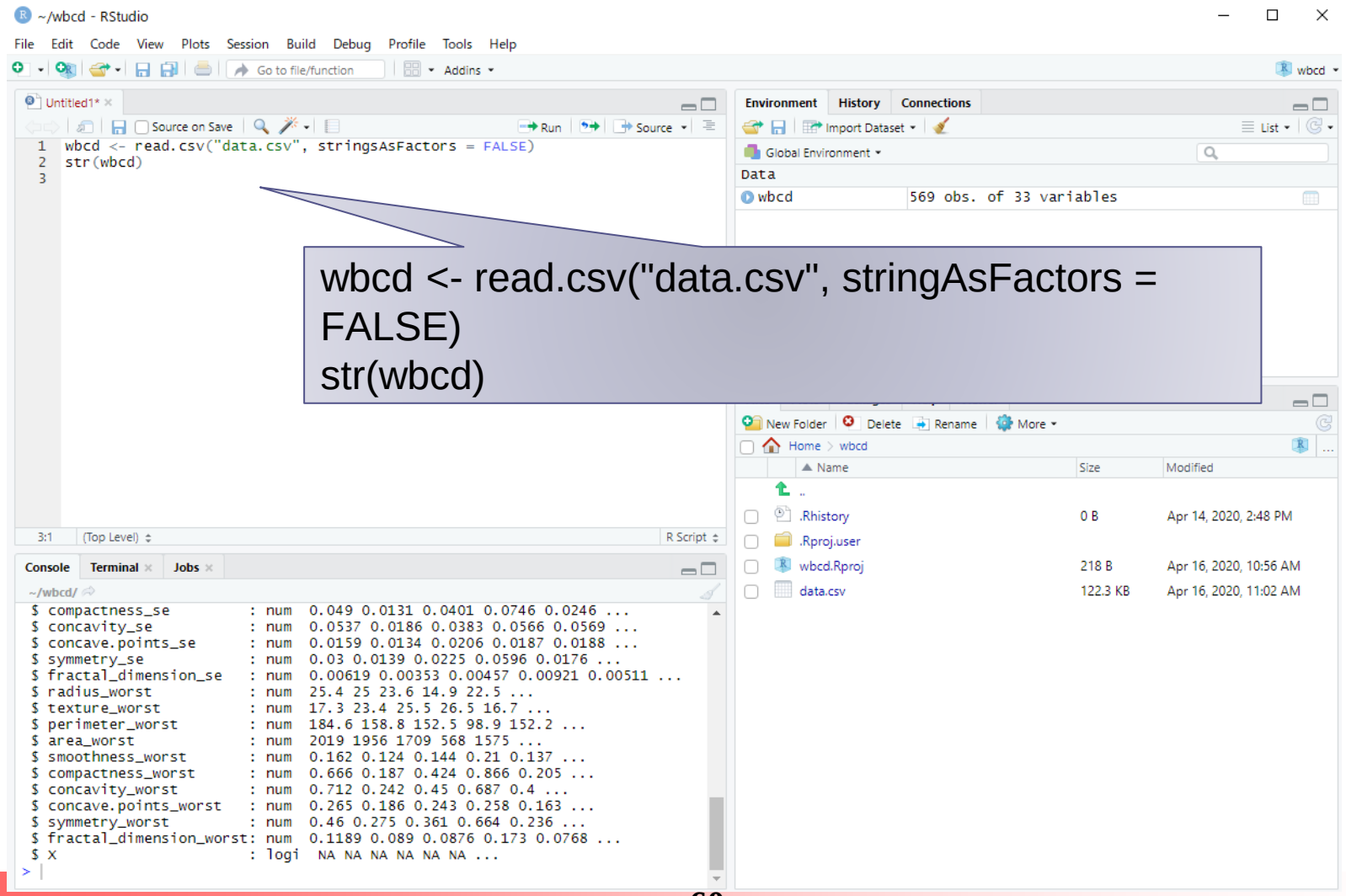
2. 先ほど用意した"data.csv"を作業ディレクトリに移動

3. R Studio のFilesペインでdata.csvの存在が確認できればOK





データの読込みとデータ構造確認

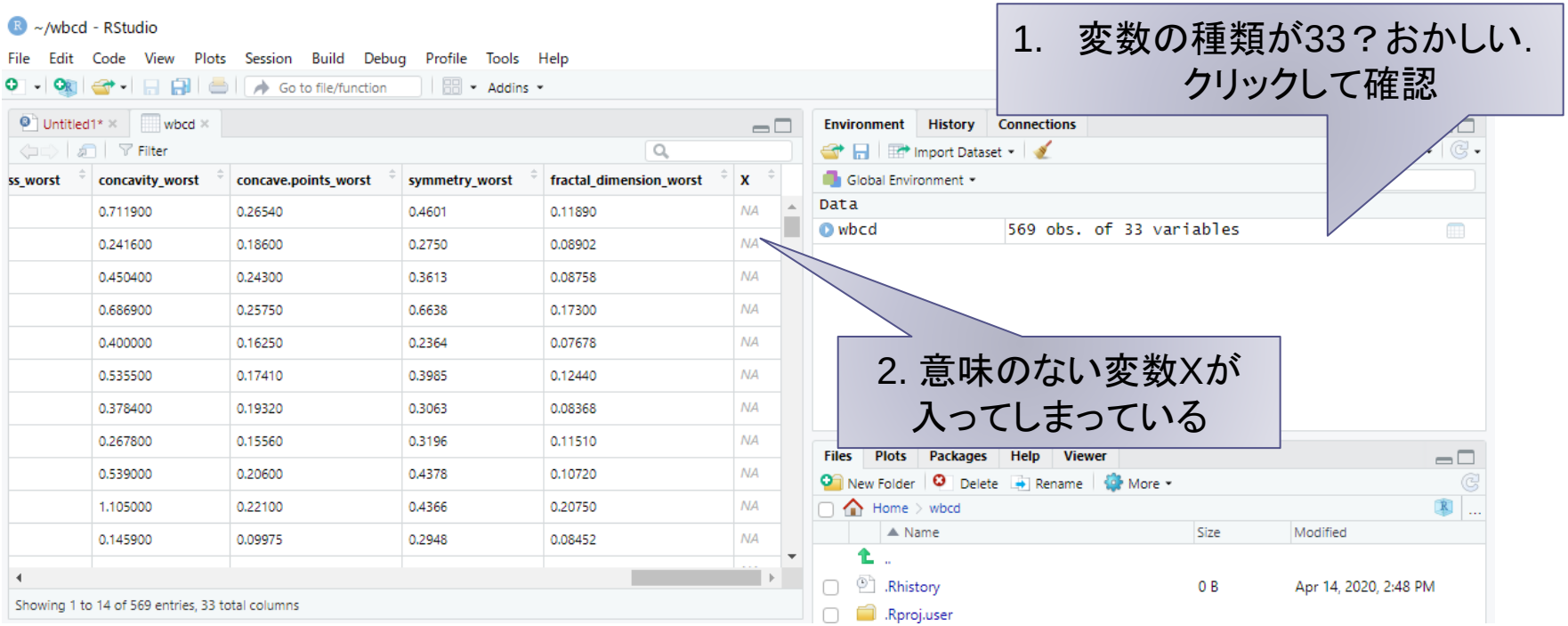


569事例、32変数
1) ID, 2) 診断結果（M=悪性、B=良性）, 3-32) 細胞核の10の特徴量各々について平均、標準偏差、最大値
欠損値なし
357 良性、212 悪性

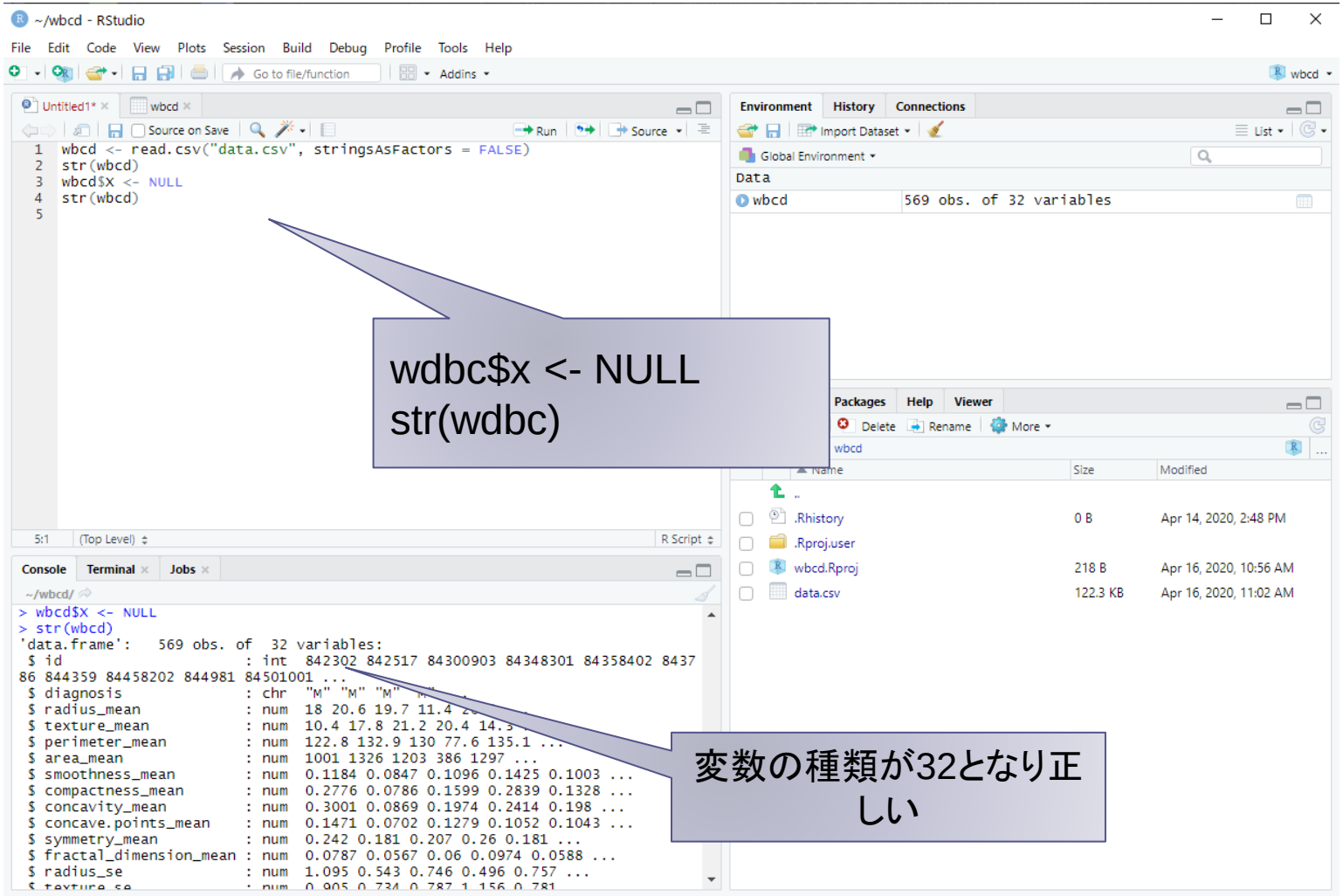
データの意味

- Number of instances: 569
- Number of attributes: 32 (ID, diagnosis, 30 real-valued input features)
- Attribute information
  - 1) ID number
  - 2) Diagnosis (M = malignant, B = benign)
  - 3-32)Ten real-valued features are computed for each cell nucleus:
    - a) radius (mean of distances from center to points on the perimeter)
    - b) texture (standard deviation of gray-scale values)
    - c) perimeter
    - d) area
    - e) smoothness (local variation in radius lengths)
    - f) compactness (perimeter^2 / area - 1.0)
    - g) concavity (severity of concave portions of the contour)
    - h) concave points (number of concave portions of the contour)
    - i) symmetry
    - j) fractal dimension ("coastline approximation" - 1)
    - The mean, standard error, and "worst" or largest (mean of the three largest values) of these features were computed for each image, resulting in 30 features. For instance, field 3 is Mean Radius, field13 is Radius SE, field 23 is Worst Radius.
    - All feature values are recoded with four significant digits.
- Missing attribute values: none
- Class distribution: 357 benign, 212 malignant

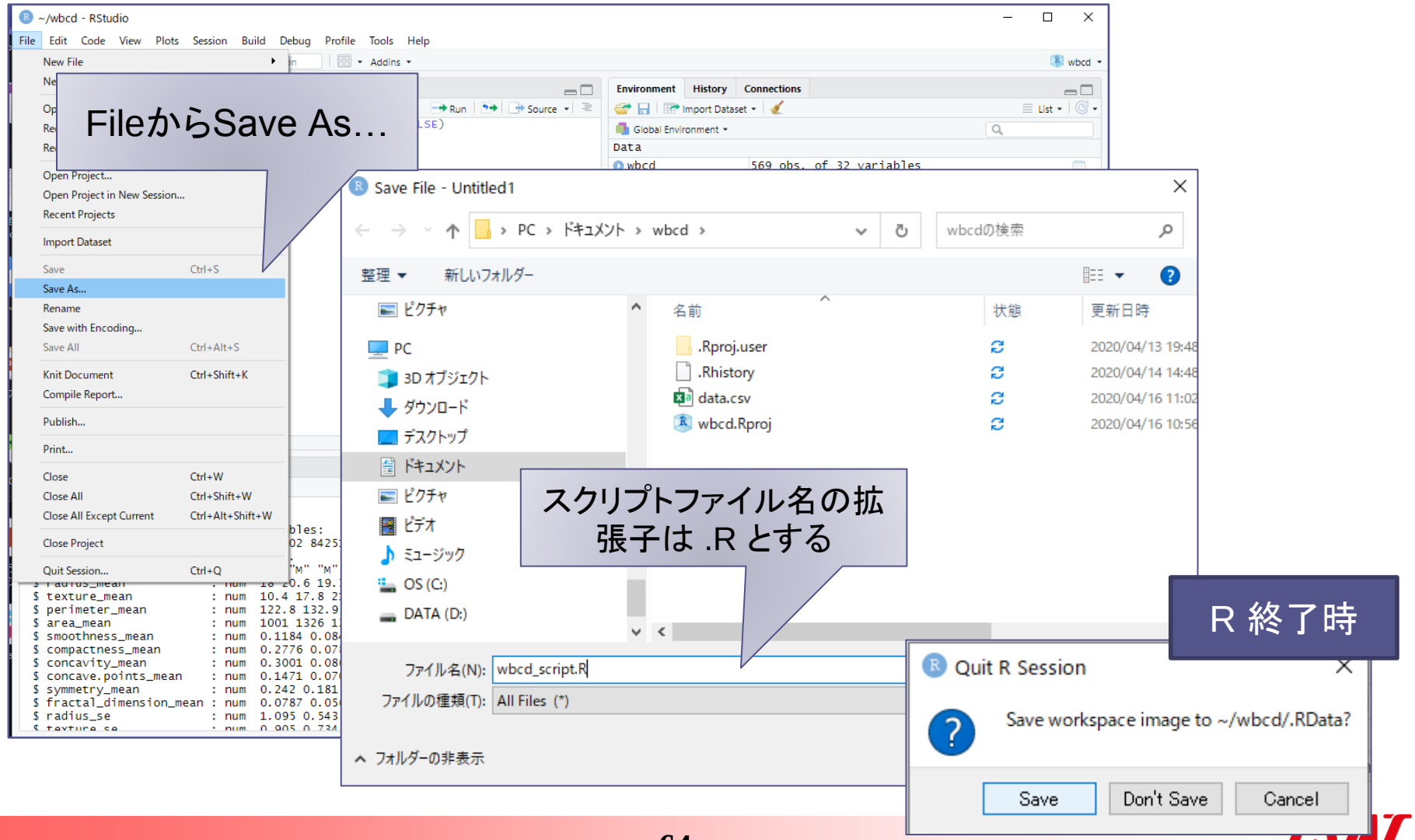
データ確認と不要な変数の発見



不要な変数xの削除


~/wbcd - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1*
wbcd
Source on Save
Run
Source
1 wbcd <- read.csv("data.csv", stringsAsFactors = FALSE)
2 str(wbcd)
3 wbcd$X <- NULL
4 str(wbcd)
5
wdbc$x <- NULL
str(wdbc)
Environment
History
Connections
Import Dataset
List
Global Environment
Data
wbcd
569 obs. of 32 variables
Packages
Help
Viewer
Delete
Rename
More
wbcd
Name
Size
Modified
.Rhistory
0 B
Apr 14, 2020, 2:48 PM
.Rproj.user
wbcd.Rproj
218 B
Apr 16, 2020, 10:56 AM
data.csv
122.3 KB
Apr 16, 2020, 11:02 AM
Console
Terminal
Jobs
~/wbcd/
> wbcd$X <- NULL
> str(wbcd)
'data.frame': 569 obs. of 32 variables:
$ id : int 842302 842517 84300903 84348301 84358402 8437
86 844359 84458202 844981 84501001 ...
$ diagnosis : chr "M" "M" "M"
$ radius_mean : num 18 20.6 19.7 11.4
$ texture_mean : num 10.4 17.8 21.2 20.4 14.3
$ perimeter_mean : num 122.8 132.9 130 77.6 135.1 ...
$ area_mean : num 1001 1326 1203 386 1297 ...
$ smoothness_mean : num 0.1184 0.0847 0.1096 0.1425 0.1003 ...
$ compactness_mean : num 0.2776 0.0786 0.1599 0.2839 0.1328 ...
$ concavity_mean : num 0.3001 0.0869 0.1974 0.2414 0.198 ...
$ concave.points_mean : num 0.1471 0.0702 0.1279 0.1052 0.1043 ...
$ symmetry_mean : num 0.242 0.181 0.207 0.26 0.181 ...
$ fractal_dimension_mean : num 0.0787 0.0567 0.06 0.0974 0.0588 ...
$ radius_se : num 1.095 0.543 0.746 0.496 0.757 ...
変数の種類が32となり正しい

スクリプトの保存・ワークスペースの保存



教師あり学習（回帰）

教師あり学習

- 正解の用意された問題集（教師データ）がある
- 教師は正解の導き方を詳しく説明してくれない
- 学習者（コンピュータ）は問題と正解の関係を推測する
- これまで見たことの無い新しい類似の問題に対して正解を導けるようになることを目指す

- 教師あり学習の概要
  - 入力とそれに対応する正解出力の組からなる教師データを用いる
  - 教師データに基づいて入力と出力の関係を表現するモデルを構築する
  - 新しい入力に対して精度の高い予測出力ができることを目指す
  - 出力が連続値の場合「回帰」モデル、出力がカテゴリの場合「分類」モデルを構築する

- ex. 不動産価格の予測（回帰）、検査結果に基づく陽性判定（分類）

回帰（Regression）

教師あり学習

入力データから連続値を予測する

- ex. 都市の電力消費量や住宅価格など

回帰のアルゴリズム

- 線形回帰（通常最小二乗法）
- Ridge回帰、Lasso回帰 、Elastic Net
- SVR(Support Vector Regression)

