

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）

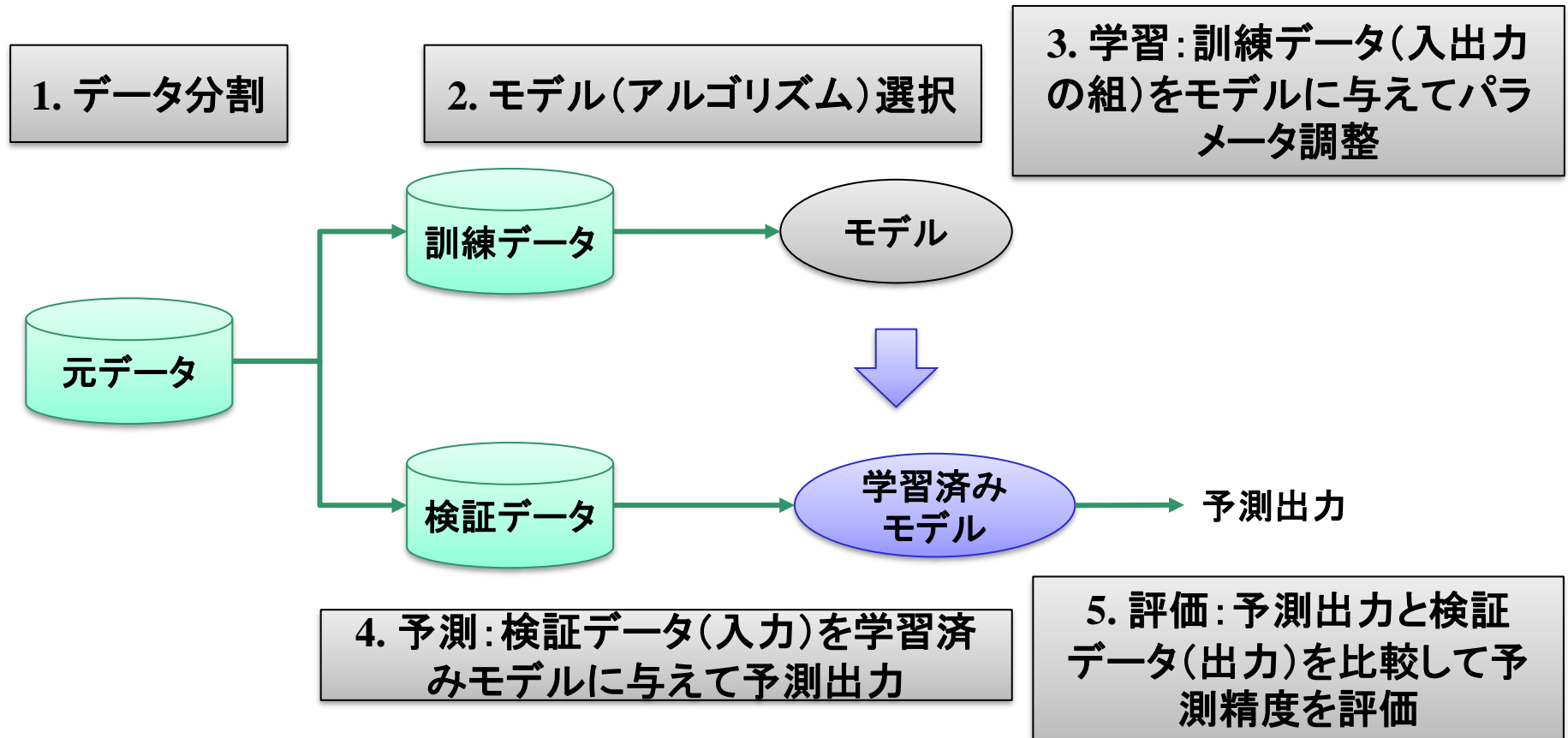
強化学習（本授業の範囲外）

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

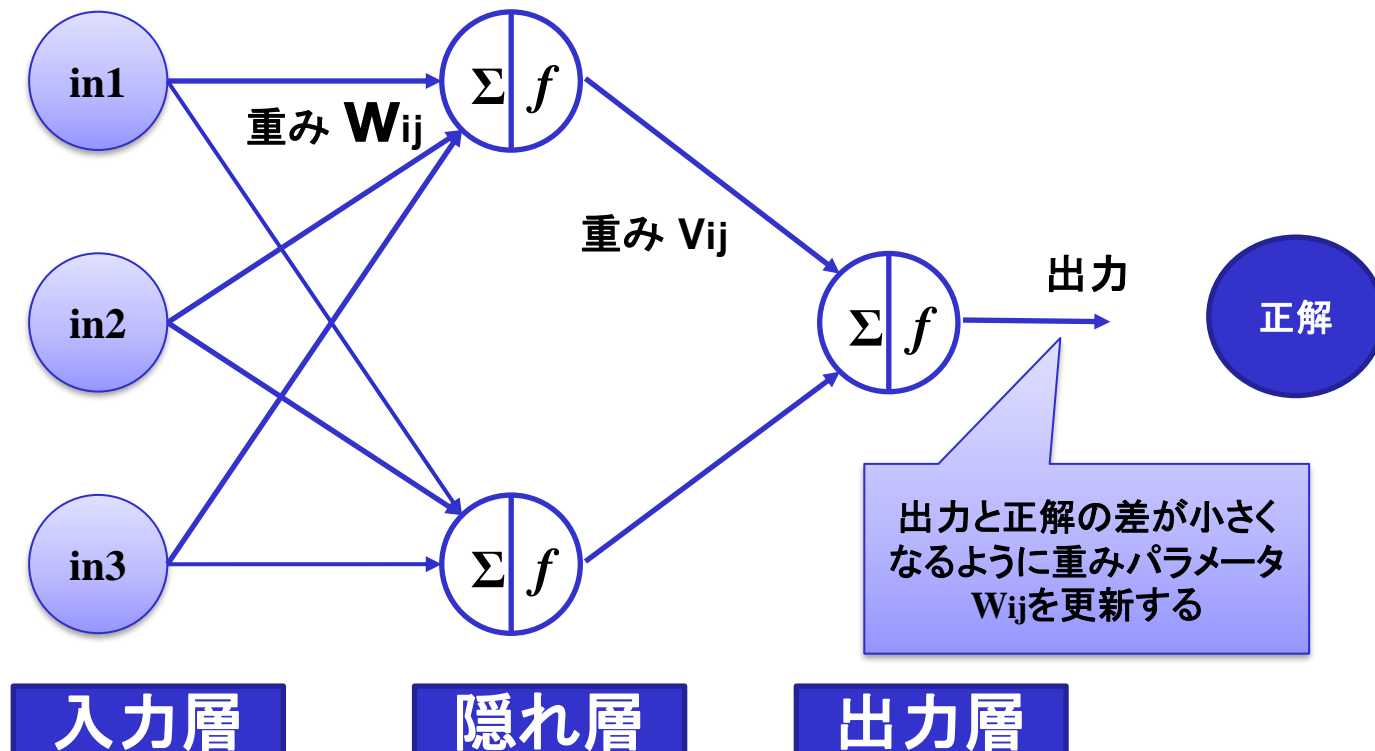
- クラスタリング
 - k平均法
- 次元削減
 - PCA、自己組織化マップ

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