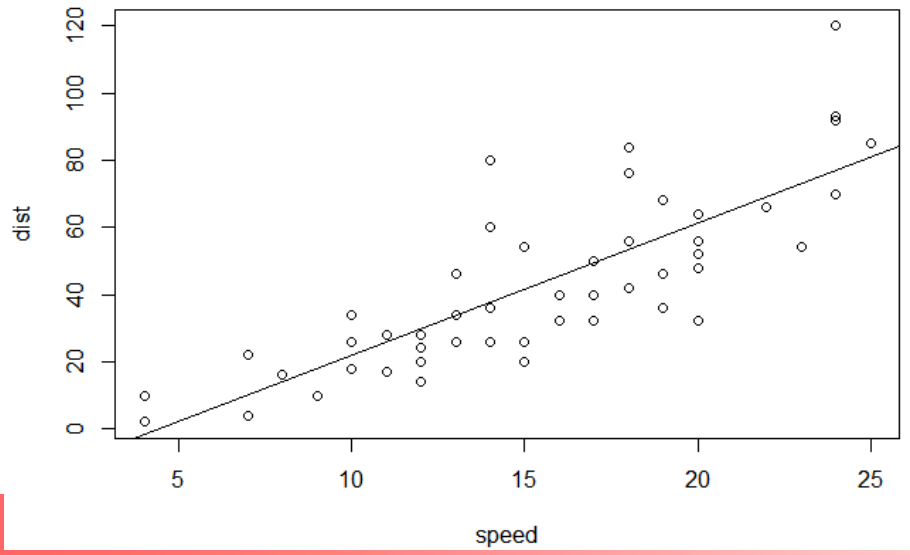
線形回帰

教師あり学習（回帰）

線形回帰の概要と例

説明変数の線型結合で目的変数を表現するモデル

- 単回帰
  - 説明変数が1つ
  - 目的変数 y , 説明変数 x として$y = ax + b$ と表現される. aは回帰係数, bは切片
- 重回帰
  - 説明変数が2つ以上
  - 目的変数 y , 説明変数 $x_1, x_2, \cdots, x_d$ として$y = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_d x_d$ と表現される. $w_1, w_2, \cdots, w_d$は偏回帰係数, w_0は切片



単回帰の例

- R標準データセットcars
- 車速(mph)と停止距離(ft)の組50組
- 車速を説明変数、停止距離を目的変数として単回帰
- 回帰式$y = 3.9\,x\, - 17.6$
- 回帰式は最小二乗法により求まる

※実際は、空走距離は車速に比例するが、制動距離は速度の二乗に比例する

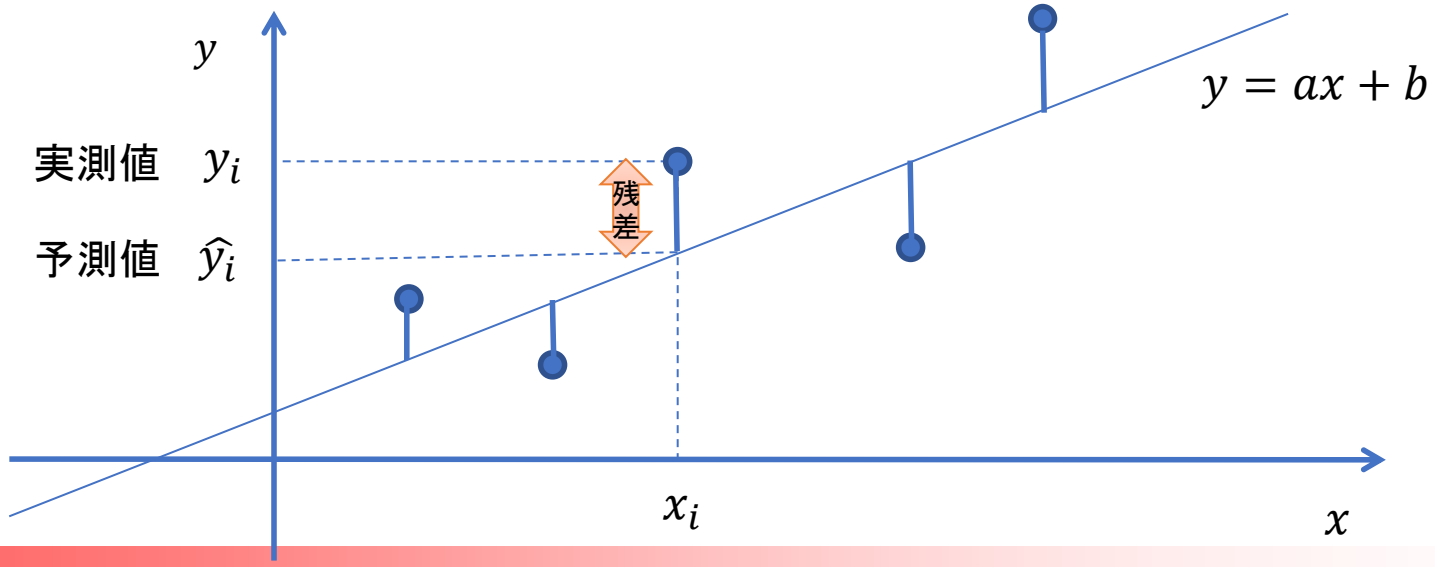
最小二乗法による回帰係数の導出(1/2)

予測値と実測値の差（残差）の平方和を最小にするよう回帰係数を決定する方法

ex. 単回帰式

- データセット $(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)$ に基づき回帰式 $y = ax + b$ を導出する
- 予測値を $\hat{y}_i = ax_i + b$ と書くと、残差平方和は以下の式であらわされる

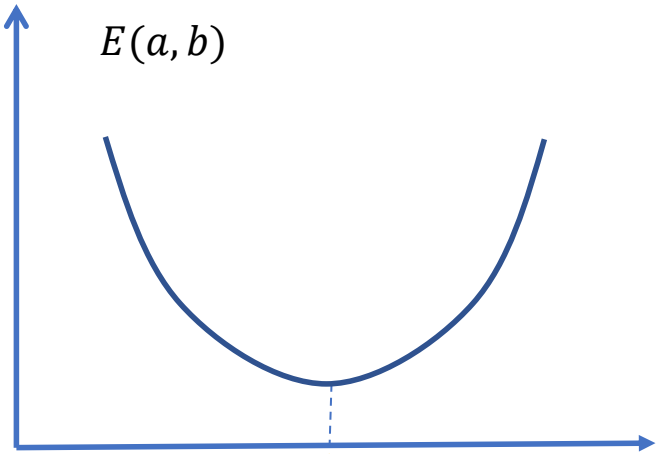
$$E = \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 = \sum_{i=1}^{n} (ax_i + b - y_i)^2$$



最小二乗法による回帰係数の導出(2/2)

$$E = \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 = \sum_{i=1}^{n} (ax_i + b - y_i)^2$$

Eはa, bの二変数関数$E(a, b)$と見ることができ、下に凸の二次関数となる



$\frac{\partial E}{\partial a} = 0, \frac{\partial E}{\partial b} = 0$ なる (a, b)

$\frac{\partial E}{\partial a} = 0, \frac{\partial E}{\partial b} = 0$ でEは最小となるため、

$$\frac{\partial E}{\partial a} = \sum 2(ax_i + b - y_i)x_i = 0$$

$$\frac{\partial E}{\partial b} = \sum 2(ax_i + b - y_i) = 0$$

連立方程式を解くと、$\bar{x} = \frac{1}{n}\sum x_i$, $\bar{y} = \frac{1}{n}\sum y_i$として

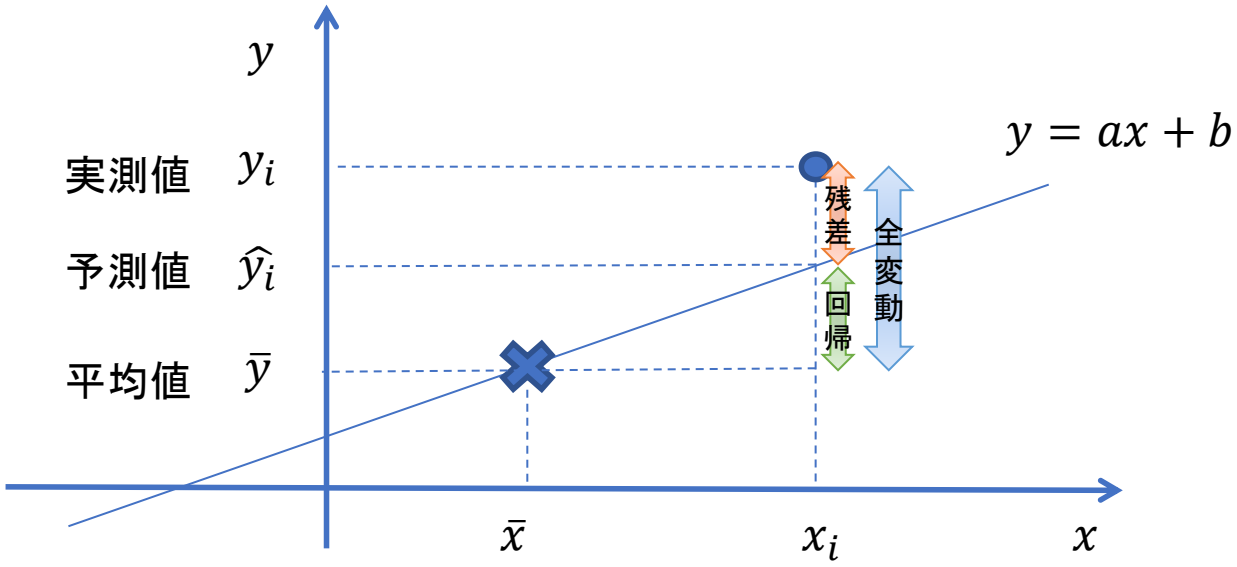
$$a = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\frac{1}{n}\sum_{i=1}^{n} x_i^2 - \bar{x}^2} = \frac{\sigma_{xy}}{\sigma_x^2},$$

$$\mathrm{b} = \bar{\mathrm{y}} - a\bar{x}$$

σ_x^2: xの分散
σ_{xy}: xとyの共分散

決定係数

回帰式が全体の変動のうちどの程度を説明できるか示す係数



- 全変動平方和 = 回帰変動平方和 + 残差変動平方和

$$\sum(y_i - \bar{y})^2 = \sum(\hat{y}_i - \bar{y})^2 + \sum(y_i - \hat{y}_i)^2$$

- 決定係数＝回帰変動平方和／全変動平方和

$$R^2 = \frac{\sum(\hat{y}_i - \bar{y})^2}{\sum(y_i - \bar{y})^2} = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}$$

どの残差も0ならば
決定係数は1になる

重回帰の場合の最小二乗法による回帰係数の導出(1/3)

重回帰式は

$$y = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_d x_d$$

$\boldsymbol{x} = (x_1, x_2, \cdots, x_d)^T$ は入力変数、$w_0, w_1, \cdots, w_d$はパラメータ

ここで $\tilde{\boldsymbol{x}} = (1, x_1, x_2, \cdots, x_d)^T, \boldsymbol{w} = (w_0, w_1, \cdots, w_d)^T$ と定義すると

重回帰式は

$$y = \boldsymbol{w}^T \tilde{\boldsymbol{x}}$$

とあらわせる

重回帰の場合の最小二乗法による回帰係数の導出 (2/3)

重回帰式は

$$y = \boldsymbol{w}^T \tilde{\boldsymbol{x}}$$

データ行列

$$\boldsymbol{X} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{pmatrix}$$

をもとに

$$\tilde{\boldsymbol{X}} = \begin{pmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{nd} \end{pmatrix}$$

を定義すると、各サンプル行（$1 \sim n$）に対して回帰式をあてはめた結果は

$$\begin{pmatrix} w_0 + w_1 x_{11} + & \cdots & + w_d x_{1d} \\ w_0 + w_1 x_{21} + & \ldots & + w_d x_{2d} \\ \vdots & \vdots & \vdots \\ w_0 + w_1 x_{n1} + & \cdots & + w_d x_{nd} \end{pmatrix} = \begin{pmatrix} \boldsymbol{w}^T \tilde{\boldsymbol{x}}_1 \\ \boldsymbol{w}^T \tilde{\boldsymbol{x}}_2 \\ \vdots \\ \boldsymbol{w}^T \tilde{\boldsymbol{x}}_n \end{pmatrix} = \tilde{\boldsymbol{X}} \boldsymbol{w}$$

とあらわすことができる

重回帰の場合の最小二乗法による回帰係数の導出(3/3)

重回帰式をwの関数とみて

$$\hat{y}(w) = \tilde{X}w$$

目的変数yとの差の二乗和を目的関数として、これを最小にするようなwを求める

$$E(w) = \left|y - \tilde{X}w\right|^2 = \left(y - \tilde{X}w\right)^{\mathrm{T}}\left(y - \tilde{X}w\right)$$
$$= y^T y - w^T \tilde{X}^T y - y^T \tilde{X}w + w^T \tilde{X}^T \tilde{X}w$$

目的関数$E(w)$の勾配を求めると

$$\nabla E(w) = -2\tilde{X}^T y + 2\tilde{X}^T \tilde{X}w$$

これを$= 0$とおくことで

$$\tilde{X}^T y = \tilde{X}^T \tilde{X}w$$

よって

$$w = \left(\tilde{X}^T \tilde{X}\right)^{-1} \tilde{X}^T y$$

データ行列と目的変数ベクトルからパラメータベクトルが決まる

線型回帰の処理手順と実装

処理手順

- 学習
  - 訓練データセット（特徴量・目的変数）を与えると、残差平方和を最小化する回帰式が得られる
- 予測
  - 回帰式にテストデータセットの特徴量を与えると目的変数の予測値が得られる

実装

- RのstatsパッケージにふくまれるIm実装など

Ridge回帰・Lasso回帰

Ridge回帰

- 最小にすべき目的関数にペナルティ$\lambda|\boldsymbol{w}|^2$ (L2ノルム)を付加

$$E(\boldsymbol{w}) = \left|\boldsymbol{y} - \tilde{\boldsymbol{X}}\boldsymbol{w}\right|^2 + \lambda|\boldsymbol{w}|^2$$

- 係数があまりに大きくならないようにしている

Lasso回帰

- 最小にすべき目的関数にペナルティ $\lambda|w|_1$（L1ノルム）を付加

$$E(\boldsymbol{w}) = \left|\boldsymbol{y} - \tilde{\boldsymbol{X}}\boldsymbol{w}\right|^2 + \lambda|w|_1$$

- ここで$|w|_1 = \sum_{i=1}^{d}|w_i|$
- 係数があまりに大きくならないようにしている

Ridge回帰・Lasso回帰

Ridge回帰

- 最小にすべき目的関数にペナルティ$\lambda|\boldsymbol{w}|^2$ (L2ノルム)を付加

$$E(\boldsymbol{w}) = \left|\boldsymbol{y} - \tilde{\boldsymbol{X}}\boldsymbol{w}\right|^2 + \lambda|\boldsymbol{w}|^2$$

- 係数があまりに大きくならないようにしている

Lasso回帰

- 最小にすべき目的関数にペナルティ $\lambda|w|_1$（L1ノルム）を付加

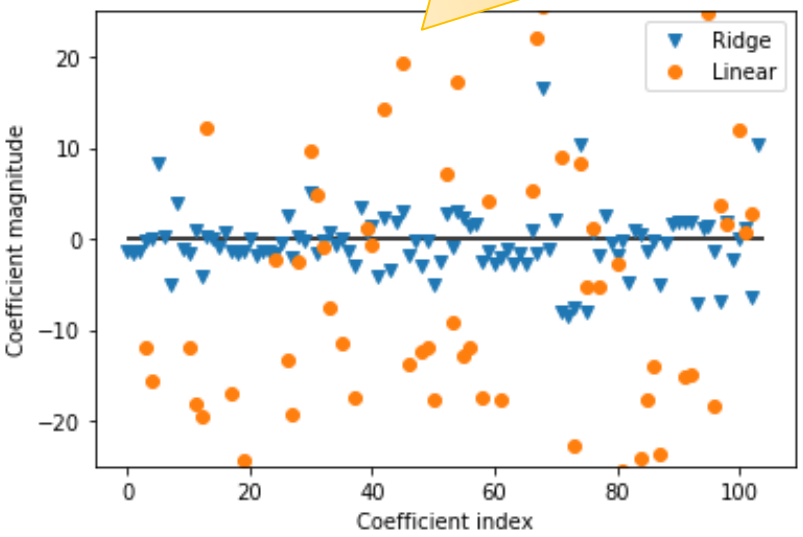
$$E(\boldsymbol{w}) = \left|\boldsymbol{y} - \tilde{\boldsymbol{X}}\boldsymbol{w}\right|^2 + \lambda|w|_1$$

- ここで$|w|_1 = \sum_{i=1}^{d}|w_i|$
- 係数があまりに大きくならないようにしている

線形回帰とRidge回帰の比較

- Boston house prices dataset
- 目的変数：ボストン近郊の住宅地の住宅価格の中央値
- 説明変数：犯罪率、固定資産税率、生徒教師比率等13の特徴量と、13から2つの特徴量を重複を許して選んだ組み合わせ91の合計104の特徴量
- サンプル数506
  - 379を訓練データにする
  - 127をテストデータにする
- 訓練データを用いて104の説明変数の重みを求めて、テストデータで推定する

▼Ridgeは係数小さい
●Linearはどれも係数大きい



(Linear) Training set score: 0.95
(Linear) Test set score: 0.61
(Ridge) Training set score: 0.89
(Ridge) Test set score: 0.75

Ridgeの方がテストのscoreが高くなる

Ridge回帰は係数の大きさにペナルティを与えることで過剰適合を防ぐ

※データ出典：A. C. Muller et al. mglearn.datasets

SVM

教師あり学習（分類／回帰）

サポートベクターマシン（SVM）

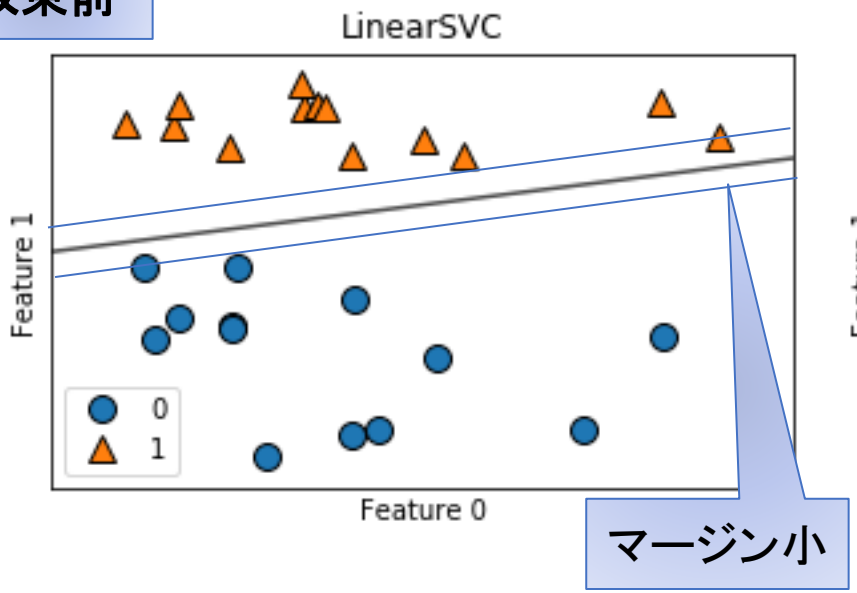
教師あり学習のひとつ
- 分類もしくは回帰に利用可能

SVMの種類
- ハードマージンSVM
  - 各クラスのデータが分類境界からなるべく離れるように分類境界を定める
- ソフトマージンSVM
  - 線形分離できない場合に、スラック変数ξを導入して、分類境界で分離できないデータを許容する
- カーネルSVM
  - カーネル法によって入力データをより高次元の特徴空間に写像し、特徴空間上で線形分離することで、実空間での非線形分離を可能にする
- SV回帰
  - SVMを回帰問題へ応用したもの

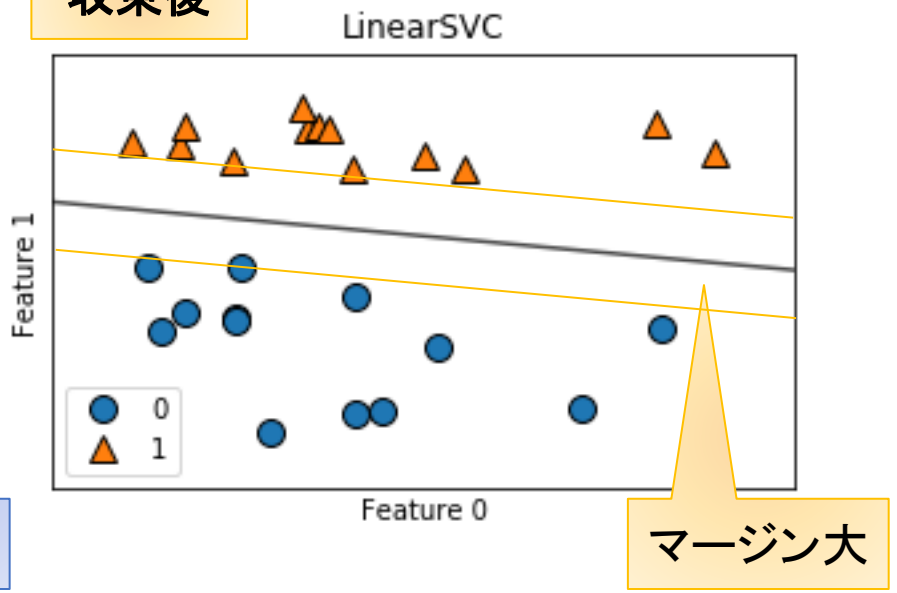
SVMの実装
- Rのkernlabパッケージなどで実装されている

ハードマージンSVM

収束前



収束後

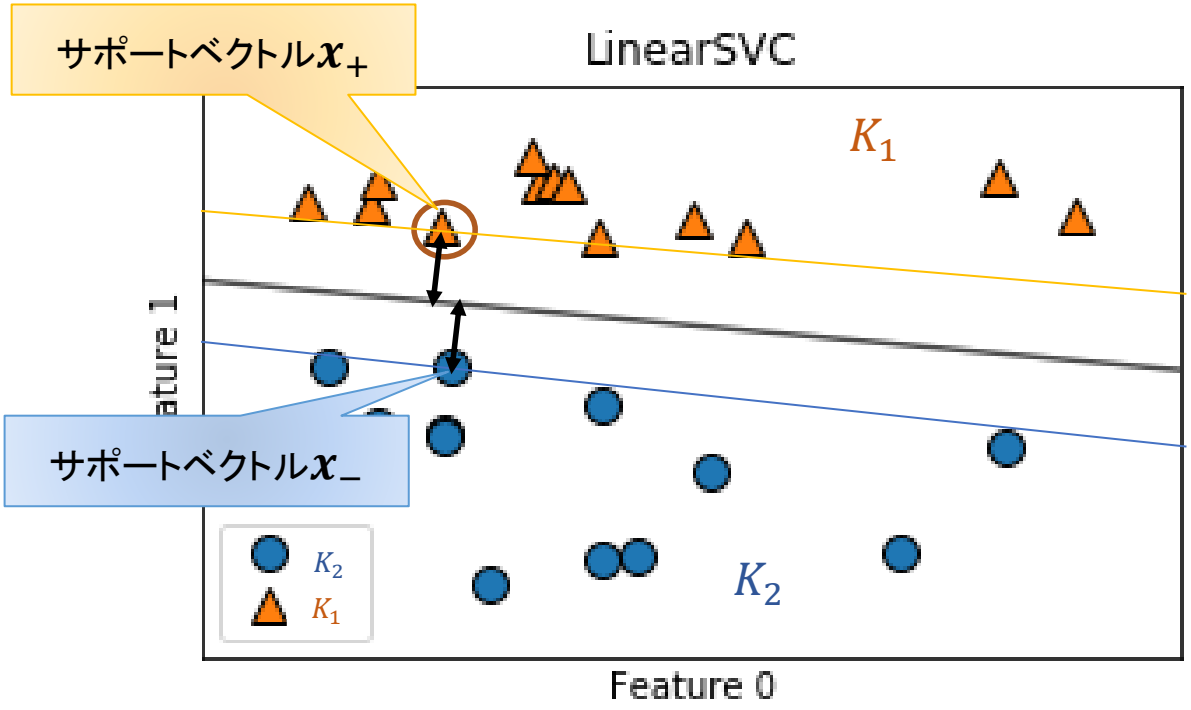


それぞれのクラスのデータが分類境界からなるべく離れる（マージンを最大化する）ように分類境界を定める

サポートベクトル：分類境界に最も近いサンプル
マージン：サポートベクトルと分類境界の距離

※データ出典：A. C. Muller et al. mglearn.datasetsを改変

マージン最大化による分類境界の導出(1/3)



分類境界

$$w^T x + b = 0$$

$$w^T x_i + b > 0$$

$$t_i = \begin{cases} 1\ (x_i \in K_1) \\ -1\ (x_i \in K_2) \end{cases}$$

$$w^T x_i + b < 0$$

なるラベル変数t_iを導入すると，

任意の点x_iと分類境界平面との距離は $\dfrac{t_i(w^T x_i + b)}{|w|}$ と表される．これより，マージン最大化は

$$\underset{w,b}{\text{Maximize}} \min_i \frac{t_i(w^T x_i + b)}{|w|}$$

※データ出典：A. C. Muller et al. mglearn.datasetsを改変

マージン最大化による分類境界の導出(2/3)

$$\underset{\boldsymbol{w},b}{\text{Maximize}} \min_i \frac{t_i(\boldsymbol{w}^T\boldsymbol{x}_i + b)}{|\boldsymbol{w}|} = \underset{\boldsymbol{w},b}{\text{Maximize}} \frac{1}{|\boldsymbol{w}|} \min_i t_i(\boldsymbol{w}^T\boldsymbol{x}_i + b)$$

今、$\min_i t_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) = 1$ の場合だけ考えればよい．これより最適化の式は，

$$\underset{\boldsymbol{w},b}{\text{Maximize}} \frac{1}{|\boldsymbol{w}|} \quad \text{Subject to } t_i(\boldsymbol{w}^T\boldsymbol{x_i} + b) \geq 1 \ (i = 1,2,\cdots,n)$$

さらに簡単化して，

$$\underset{\boldsymbol{w},b}{\text{Minimize}} \frac{1}{2}|\boldsymbol{w}|^2 \quad \text{Subject to } t_i(\boldsymbol{w}^T\boldsymbol{x_i} + b) \geq 1 \ (i = 1,2,\cdots,n)$$

ラグランジュ未定乗数$\boldsymbol{a}$を導入して以下のラグランジュ関数を定義する

$$L(b,\boldsymbol{w},\boldsymbol{a}) = \frac{1}{2}|\boldsymbol{w}|^2 - \sum_{i=1}^{n} a_i\{t_i(\boldsymbol{b} + \boldsymbol{w}^T\boldsymbol{x_i}) - 1\}$$

Lをb,wで偏微分して0とおいたものを用いてLを変形すると以下のような$\boldsymbol{a}$だけの関数になる

$$L(\boldsymbol{a}) = \sum_{i=1}^{n} a_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j t_i t_j \boldsymbol{x}_i^T\boldsymbol{x}_j$$

マージン最大化による分類境界の導出（3/3）

主問題であったマージン最大化問題は双対問題である以下の二次計画問題に帰着される

$$\text{Maximize } L(\boldsymbol{a}) = \sum_{i=1}^{n} a_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j t_i t_j \boldsymbol{x}_i^T \boldsymbol{x}_j$$

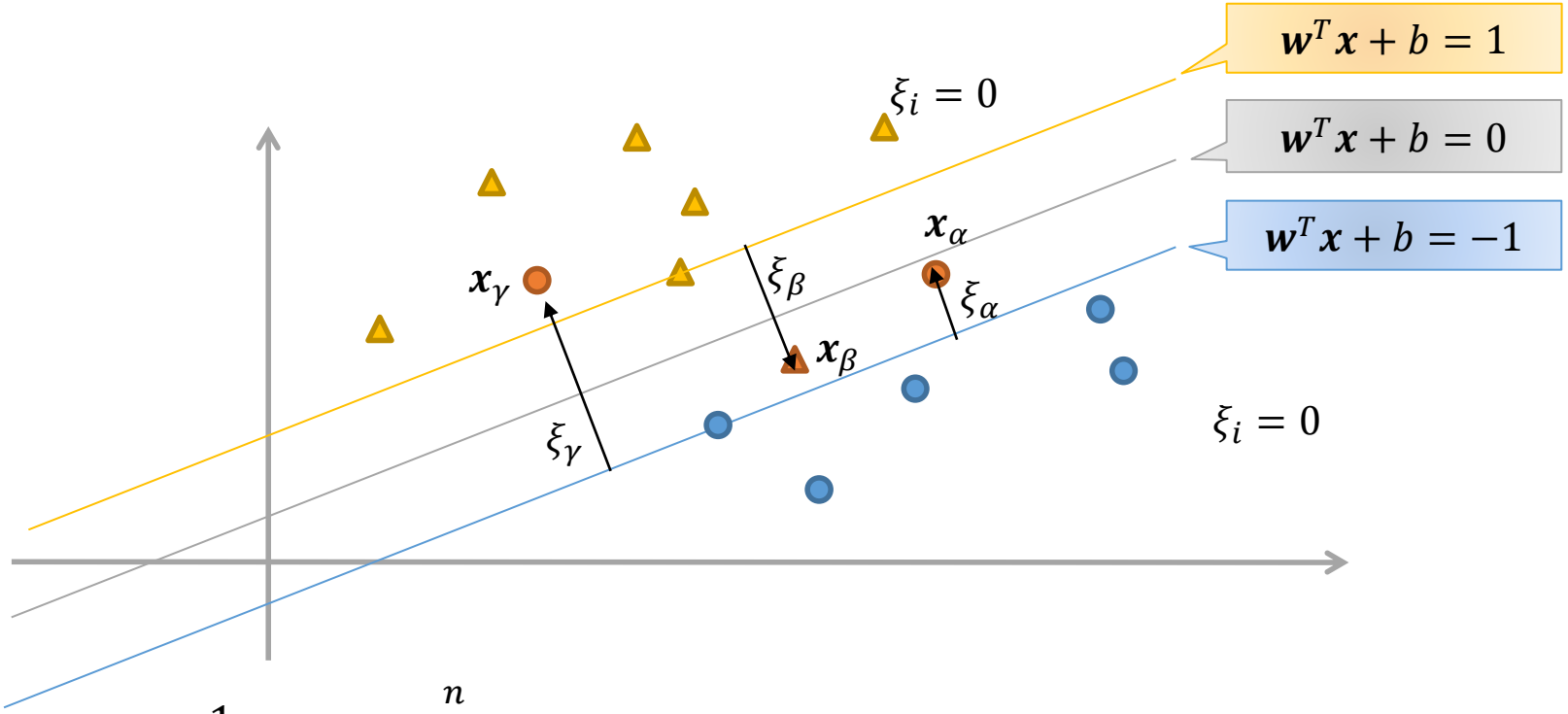
$$\text{Subject to } \sum_{i=1}^{n} a_i t_i = 0$$

$$a_i \geq 0 \qquad a_i\{t_i(b + \boldsymbol{w}^T \boldsymbol{x}_i) - 1\} = 0$$

SVMのライブラリ内部では、このような双対問題を解くような実装がなされている

ソフトマージンSVM

- 線形分離できない場合に、スラック変数ξを導入して、分離境界で分離できないデータを許容する方法



$$\underset{w,b,\xi}{\text{Minimize}} \quad \frac{1}{2}|\boldsymbol{w}|^2 + C\sum_{i=1}^{n}\xi_i$$

$$\text{Subject to } t_i(\boldsymbol{w}^T\boldsymbol{x_i} + b) \geq 1 - \xi_i \ (i = 1,2,\cdots,n)$$

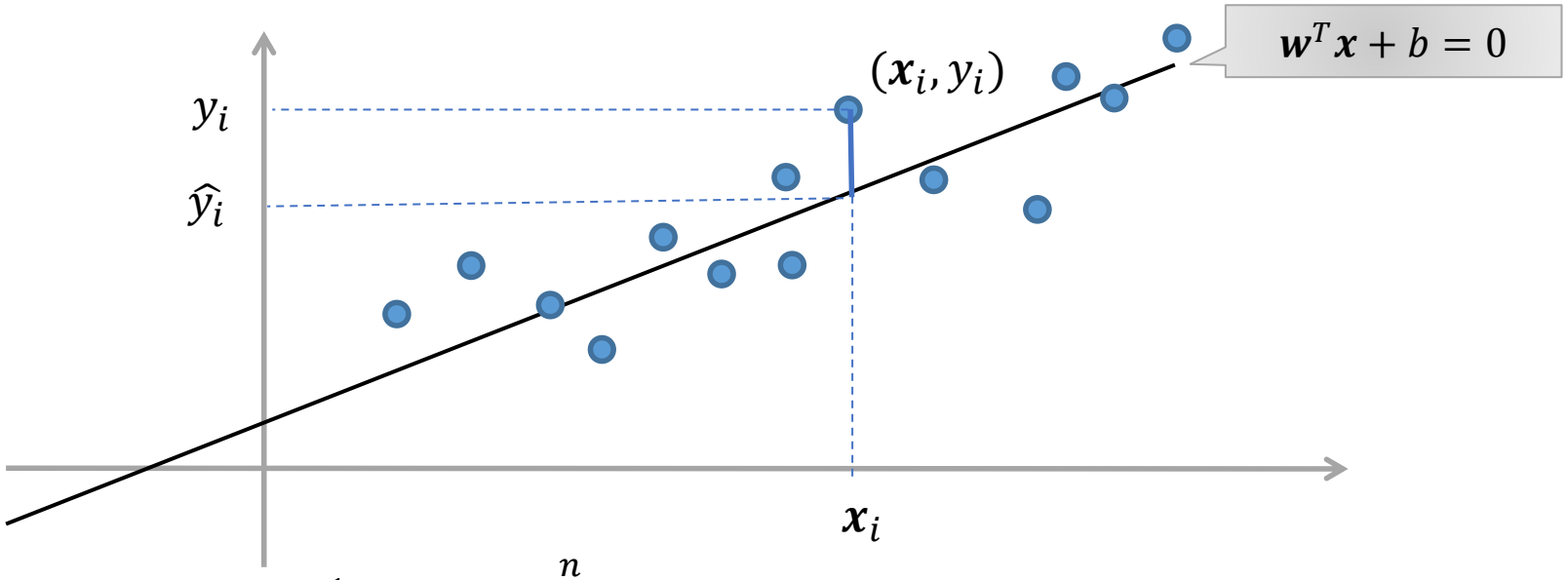
$$\xi_i \geq 0 \ (i = 1,2,\cdots,n)$$

C:小さい⇒大きいξを許容

C:→ ∞ ⇒ $\xi = 0$のみ許容（ハードマージン）

SV回帰

- SVMの回帰問題への応用



$$\underset{\boldsymbol{w},b}{\text{Minimize}} \quad \frac{1}{2}|\boldsymbol{w}|^2 + C\sum_{i=1}^{n} \max\{|y_i - (\boldsymbol{w}^T\boldsymbol{x}_i + b)| - \epsilon, 0\}$$

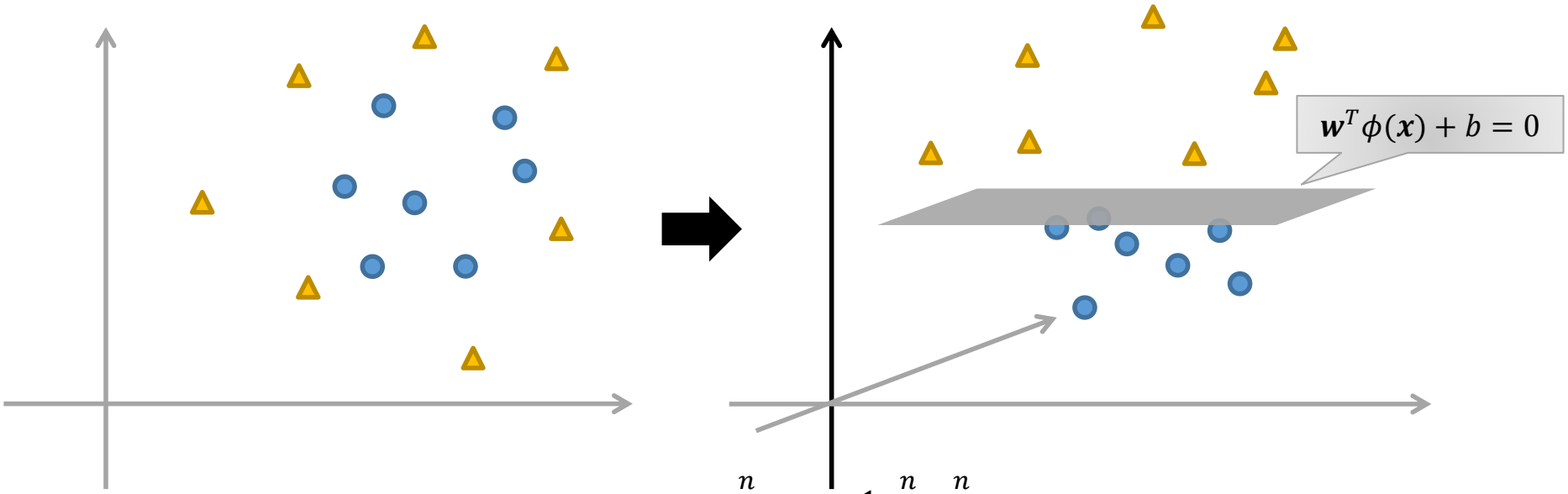
$\epsilon > 0$： 不感度係数
$C > 0$ ：正則化係数

不感度：残差がϵ以下の時は損失0
正則化：小さい ⇒ 残差がもたらす損失に対して寛容

※カーネル法を用いれば非線形回帰も可能

非線形SVM

入力データをより高次元の特徴空間に写像し、特徴空間上で線形分離することで、実空間での非線形分離を可能にする方法



$$\text{Maximize } L(\boldsymbol{a}) = \sum_{i=1}^{n} a_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j t_i t_j \phi(\boldsymbol{x}_i)^T \phi(\boldsymbol{x}_j)$$

$$\text{Subject to } \sum_{i=1}^{n} a_i t_i = 0 \qquad a_i \geq 0$$

カーネル法

$\phi(\boldsymbol{x}_i)^T \phi(\boldsymbol{x}_j) = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ **とおくと、目的関数は次のようになる**

$$L(\boldsymbol{a}) = \sum_{i=1}^{n} a_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j t_i t_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

ここでKはカーネル関数と呼ばれる

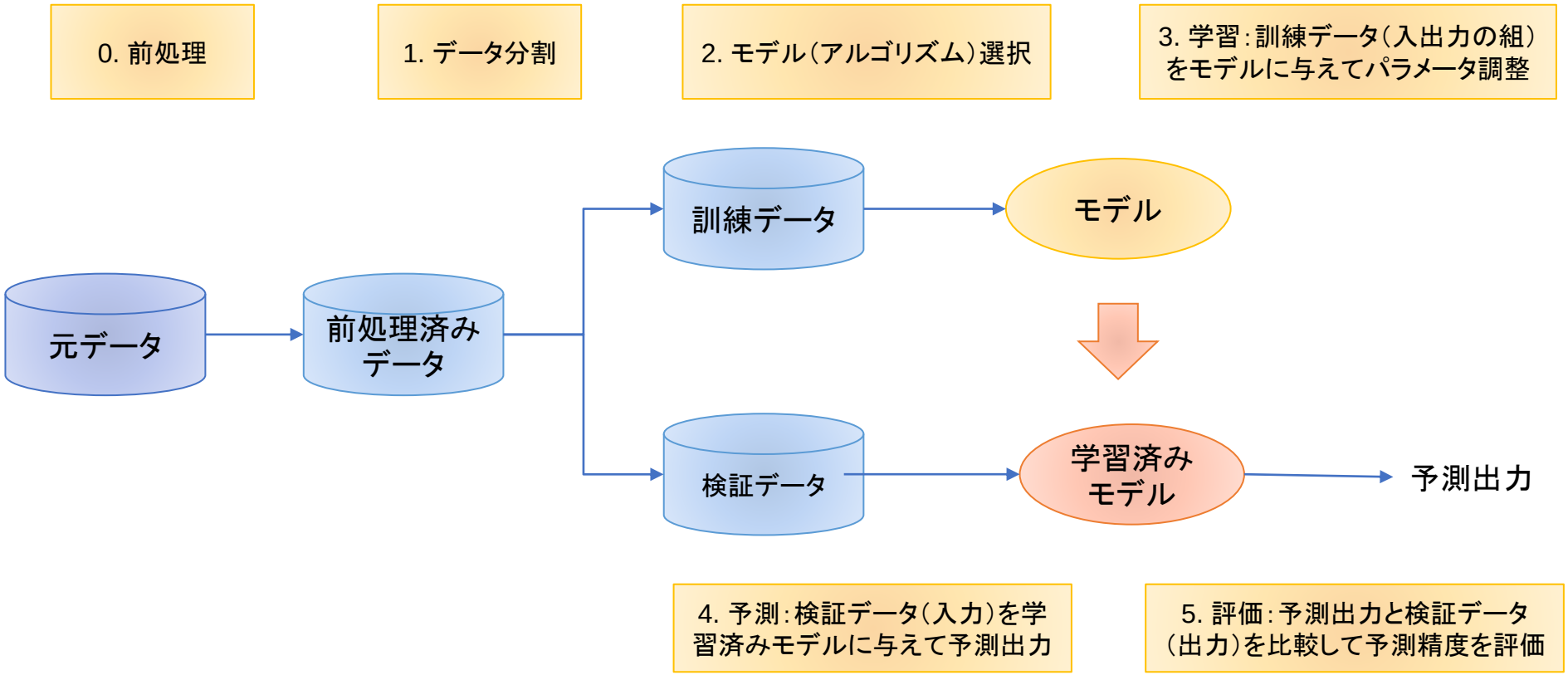
カーネル関数を使うことで、特徴空間における内積$\phi(\boldsymbol{x}_i)^T \phi(\boldsymbol{x}_j)$を明示的に計算せずに、$K$だけの評価で問題を解くことができる
特徴空間中の座標を計算しないことで計算量を抑えることができる
カーネル関数の一例

- RBFカーネル（Radial Basis Function kernel）
- 実データ　$\boldsymbol{x_i}$ と　$\boldsymbol{x}_j$の特徴空間における内積

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{|\boldsymbol{x}_i - \boldsymbol{x}_j|^2}{2\sigma^2}\right)$$

演習：教師あり学習（回帰）演習ガイド参照

機械学習モデル構築（演習）手順



演習：Wine Qualityデータセットを利用

- 11種類の科学的特性に基づいて品質スコアを推定する機械学習モデルを構築
- 線形重回帰かSVMを採用
- 必要なら正規化・標準化等の前処理を実施
- 白ワイン4,898事例のデータを訓練データ3,750事例、検証データ1,148事例に分割
- 訓練データでモデルを訓練
- 検証データを学習済みモデルに与え、その予測精度を評価

第4章

教師なし学習（次元削減）

教師なし学習

出力に関するデータ（正解）が与えられていない
クラスタリングか次元圧縮を目的とする

- クラスタリング
  - 入力データに基づいてサンプルをいくつかのクラスタに分類する
- 次元圧縮
  - 多種類の入力を少数種類の入力にまとめる

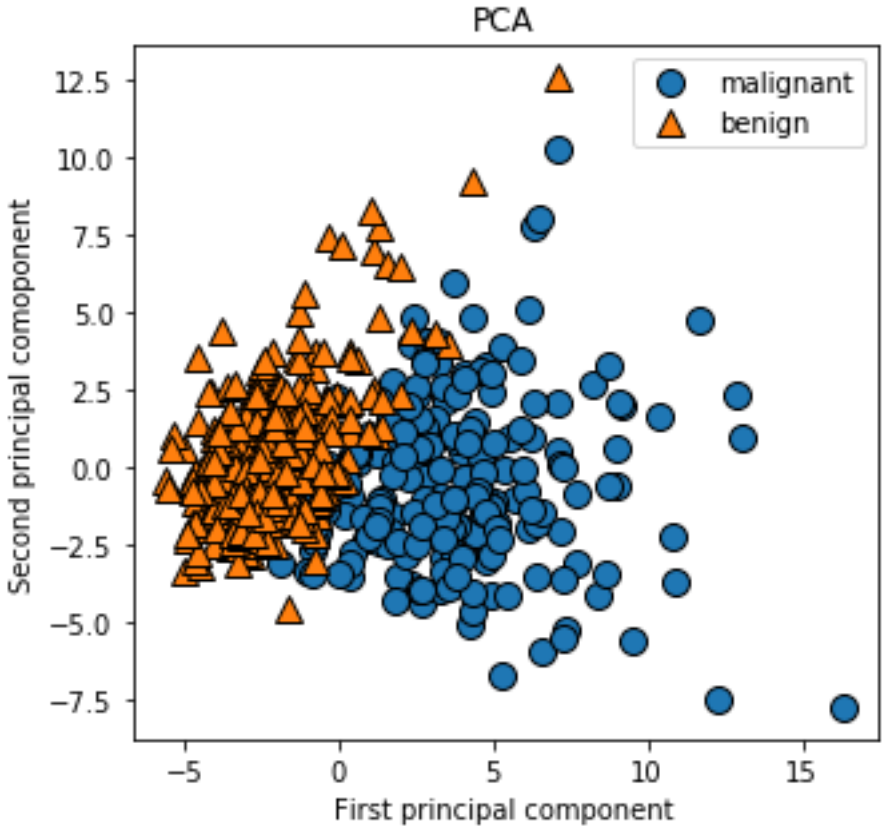
ex. 顧客の属性に基づいて顧客をいくつかのクラスタに分類する（クラスタリング）、学生の複数科目の試験成績データに基づいて学生の能力を説明できる少数の変数を作る（次元圧縮）

主成分分析（PCA）

教師なし学習（次元圧縮）

主成分分析の概要と事例

主成分分析：データの特徴を表現できる少数の合成変数を見出す手法



例：Breast cancer wisconsin (diagnostic) dataset

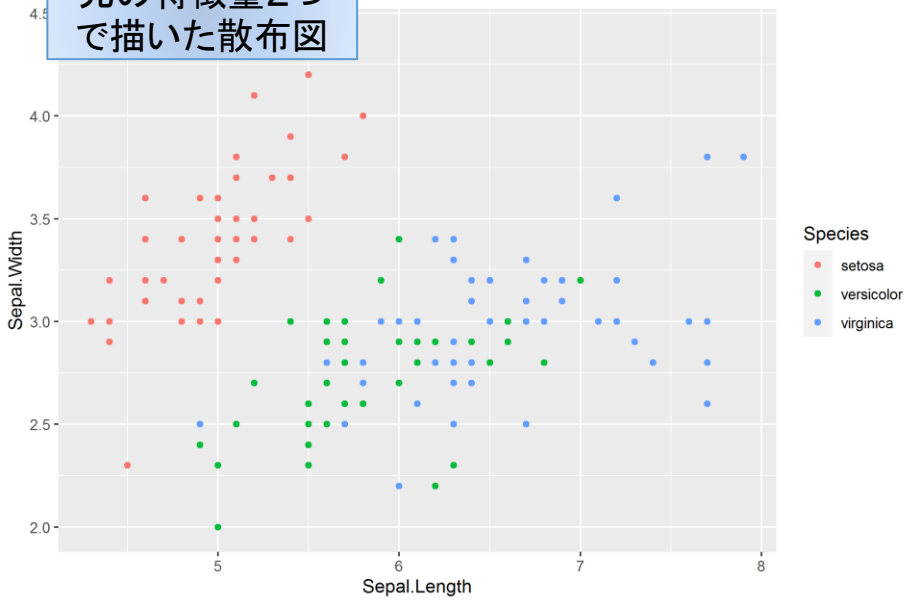
- 腫瘍細胞569例のそれぞれに関する直径や滑らかさ、対称性等30の特徴量の測定値からなるデータセット
- 良性（benign）357例、悪性（malignant）212例
- 30次元特徴空間は描画できない
- 主成分分析を行い30の特徴量を2つの主成分に縮約して散布図を描画

多次元特徴量を次元縮約して可視化が可能

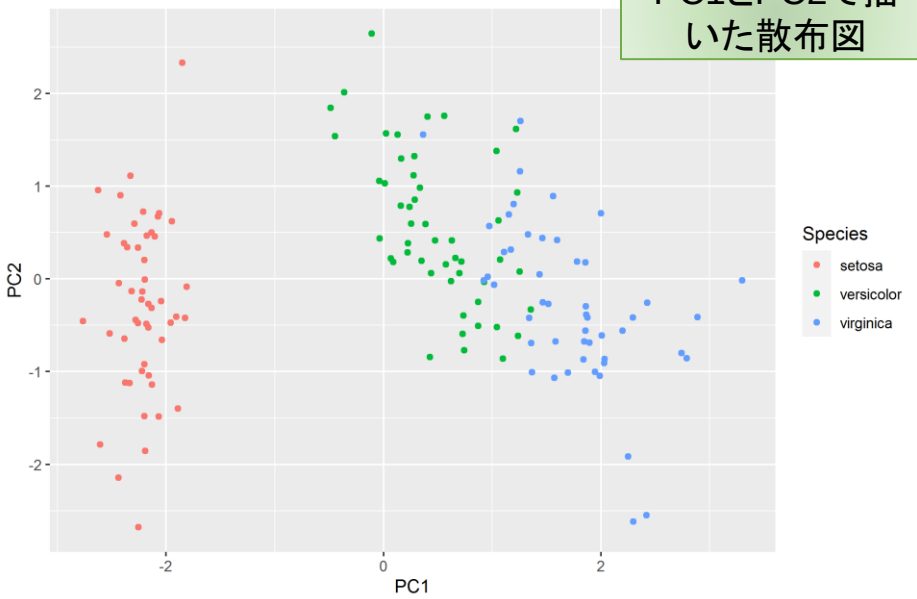
※データ出典：sklearn.datasets.load_breast_cancer()

主成分分析事例

元の特徴量2つで描いた散布図



PC1とPC2で描いた散布図

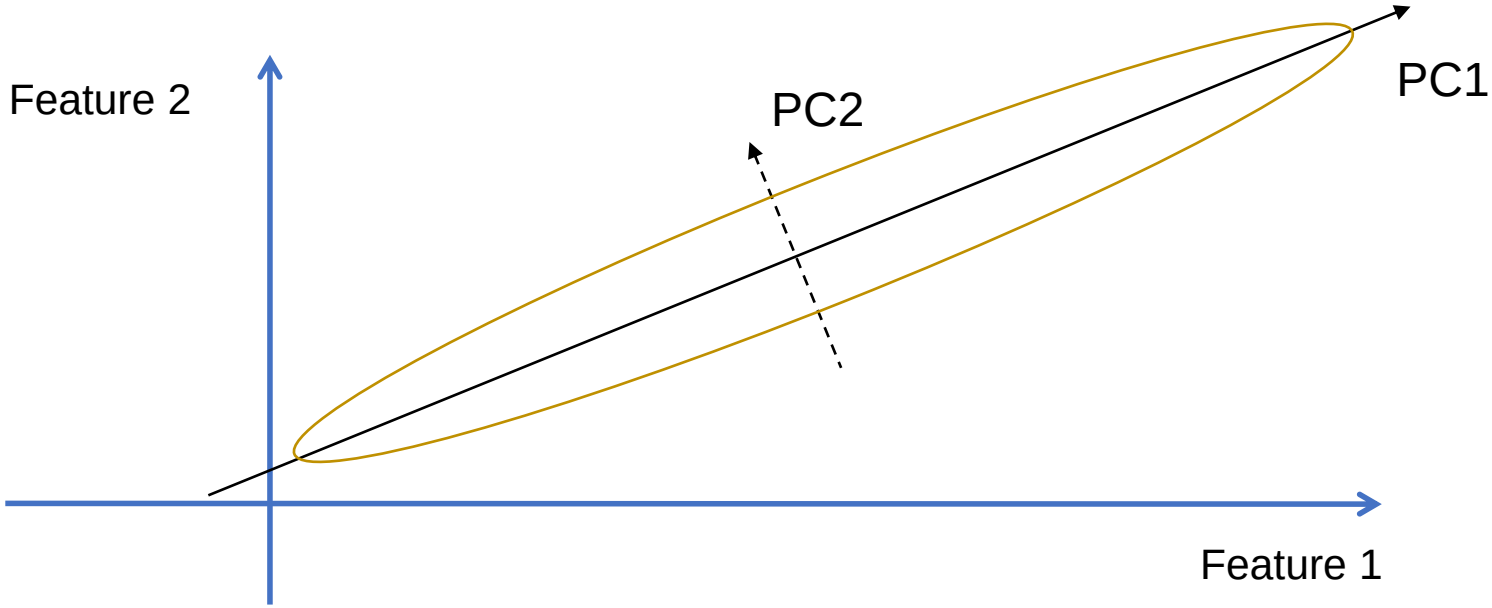


例：Iris Species Dataset

- 3種類のアヤメsetosa 50、versicolor 50、virginica 50の各個体に関するSepal.Width, Sepal.Length, Petal.Width, Petal.Lengthの4つの特徴量の測定値からなるデータセット
- 任意の2変数で散布図を描画するが（左）、どの変数も3種類のアヤメを分離できるような変数ではない
- 主成分分析を行い、第1主成分（PC1）と第2主成分（PC2）の2変数で散布図を描画（右）すると、PC1はそれだけでも3種類のアヤメを分離できるような変数になっている

識別しやすい特徴の抽出

主成分の見つけ方 -分散最大基準-



1. データの分散が最大となる方向を見つけ第1主成分とする
2. それに直交する方向で分散が最大となる方向を第2主成分とする
3. 次元数に至るまで繰り返す

- 特徴量が元々2つの場合、上図の黄色楕円のようにデータが分布している場合、長軸方向が第1主成分、それに直交する方向が第2主成分となり完了
- 特徴量がd個の場合、d次元特徴空間において同様のことを行う

主成分分析のアルゴリズムと実装

- データ行列に基づいて共分散行列Sを求める
- Sの固有値問題を解く
  - $S\boldsymbol{v} = \lambda\boldsymbol{v}$
  - λ: 固有値、$\boldsymbol{v}$: 固有ベクトル
  - 固有値λは固有ベクトル$\boldsymbol{v}$方向に射影したデータの分散となっている
- 対応する固有値の大きい固有ベクトルから順に第1主成分、第2主成分…となる

分散最大化問題は共分散行列の固有値問題に帰着するため、固有値問題を解けばよい

- 実装
  - Rのprcompパッケージなどで利用可能

共分散行列の固有値問題(1/2)

分散最大化問題は共分散行列の固有値問題に帰着する

- 簡単のため1次元への射影を考える。射影先の単位ベクトルを$\boldsymbol{w}_1$とすると$\boldsymbol{w}_1^T\boldsymbol{w}_1 = 1$である。データ$\{\boldsymbol{x}_i\}_{i=1}^n$の平均を$\bar{\boldsymbol{x}} = \frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i$とする。
- データ点$\boldsymbol{x}_i$の$\boldsymbol{w}_1$方向への射影は内積$\boldsymbol{w}_1 \cdot \boldsymbol{x}_i = \boldsymbol{w}_1^T\boldsymbol{x}_i = \boldsymbol{x}_i^T\boldsymbol{w}_1$で表されることをふまえ、$\{\boldsymbol{w}_1^T\boldsymbol{x}_i\}_{i=1}^n$の平均を計算すると

$$\frac{1}{n}\sum_{i=1}^n \boldsymbol{w}_1^T\boldsymbol{x}_i = \frac{1}{n}\boldsymbol{w}_1^T\sum_{i=1}^n \boldsymbol{x}_i = \boldsymbol{w}_1^T\bar{\boldsymbol{x}}$$

- したがって$\{\boldsymbol{w}_1^T\boldsymbol{x}_i\}_{i=1}^n$の分散は

$$\frac{1}{n}\sum_{i=1}^n \left|\boldsymbol{w}_1^T\boldsymbol{x}_i - \boldsymbol{w}_1^T\bar{\boldsymbol{x}}\right|^2 = \frac{1}{n}\sum_{i=1}^n \{\boldsymbol{w}_1^T(\boldsymbol{x}_i - \bar{\boldsymbol{x}})\}\{(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^T\boldsymbol{w}_1\}$$

$$= \boldsymbol{w}_1^T\left\{\frac{1}{n}\sum_{i=1}^n (\boldsymbol{x}_i - \bar{\boldsymbol{x}})(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^T\right\}\boldsymbol{w}_1$$

共分散行列の固有値問題(2/2)

ここで、$\frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{x}_i-\bar{\boldsymbol{x}})(\boldsymbol{x}_i-\bar{\boldsymbol{x}})^T$ は$\boldsymbol{x}_i$の共分散行列であり、Sとおく。Sは(d,d)行列で、解くべき問題は次の最適化問題になる

$$\boldsymbol{w}_1^T\boldsymbol{w}=1\text{の条件の下 }\boldsymbol{w}_1^T\mathrm{S}\boldsymbol{w}_1\text{を最大化する}$$

ラグランジュ未定乗数法で解くため以下の関数ϕを考える

$$\phi(\boldsymbol{w}_1,\lambda_1)=\boldsymbol{w}_1^T\mathrm{S}\boldsymbol{w}_1-\lambda_1(\boldsymbol{w}_1^T\boldsymbol{w}_1-1)$$

$\frac{\partial\phi}{\partial\boldsymbol{w}_1}=2\mathrm{S}\boldsymbol{w}_1-2\lambda_1\boldsymbol{w}_1=0$ より

$$\mathrm{S}\boldsymbol{w}_1=\lambda_1\boldsymbol{w}_1$$

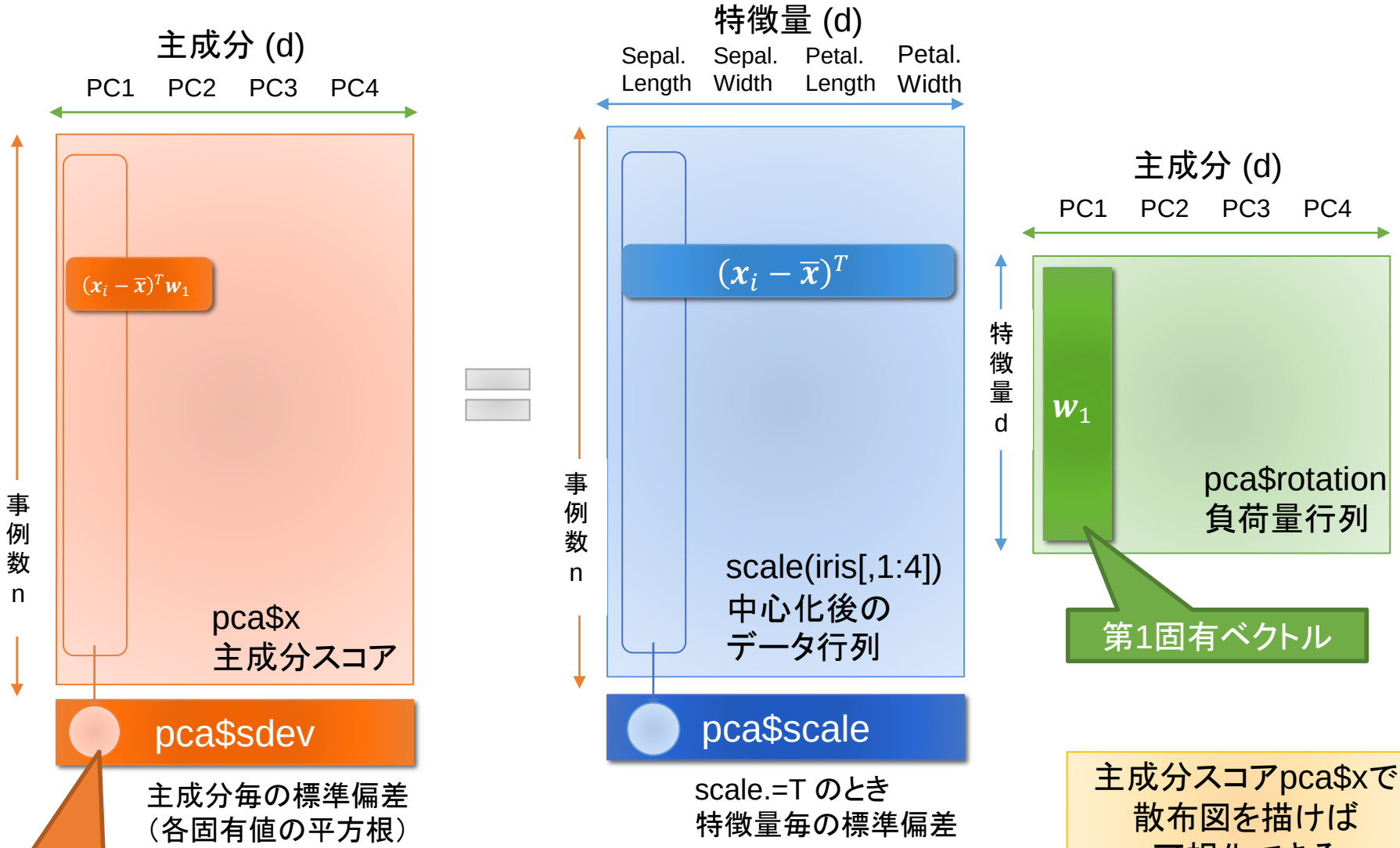
λ_1はSの固有値である。また、このとき

$$\boldsymbol{w}_1^T\mathrm{S}\boldsymbol{w}_1=\boldsymbol{w}_1^T\lambda_1\boldsymbol{w}_1=\lambda_1$$

であるため、分散は固有値に一致し、分散最大化問題は最大固有値を求める問題に帰着した

Rの主成分分析で得られるデータの構造

pca <- prcomp(iris[,1:4])
を例として



主成分スコアpca$xで
散布図を描けば
可視化できる

累積寄与率等

summary(pca)で以下の情報が得られる

- Standard deviation
  - 各主成分の標準偏差
- Proportion of Variance（寄与率）
  - 各主成分の固有値（分散）をその総和でわったもの
- Cumulative Proportion（累積寄与率）
  - 第1主成分から当該主成分までの寄与率の和

累積寄与率から、どの主成分まで採用すれば
全体の分散を十分に説明できるか判断する

教師なし学習（次元削減）

教師なし学習

出力に関するデータ（正解）が与えられていない
クラスタリングか次元圧縮を目的とする

- クラスタリング
  - 入力データに基づいてサンプルをいくつかのクラスタに分類する
- 次元圧縮
  - 多種類の入力を少数種類の入力にまとめる

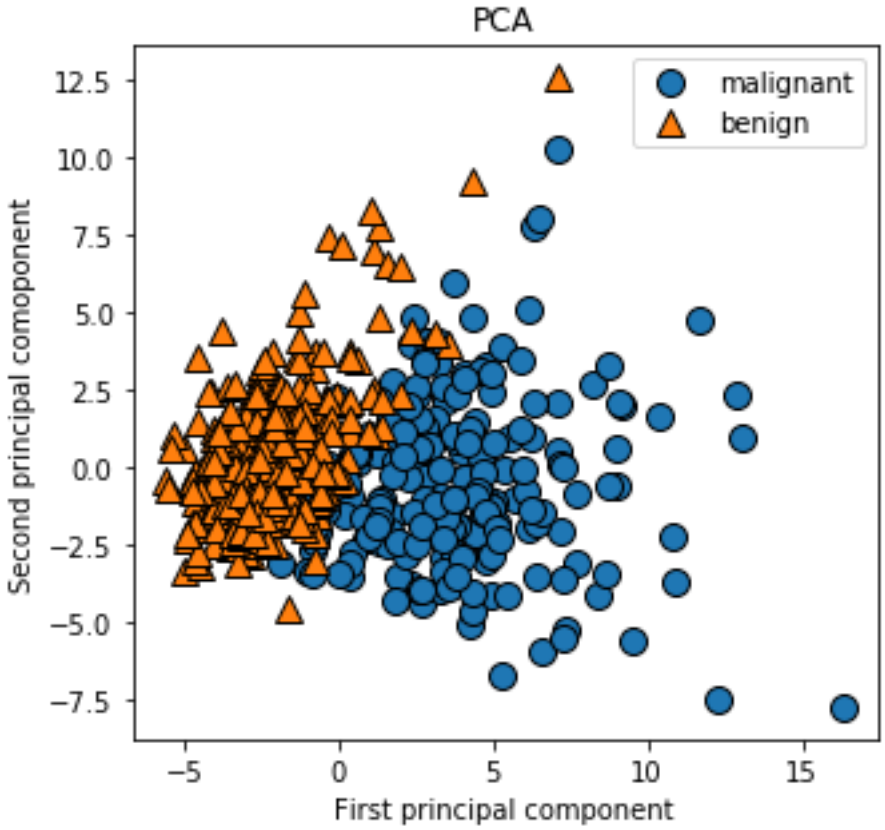
ex. 顧客の属性に基づいて顧客をいくつかのクラスタに分類する（クラスタリング）、学生の複数科目の試験成績データに基づいて学生の能力を説明できる少数の変数を作る（次元圧縮）

主成分分析（PCA）

教師なし学習（次元圧縮）

主成分分析の概要と事例

主成分分析：データの特徴を表現できる少数の合成変数を見出す手法



例：Breast cancer wisconsin (diagnostic) dataset

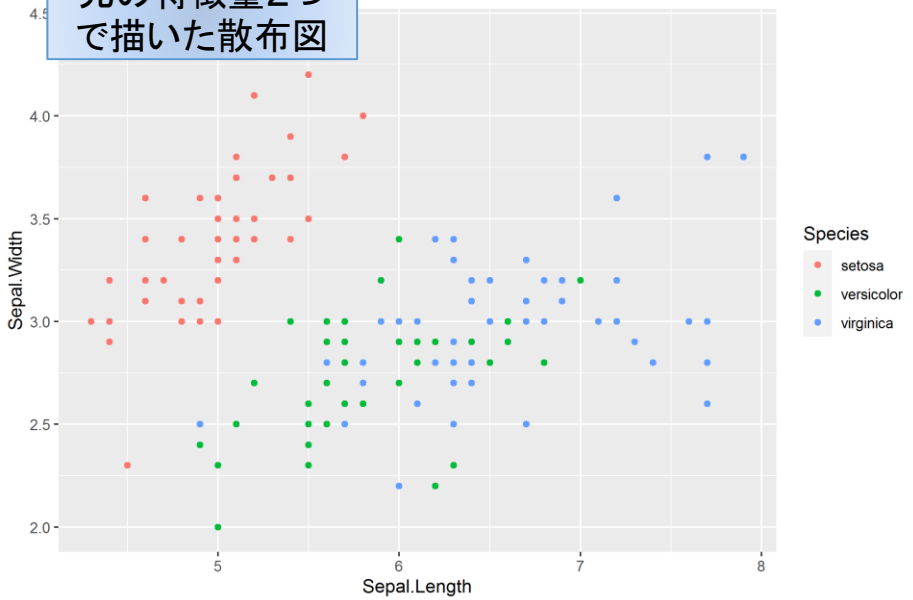
- 腫瘍細胞569例のそれぞれに関する直径や滑らかさ、対称性等30の特徴量の測定値からなるデータセット
- 良性（benign）357例、悪性（malignant）212例
- 30次元特徴空間は描画できない
- 主成分分析を行い30の特徴量を2つの主成分に縮約して散布図を描画

多次元特徴量を次元縮約して可視化が可能

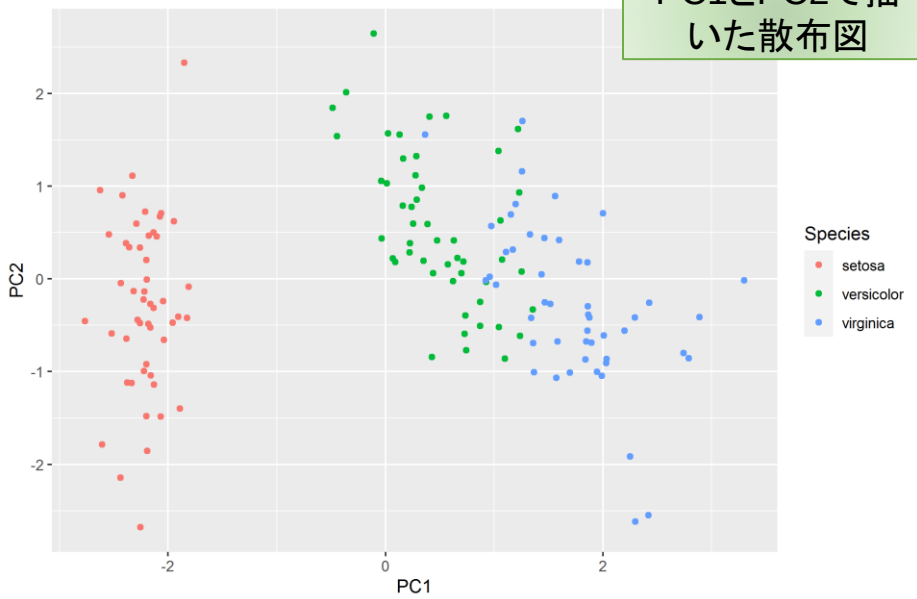
※データ出典：sklearn.datasets.load_breast_cancer()

主成分分析事例

元の特徴量2つで描いた散布図



PC1とPC2で描いた散布図

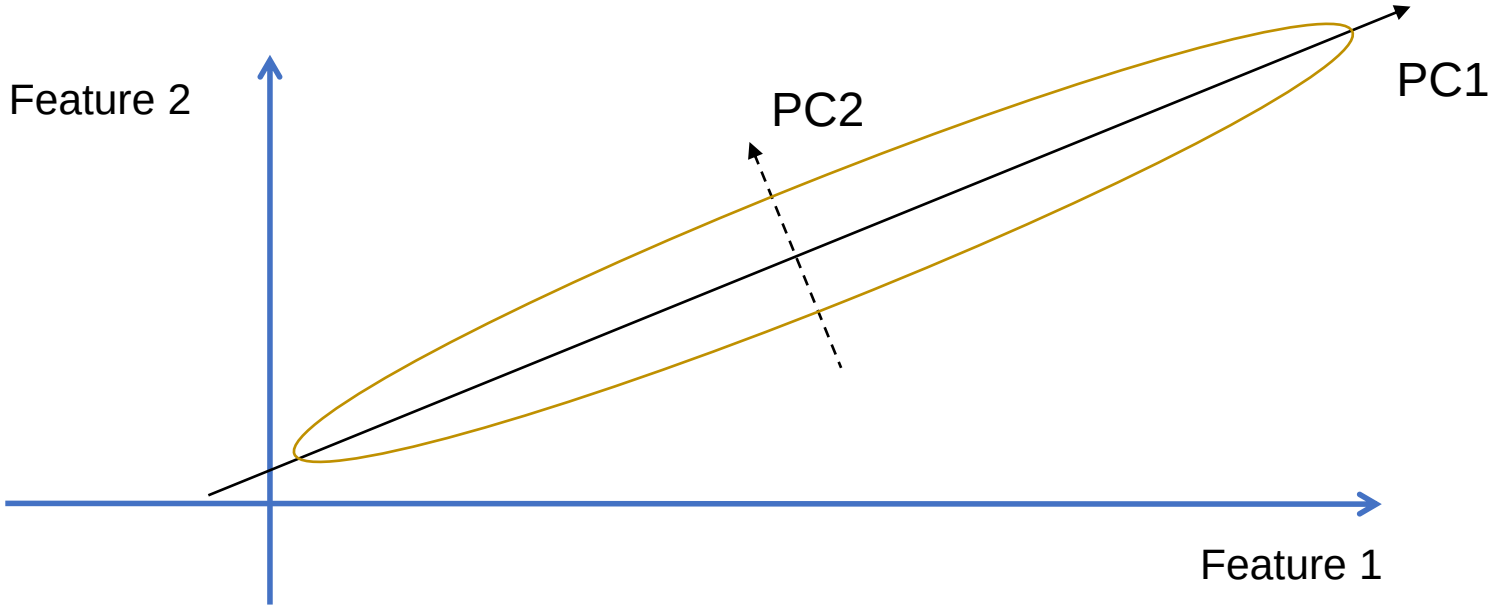


例：Iris Species Dataset

- 3種類のアヤメsetosa 50、versicolor 50、virginica 50の各個体に関するSepal.Width, Sepal.Length, Petal.Width, Petal.Lengthの4つの特徴量の測定値からなるデータセット
- 任意の2変数で散布図を描画するが（左）、どの変数も3種類のアヤメを分離できるような変数ではない
- 主成分分析を行い、第1主成分（PC1）と第2主成分（PC2）の2変数で散布図を描画（右）すると、PC1はそれだけでも3種類のアヤメを分離できるような変数になっている

識別しやすい特徴の抽出

主成分の見つけ方 -分散最大基準-



1. データの分散が最大となる方向を見つけ第1主成分とする
2. それに直交する方向で分散が最大となる方向を第2主成分とする
3. 次元数に至るまで繰り返す

- 特徴量が元々2つの場合、上図の黄色楕円のようにデータが分布している場合、長軸方向が第1主成分、それに直交する方向が第2主成分となり完了
- 特徴量がn個の場合、n次元特徴空間において同様のことを行う

主成分分析のアルゴリズムと実装

- データ行列に基づいて共分散行列Sを求める
- Sの固有値問題を解く
  - $S\boldsymbol{v} = \lambda\boldsymbol{v}$
  - λ: 固有値、$\boldsymbol{v}$: 固有ベクトル
  - 固有値λは固有ベクトル$\boldsymbol{v}$方向に射影したデータの分散となっている
- 対応する固有値の大きい固有ベクトルから順に第1主成分、第2主成分…となる

分散最大化問題は共分散行列の固有値問題に帰着するため、固有値問題を解けばよい

- 実装
  - Rのprcompパッケージなどで利用可能

共分散行列の固有値問題(1/2)

分散最大化問題は共分散行列の固有値問題に帰着する

- 簡単のため1次元への射影を考える。射影先の単位ベクトルを$\boldsymbol{w}_1$とすると$\boldsymbol{w}_1^T\boldsymbol{w}_1 = 1$である。データ$\{\boldsymbol{x}_i\}_{i=1}^n$の平均を$\bar{\boldsymbol{x}} = \frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i$とする。
- データ点$\boldsymbol{x}_i$の$\boldsymbol{w}_1$方向への射影は内積$\boldsymbol{w}_1 \cdot \boldsymbol{x}_i = \boldsymbol{w}_1^T\boldsymbol{x}_i = \boldsymbol{x}_i^T\boldsymbol{w}_1$で表されることをふまえ、$\{\boldsymbol{w}_1^T\boldsymbol{x}_i\}_{i=1}^n$の平均を計算すると

$$\frac{1}{n}\sum_{i=1}^n \boldsymbol{w}_1^T\boldsymbol{x}_i = \frac{1}{n}\boldsymbol{w}_1^T\sum_{i=1}^n \boldsymbol{x}_i = \boldsymbol{w}_1^T\bar{\boldsymbol{x}}$$

- したがって$\{\boldsymbol{w}_1^T\boldsymbol{x}_i\}_{i=1}^n$の分散は

$$\frac{1}{n}\sum_{i=1}^n \left|\boldsymbol{w}_1^T\boldsymbol{x}_i - \boldsymbol{w}_1^T\bar{\boldsymbol{x}}\right|^2 = \frac{1}{n}\sum_{i=1}^n \{\boldsymbol{w}_1^T(\boldsymbol{x}_i - \bar{\boldsymbol{x}})\}\{(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^T\boldsymbol{w}_1\}$$

$$= \boldsymbol{w}_1^T\left\{\frac{1}{n}\sum_{i=1}^n (\boldsymbol{x}_i - \bar{\boldsymbol{x}})(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^T\right\}\boldsymbol{w}_1$$

共分散行列の固有値問題(2/2)

ここで、$\frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{x}_i-\bar{\boldsymbol{x}})(\boldsymbol{x}_i-\bar{\boldsymbol{x}})^T$ は$\boldsymbol{x}_i$の共分散行列であり、Sとおく。Sは(d,d)行列で、解くべき問題は次の最適化問題になる

$$\boldsymbol{w}_1^T\boldsymbol{w}=1\text{の条件の下 }\boldsymbol{w}_1^T\mathrm{S}\boldsymbol{w}_1\text{を最大化する}$$

ラグランジュ未定乗数法で解くため以下の関数ϕを考える

$$\phi(\boldsymbol{w}_1,\lambda_1)=\frac{1}{2}\boldsymbol{w}_1^T\mathrm{S}\boldsymbol{w}_1-\lambda_1(\boldsymbol{w}_1^T\boldsymbol{w}_1-1)$$

$\frac{\partial\phi}{\partial\boldsymbol{w}_1}=\mathrm{S}\boldsymbol{w}_1-\lambda_1\boldsymbol{w}_1=0$ より

$$\mathrm{S}\boldsymbol{w}_1=\lambda_1\boldsymbol{w}_1$$

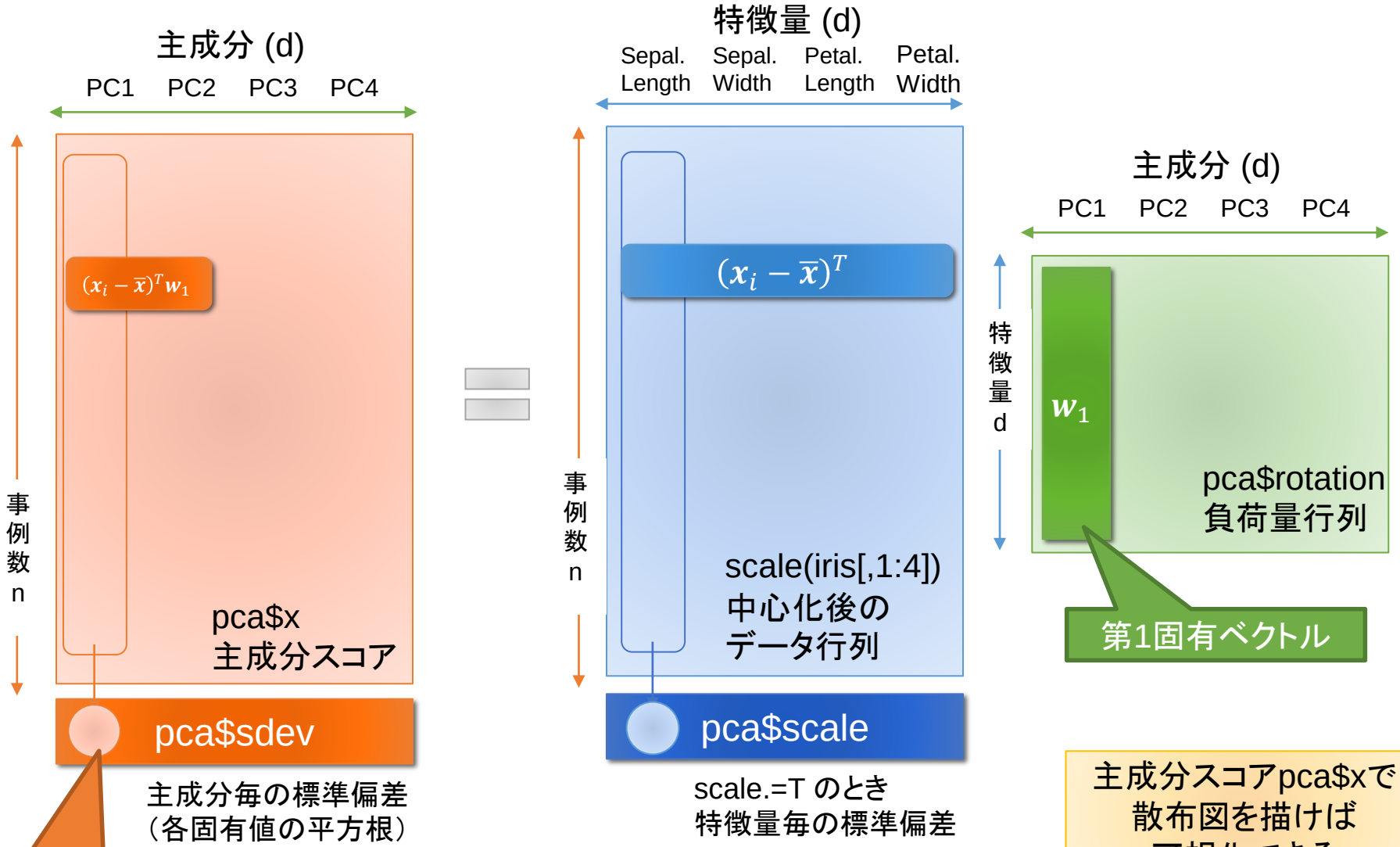
λ_1はSの固有値である。また、このとき

$$\boldsymbol{w}_1^T\mathrm{S}\boldsymbol{w}_1=\boldsymbol{w}_1^T\lambda_1\boldsymbol{w}_1=\lambda_1$$

であるため、分散は固有値に一致し、分散最大化問題は最大固有値を求める問題に帰着した

Rの主成分分析で得られるデータの構造

pca <- prcomp(iris[,1:4])
を例として



主成分スコアpca$xで
散布図を描けば
可視化できる

累積寄与率等

summary(pca)で以下の情報が得られる

- Standard deviation
  - 各主成分の標準偏差
- Proportion of Variance（寄与率）
  - 各主成分の固有値（分散）をその総和でわったもの
- Cumulative Proportion（累積寄与率）
  - 第1主成分から当該主成分までの寄与率の和

累積寄与率から、どの主成分まで採用すれば
全体の分散を十分に説明できるか判断する

教師なし学習
(クラスタリング)

クラスタリング（Clustering）

教師なし学習

データを類似性の高いグループに分ける

- ex. 顧客を収入や借入でグループに分ける，学生を成績や出席率によってグループに分ける等

クラスタリングのアルゴリズム

- k-平均法
- 自己組織化マップ

k-平均法の概要と事例

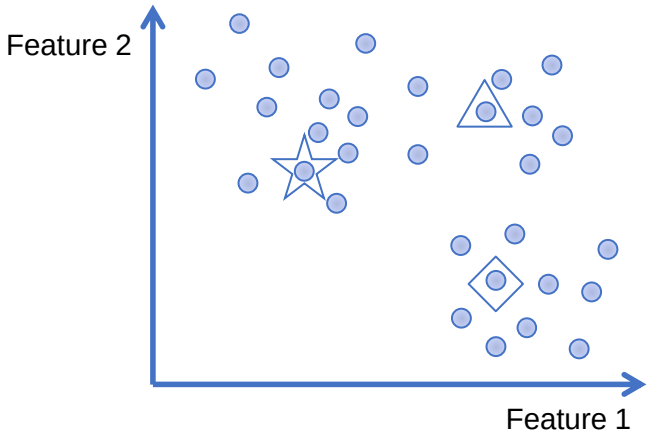
教師なし学習

特徴空間において距離が近いデータを同じクラスタとみなし、k個のクラスタを見出す手法

k-平均法のアルゴリズム(1/2)

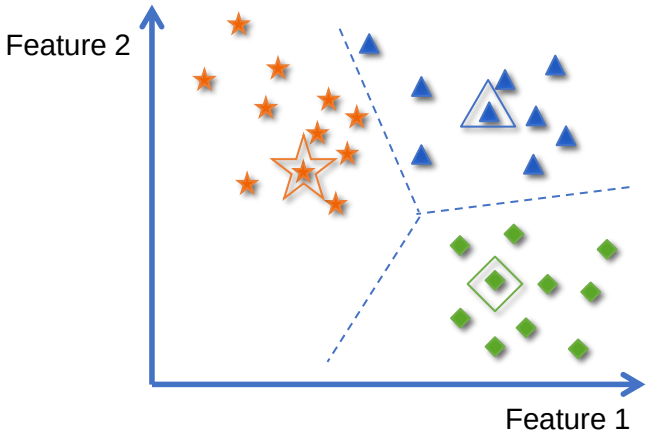
出典：Brett Lantz:「Rによる機械学習」翔泳社（2017）を基に鈴木作成

1



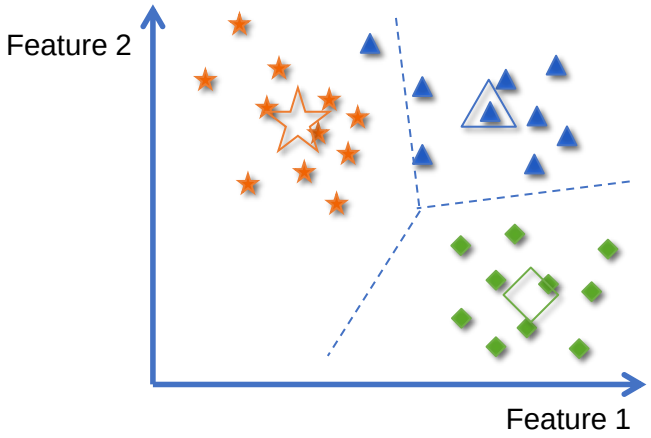
1. k個の点を訓練データセットから無作為に選ぶ（初期クラスタ中心）

2



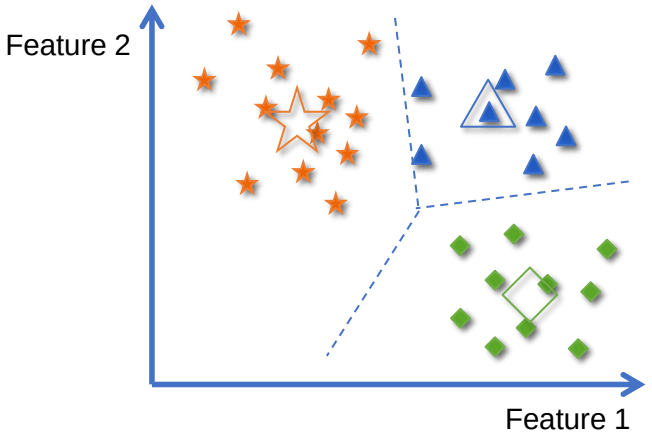
2. データ点毎に、クラスタ中心との距離を求め最も近いクラスタに割り振る

3



3. クラスタ毎に、その時点で含まれる点から重心を求め、新たなクラスタ中心とする

4

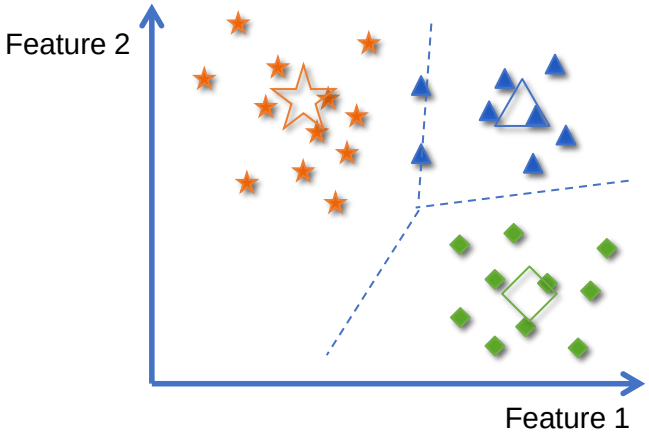


4. データ点毎に新たなクラスタ中心との距離を求め改めて割り振る

k-平均法のアルゴリズム(2/2)

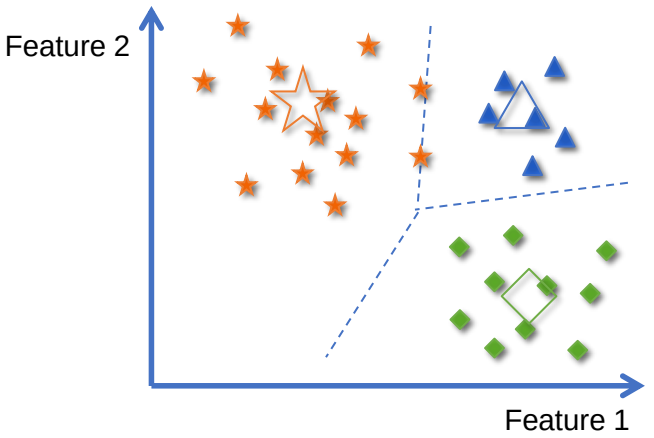
出典：Brett Lantz:「Rによる機械学習」翔泳社（2017）を基に鈴木作成

5



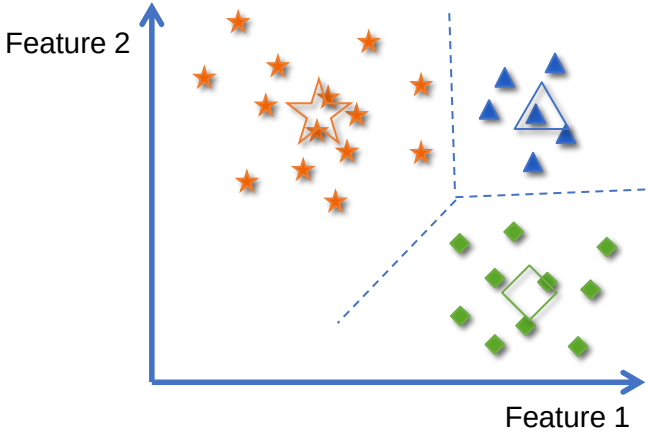
5. クラスタ毎に、その時点で含まれる点から重心を求め、新たなクラスタ中心とする

6



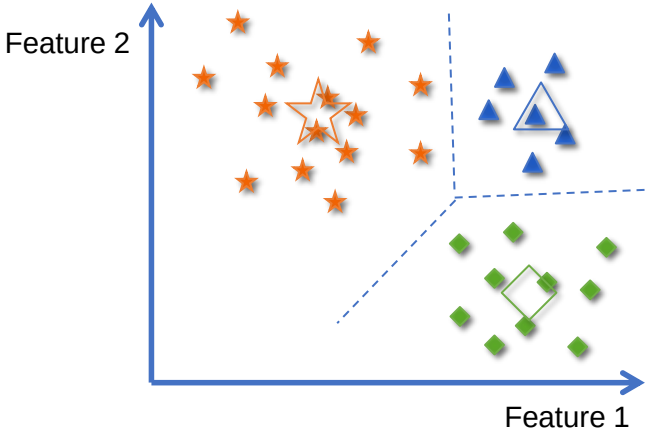
6. データ点毎に、クラスタ中心との距離を求め最も近いクラスタに割り振る

7



7. クラスタ毎に、その時点で含まれる点から重心を求め、新たなクラスタ中心とする

8



8.新たな割り振りが生じないため終了

k-平均法のアルゴリズム

1. クラスタ中心の初期設定
   - クラスタの中心となるk個の点を訓練データセットから無作為に選んで決める

2. データ点を最も近いクラスタへ割り振り
   - 各データ点について、クラスタ中心との距離を計算し、最も近いクラスタ中心を持つクラスタに割り振る

3. クラスタ中心の更新
   - クラスタ毎に、その時点で当該クラスタに含まれるすべての点から重心を求め、新たなクラスタ中心とする

4. Step.2とStep.3の繰り返し
   - クラスタ中心の更新にともない、所属クラスタが変わるデータ点が現れうる。Step.2 で各データ点について、新たなクラスタ中心との距離を計算し、最も近いクラスタへ改めて割り振る。割り振りに変化がない場合終了する

k-平均法適用上の注意(1/2)

初期クラスタ中心を無作為に選ぶため、結果が異なることがある

- 特徴空間全体から無作為な点を選ぶ方法、初期中心を選ばずにいきなり各インスタンスにクラスタを割りあてる方法などもあるが、各方法に依存したバイアスがある

距離関数を用いるため、事前にデータの正規化や標準化が必要である

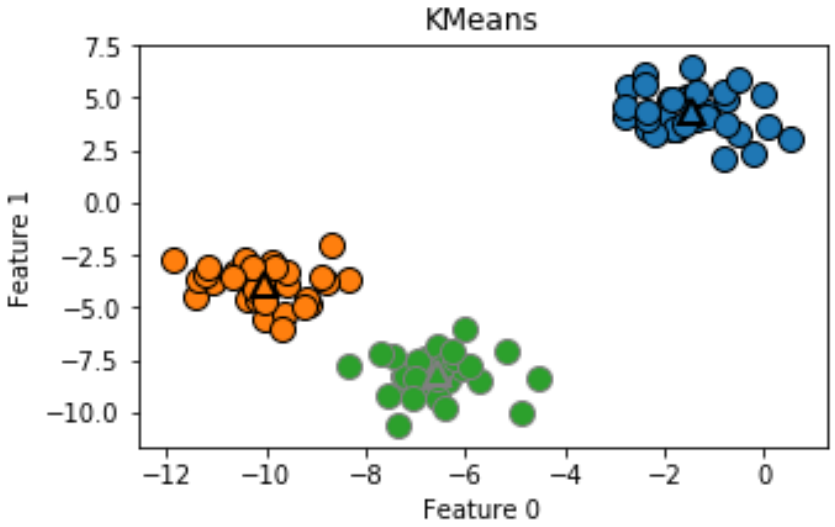
- ユークリッド距離を用いることが多いが、マンハッタン距離やミンコフスキー距離を用いることもある

クラスタ数kは分析者が決める必要がある

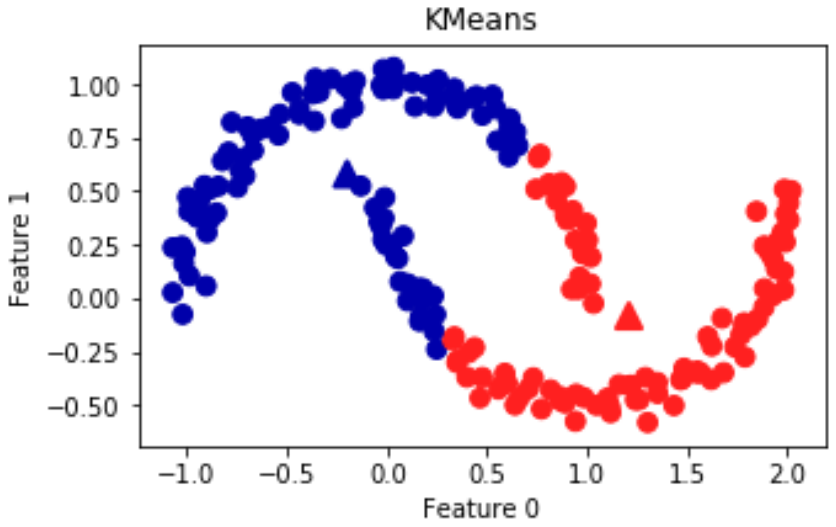
- いくつに分けるのが自然か、いくつに分けたいかで決める
- データに関する事前知識が無く目安が欲しい場合は$k = \sqrt{\frac{n}{2}}$とする方法もある
- kを増やしながらクラスタリングを試行し、グループ内の同質性の増加が鈍くなるkを採用する方法（elbow法）もあるが計算量過多

k-平均法適用上の注意(2/2)

適用可能なデータ
（丸い形状）



適用失敗するデータ
（複雑な形状）



- 各データ点は各クラスタ中心との距離に基づき、最も近いクラスタに割り当てられるため、複雑な形状のデータではクラスタリングに失敗する

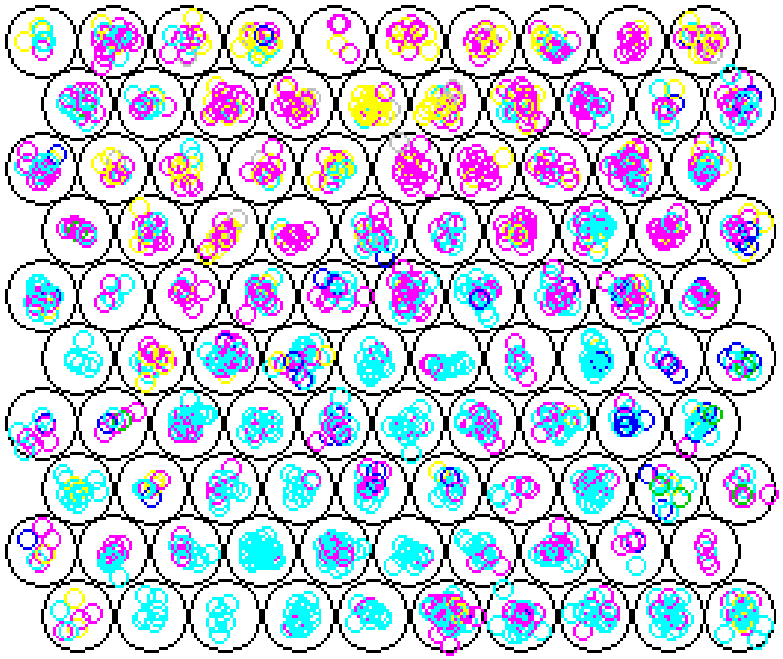
※データ出典：A. C. Muller et al. mglearn.datasets

自己組織化マップ（SOM）

教師なし学習（次元削減）

自己組織化マップ（SOM）の概要と例

高次元の特徴空間にあるインスタンスを2次元平面上に表現する手法

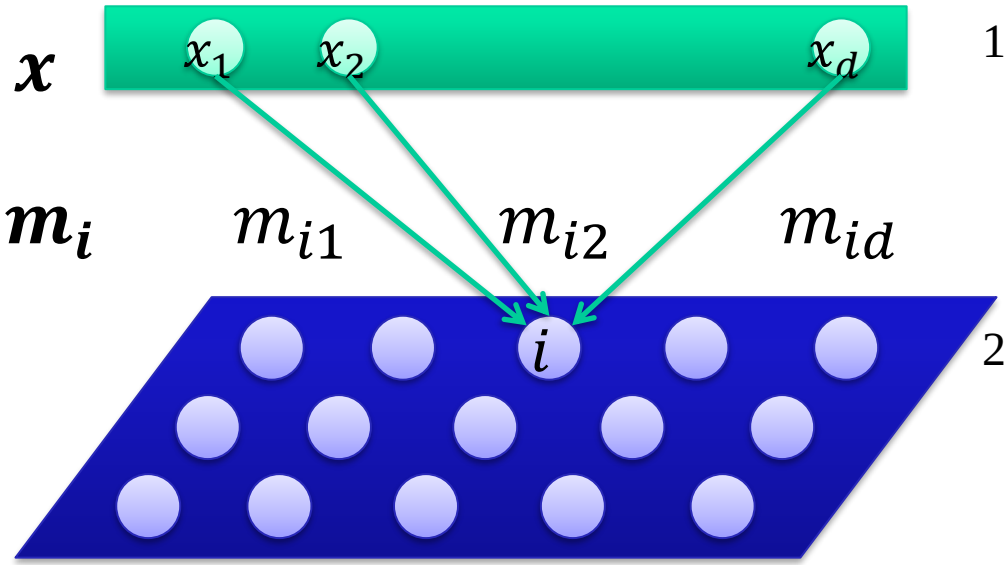


https://archive.ics.uci.edu/ml/datasets/Wine+Quality
赤ワインデータを正規化したものに対してKohonenを利用

- タスク事例
  - 11種類の科学的特性データであらわされる赤ワイン1,599品種を二次元平面上に可視化したい
  - 品質スコア（0から10の11段階）は学習に用いない（教師なし学習）
- 手法：SOM
  - 11次元特徴空間の1点となるひとつの品種は、2次元に並んだ10×10の出力ユニットのいずれかに対応付けられる
  - 黒丸は出力ユニット、中の小さい点ひとつがひとつの品種を表している
  - 科学的特性が互いに似ている品種は同じ出力ユニットか近くの出力ユニットに描画されることになるため、品種間の類似性の検討等に活用できる
- 結果の妥当性確認
  - 各品種の品質スコアを色で表現したところ、同じ色が同一か近傍のユニットに配置されている

自己組織化マップのアルゴリズム

入力層（d個のユニットからなる）



出力層（ 多数のユニットが二次元状に配置される）

入力層と出力層の間は全結合

1. 入力ベクトル$\boldsymbol{x}$と最も近い重みベクトル$\boldsymbol{m}_i$を持つ出力ユニットcを勝者とする

$$c = \arg\min_i |\boldsymbol{x} - \boldsymbol{m}_i|$$

2. 勝者ユニットやその近傍ユニットの重みベクトルを入力ベクトルに近付ける

$$\boldsymbol{\Delta m_i} = \boldsymbol{m}_i + \alpha\, h_{ci}(\boldsymbol{x} - \boldsymbol{m}_i)$$

$$h_{ci} = \exp\left(-\frac{|\boldsymbol{r}_c - \boldsymbol{r}_i|^2}{2\sigma^2}\right)$$

αは学習定数、h_{ci}は近傍関数（近傍ほど大きく、離れるほど小さい）

3. 入力ベクトルを更新して繰り返す

Rのsomパッケージ、kohonenパッケージなどで実装されている

第５章

従来法による画像・映像処理

今回の内容

- Google ColabによるOpenCV画像処理
- CondaによるPython環境構築
- 実機によるOpenCV画像処理

OpenCV

- 1999年よりIntelが開発している画像・映像処理ライブラリ
- Colabではプリインストール済み。Python環境では追加でインストールが必要
- BSDライセンスによるオープンソース
- 色変換、エッジ抽出、モザイク加工などの基本的なことから、顔や物体の検出、ラベリング、姿勢推定など様々な処理が可能
- チュートリアルも非常に豊富：
  https://docs.opencv.org/master/d9/df8/tutorial_root.html

Google ColabによるOpenCV画像処理

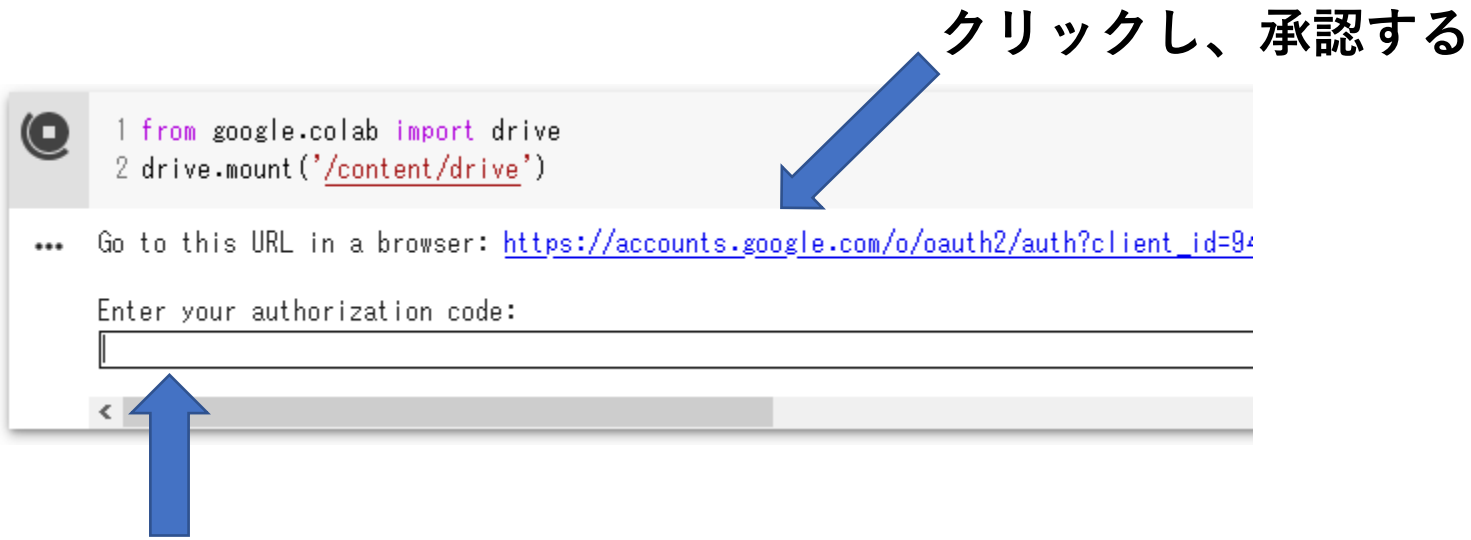
演習ファイルのColabフォルダ内をすべてColabにコピー

- Google Driveのマウント
- 画像の表示、回転、ぼかし、モザイク
- 顔や人の認識、猫の認識
- Webカメラの利用

ColabでのGoogle Driveのマウント

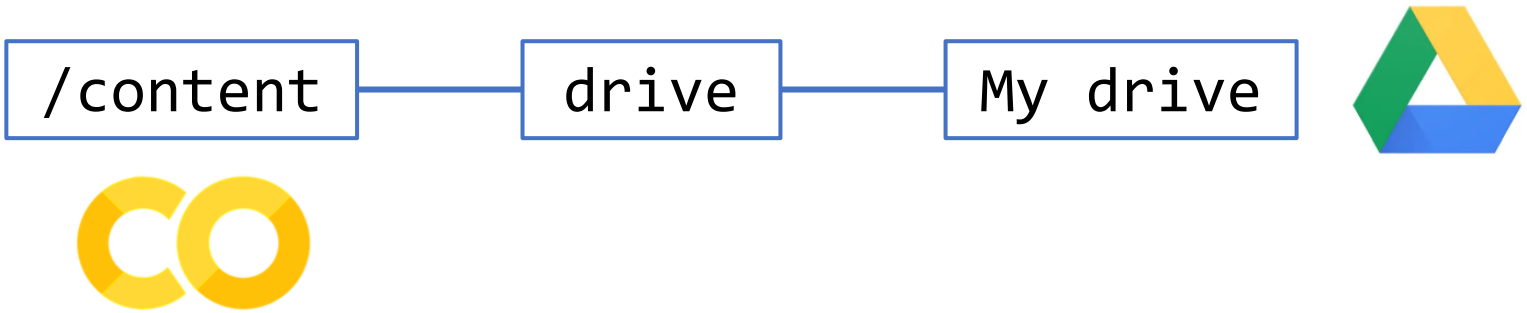
- 次のコードを実行し、認証コードを取得・入力

```
from google.colab import drive
drive.mount('/content/drive')
```



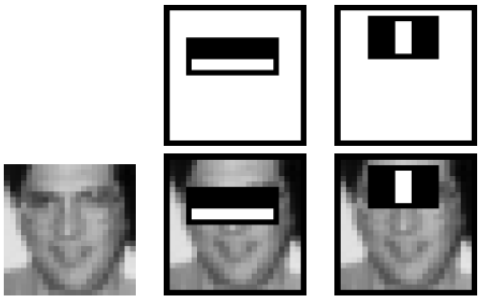
Google DriveとGoogle Colabのファイル

- 特にパスを指定しなければColabのVM上にファイルは保存
- パスを指定することで、DriveとColabに分けて保存可能
- もちろんColabに保存すると12時間後には消滅
- Colab: /content
- Drive: /content/drive/My drive



OpenCVによる画像識別

- ハール(Haar)特徴量にもとづくCascade 識別器を使用
- P. Viola (2001)によるオブジェクト検出の研究とR. Lienhart (2002)による改良がベースになっている
- 白と黒で構成される矩形を画像に適用し、対象画像の特徴量を作成する
- 例えば顔画像は、目の領域の画素は周辺よりも暗い、口の領域の画素が周辺より明るい、などの特徴が得られる
- 顔、目、正面、上半身、下半身、笑顔、などの識別器がOpenCVで使用できる



Sample face detection from, Paul Viola and Michael J. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features", IEEE CVPR, 2001.

CondaによるPython環境構築

- 別紙参照
- Minicondaをインストールし、Python環境を構築する
- Minicondaは、任意のバージョンのPython環境を構築できる一種の仮想実行環境
- Anaconda > Conda > Miniconda の順番でパッケージが大きいが、最低限の用途ならMinicondaがコンパクトで良い
- Minicondaをインストールし、Python環境を構築、必要なパッケージをpipコマンドでインストールする

PC上でのPythonの利用(Miniconda)

- Anaconda Promptから、`conda activate <環境名>`
- プロンプトが（base）から（環境名）となったことを確認できたら、`python <スクリプト>` で実行
- 遠隔受講でPythonからWebカメラを使用する場合は、必ずZoomのカメラをOFFにすること

PCで実行するファイル

開始前に、Colabフォルダ前のmodelフォルダをそのまま
PC内にコピーしておく

- 画像表示：image1.py, image2.py, image3.py
  - image1.py: システムの関連付けで表示
  - image2.py: Matplotlibで表示
  - image3.py: OpenCVで表示
- Webカメラ(モザイク、エッジ)：webcam.py
- 顔と目の検出(Webcam)：face_eye.py
- 顔領域の書き出し(Webcam)：face_file.py
- 人間検出(Webcam)：person.py
- 動体検出(Webcam)：detect.py

OpenCVによる人間の検出

- HoG特徴量＋SVMによる識別器
- 顔はだいたい誰も明暗差が変わらないが、人間全体だと服装によって明暗差が異なるため、輪郭情報を主に使用する
- HoG特徴量は、画像の輝度の勾配を主にパラメータとして用いる
- 複数人の検出も可能だが、精度はそれほどよくない＆かなり処理が重い
- getDaimlerPeopleDetectorとgetDefaultPeopleDetectorの2つの処理方法があり、ソースによってどちらが向いているか若干異なる

OpenCVによる動体検出

- Webカメラからの入力のうち、動いている部分を検出
- フレーム間差分を計算し、差が大きいところが動いている部分になる

- 参考：
  https://ensekitt.hatenablog.com/entry/2018/06/11/200000

参考：自分で識別器を作成

- 一種の教師あり学習
- 正解画像と不正解画像を用意し、モデルを構築

例：

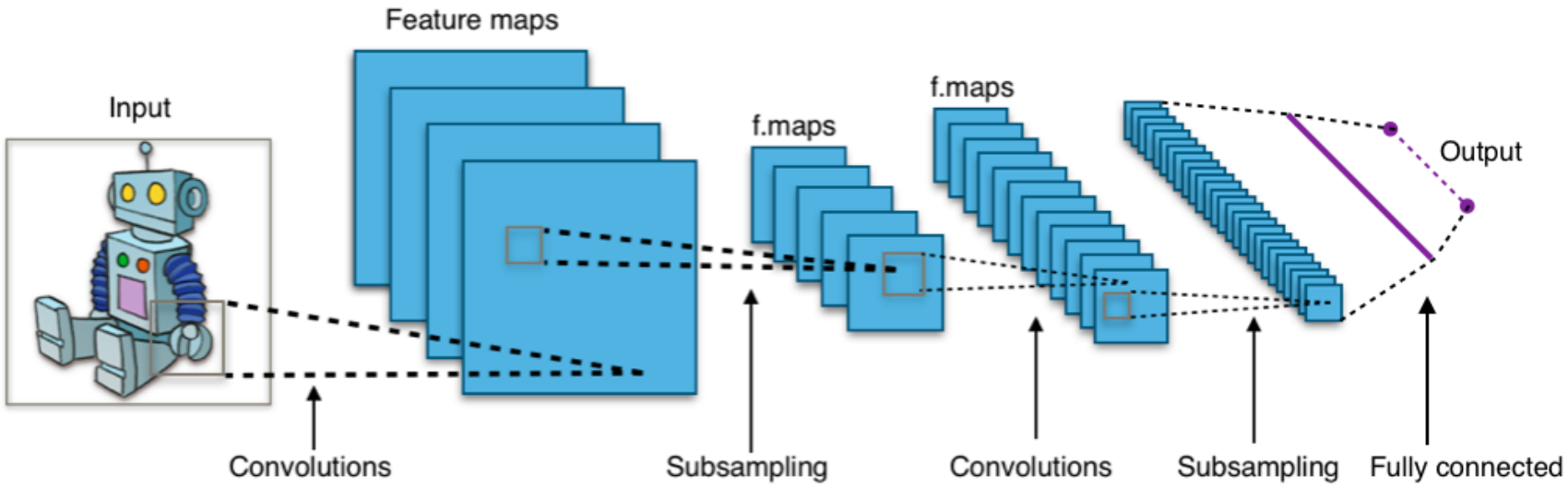
- https://premium.aidemy.net/magazine/entry/2018/05/21/210100
- https://qiita.com/penstood/items/97421a91d3f4075d39a4

第6章

CNN
(Convolutional Neural Network)
と画像の水増し
(Image Augmentation)

Convolutional Neural Network (CNN) の基礎知識

- CNNの一般的な構造



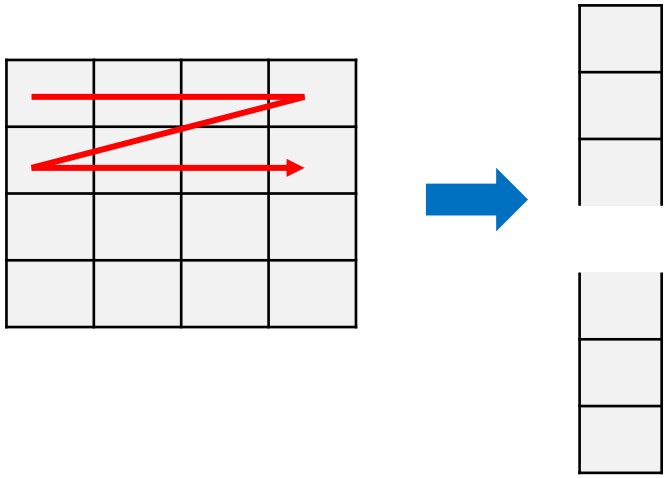
https://en.wikipedia.org/wiki/Convolutional_neural_network

- 画像データの処理
  - 畳み込み（Convolution）
  - プーリング（Pooling）（Subsamplingの一種）

画像情報の弱点

ニューラルネットワークへの入力は1次元の形にするため、2次元の画像情報が1次元になってしまう　→　画像の特徴が失われる

特徴を活かす形で入力できないといけない　→　解決策の1つがCNN

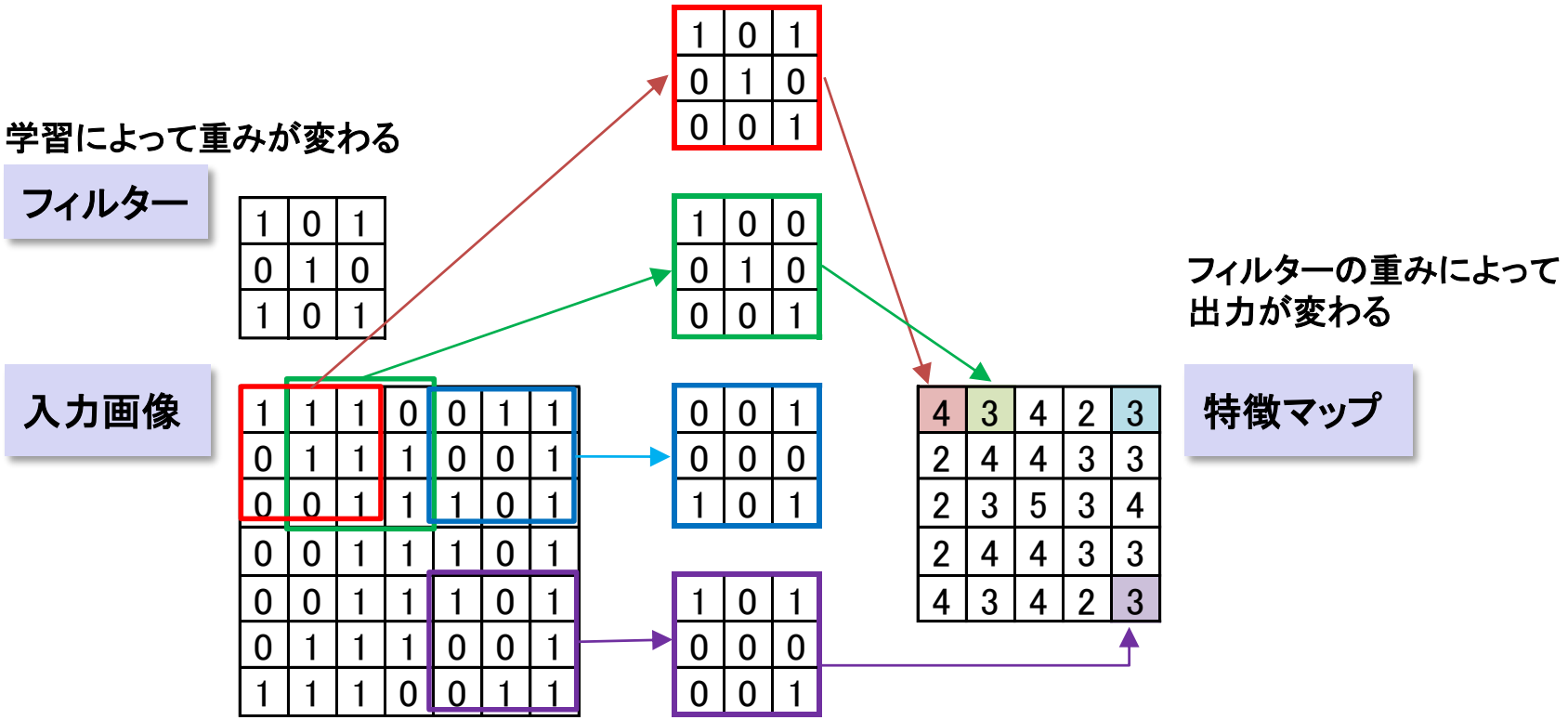


畳み込み（Convolution）

- 画像に小さなフィルターをかけ、その値をピクセル値として新たな画像を生成する
- 参考：https://ja.wikipedia.org/wiki/%E7%95%B3%E3%81%BF%E8%BE%BC%E3%81%BF (Wikipedia: 畳み込み)
- フィルターの重みを学習によって変化させることで、特徴を表す画像を生成する
  - 畳み込みで生成された特徴画像は特徴マップとも呼ばれる
- 後述するパディング、ストライドの大きさによって生成される画像のサイズが異なる

畳み込み（Convolution）

- 入力データに対してフィルターをかけることで、特徴マップを作成する
- 例：カーネルサイズ：3×3、ストライド：横1、縦1



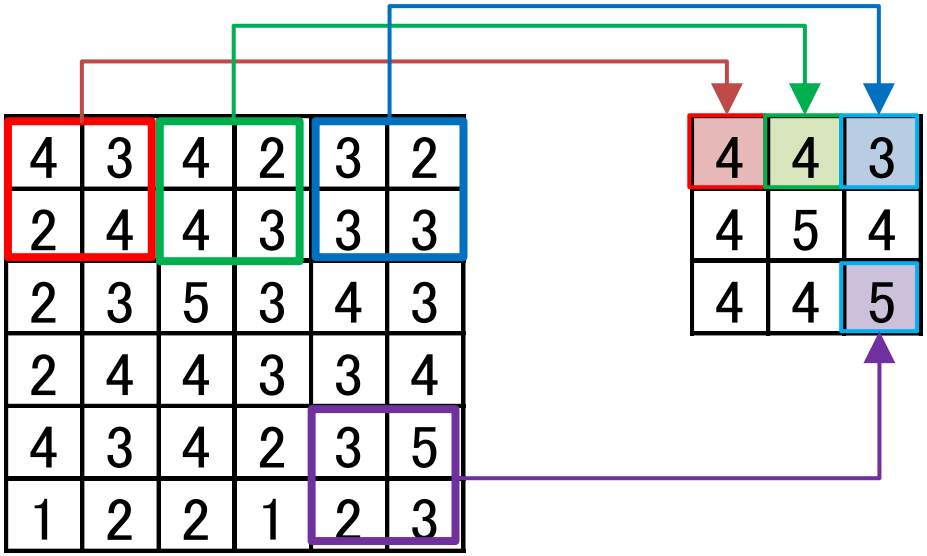
7_1_画像の畳み込みとプーリング.ipynbを参照

プーリング（Pooling）

- サブサンプリングの一種
- ある領域ごとに代表値を求め、その値をピクセル値として新たに画像を生成する
- 結果として画像サイズが小さくなる（解像度が低くなる）
  - Max pooling：最大値を代表値とする
  - Average pooling：平均値を代表値とする
  - Sum pooling：合計値を代表値とする
- 解像度が低くなることで、特徴の位置の多少のずれに頑健になる
- 領域の大きさによって、縮小率が変わる
  - 領域が2×2、ストライド（後述）が2の場合、サイズは縦1/2、横1/2の1/4になる。

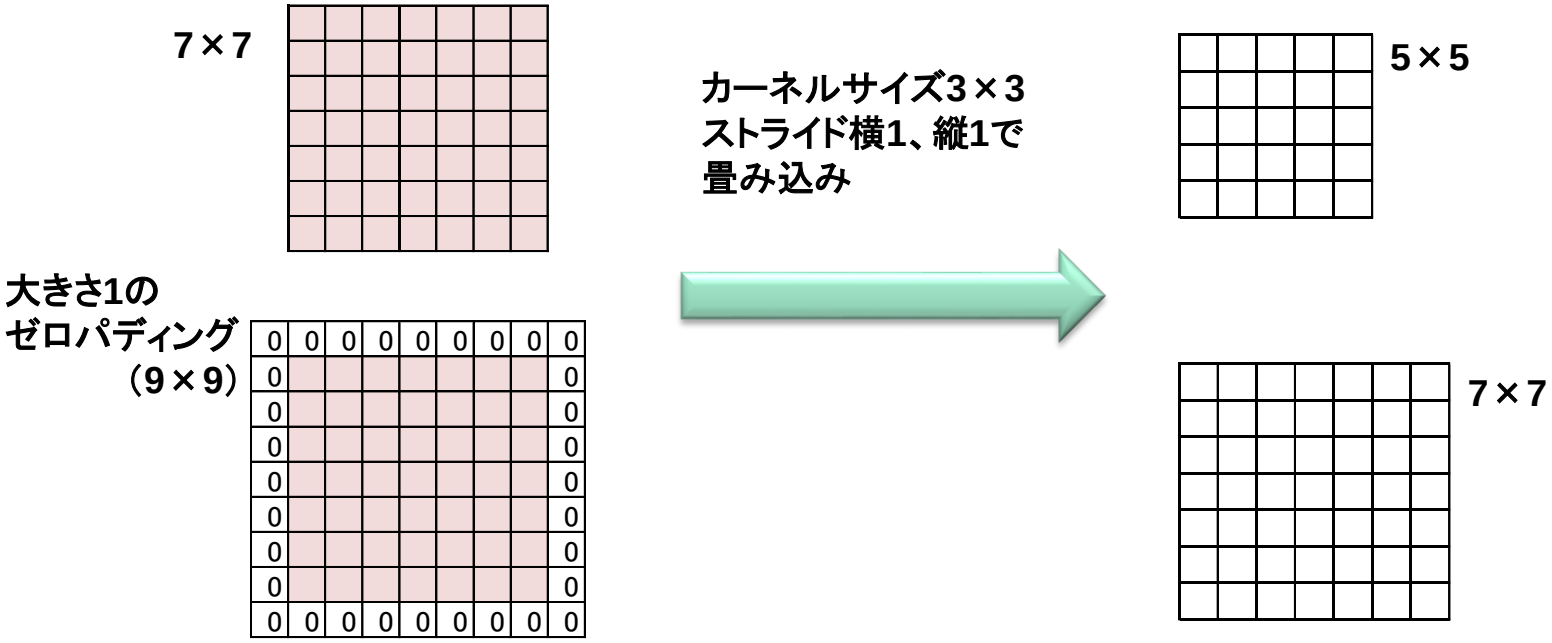
プーリング（Pooling）

- Max poolingの例
- ウィンドウサイズ（カーネルサイズ）：2×2
- ストライド：横2、縦2



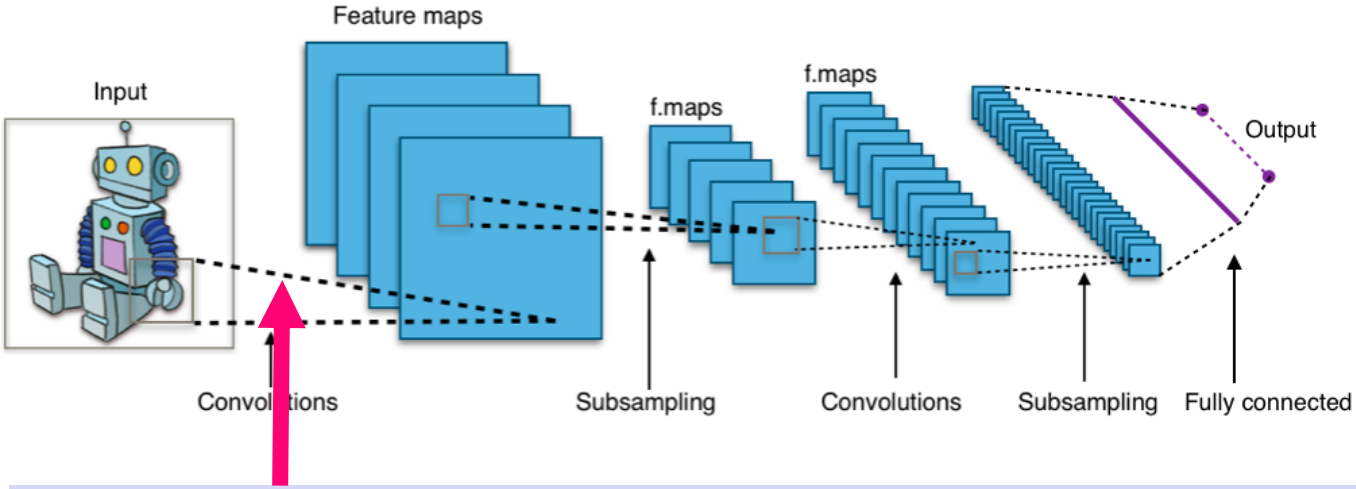
パディング（Padding）

- 畳み込みを行うと、生成される画像サイズは、元の画像よりも小さくなる
- 元の画像の外側を仮に何らかの値で埋め「ふち」をつくることで、生成される画像サイズを調整する
- 0で「ふち」を埋めるゼロパディングが良く使われる



畳み込みの意義

- 入力画像から目的にマッチした**特徴量を学習によって生成**する
  - 畳み込みフィルターの重みを学習によって決定する
- 中間層が全結合ではないので、計算量が少なくなる



もし全結合にすると、入力画素数と特徴マップの画素数×特徴マップ数の結合が必要

https://en.wikipedia.org/wiki/Convolutional_neural_network

CNNとDNNの比較演習

7_2_MNIST分類.ipynbを参照

演習のねらい

- KerasによるCNNの基本的な実装
- DNNとCNNによる分類の精度比較

精度の改善

CNNの表現力増加
- Convolution層やAffine層を深くする
- Convolution層の特徴マップ（OutMaps）数を増やす
- Affine層のOutput数を増やす
- KernelShape（フィルターサイズ）を小さくする

勾配消失の回避
- 層を深くすると勾配消失により誤差が小さくても精度が悪くなる
- 活性化関数をReLUにすると勾配損失を防ぐことがある

過剰適合の回避
- 層を増やしすぎない
- DropOut層を入れる（ランダムにノードを無効化し表現力を落とす）

BatchNormalization層を入れる
- 一部の入力のみの影響が大きくなりすぎることを防ぐ

自作データでの検証

7_3_自作手書き文字認識.ipynbを参照

演習のねらい

分類モデルが自作の手書き文字（準備されたデータ以外のデータ）でも分類できる汎用的な分類モデルであることを確認する

手書き数字データ

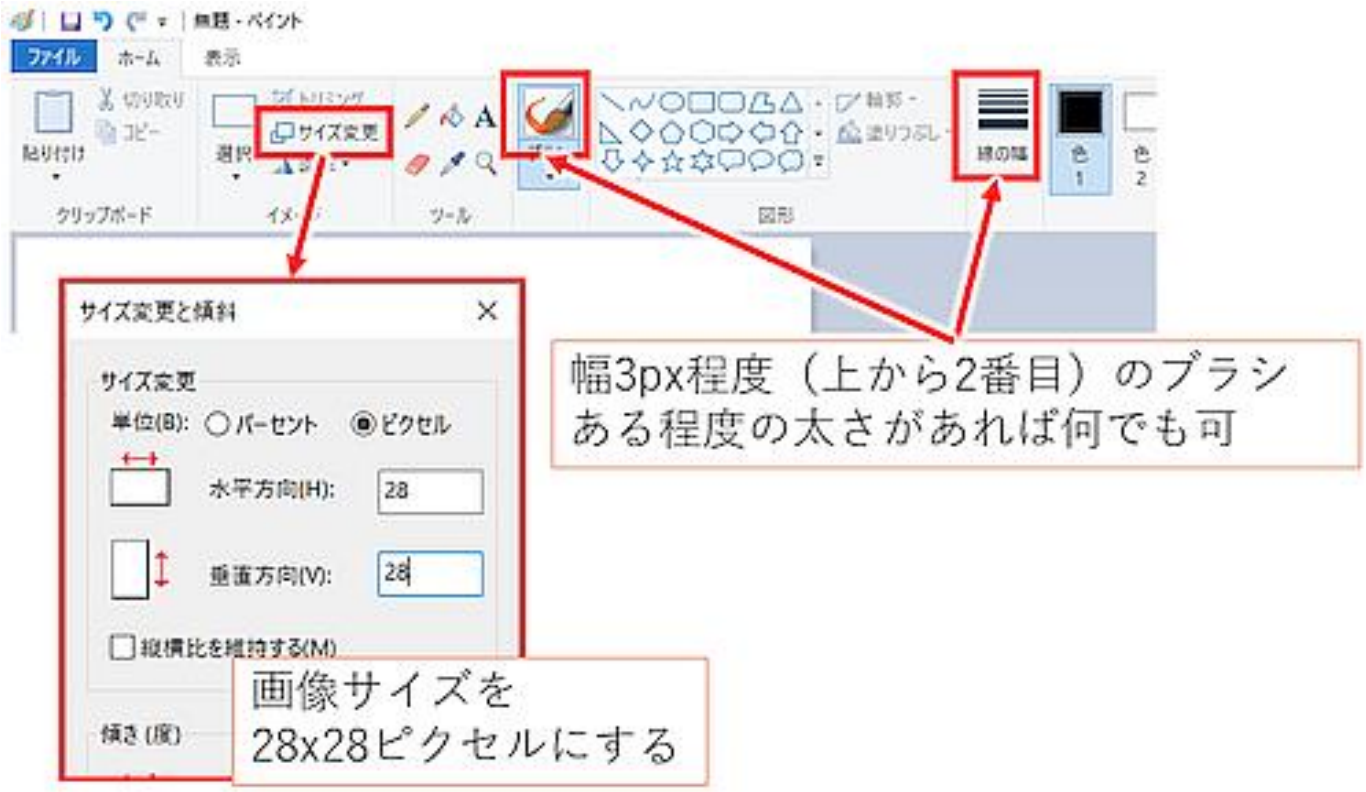
- 背景が黒で文字が白のグレイスケール画像
- サイズは28×28

画像データ作成

- ペイントソフトなどで文字を書く
  - ひとまず今回はペイントソフトで文字を書きます
- 紙に書いた文字をスキャナやカメラで画像化する

自作手書き画像の作成

ペイントを起動し、画像サイズを28x28に変更する
マウスで数字を書く（幅3px程度）
jpgとして保存したあと、GoogleDriveにアップロードする



画像データの水増し(Image augmentation)

学習には、画像内の対象の差異がある大量の画像データが必要
対象の差異の種類（手書き数字の場合）

- 文字の形
- 画像内での位置
- 画像内での文字が閉める割合（サイズ）
- アスペクト比
- 回転、反転
- その他変形
- 明るさ、コントラスト、彩度

傾きやサイズ、アスペクト比などが学習データに無い場合、精度が落ちる
画像処理で生成できるものも多い

- 一つの画像に様々な画像処理を組合せて適用し、学習データを増やす
- 予測の安定性につながる

Kerasによる画像の水増し

keras.preprocessing.image.ImageDataGeneratorの引数は次のとおり

- rotation_range:回転（degree）
- width_shift_range：横方向の移動（ピクセル）
- height_shift_range：縦方向の移動（ピクセル）
- shear_range：歪みの角度（degree）
- zoom_range：スケーリングの範囲（等倍=1.0）
- horizontal_flip：横方向反転（True or False）
- vertical_flip：縦方向反転（True or False）

画像データ水増し演習（1）

7_4_画像の水増し.ipynbを参照

演習のねらい

- 基本的な画像処理により水増しされる画像が、どのような画像になっているかを確認する
  - パラメータを一つずつ変更して、画像処理による加工の様子を確認する

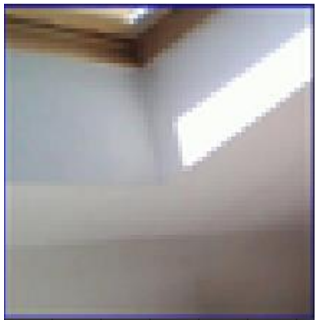
画像データ水増し演習（2）

7_5_ASL認識モデル.ipynbを参照

演習のねらい
データ水増しにより分類精度が上がるケースを体験する
CNNを改良して分類精度の向上を試みる

画像データ水増し演習（2）での学習データについて

- ASL画像データは下のような画像が、画像内の手の位置が少しずつ移動したり、画像内での手の大きさが少しずつ異なる（カメラからの距離が変わる）ような画像です。1種類につき3000枚ずつあります。
- 学習データはこれらの画像から10枚ずつ抜き出したものです。つまり、ある特定の手の位置や手の大きさの10枚の画像と言うことになります。
- CNNによる分類では、対象の位置や大きさが異なれば、それは異なる画像となります。
- 画像処理によって様々な位置や大きさの手の画像を作ることで、分類精度の向上が期待できます。



ブランク

ラベル:0

A

ラベル:1

B

ラベル:2

C

ラベル:3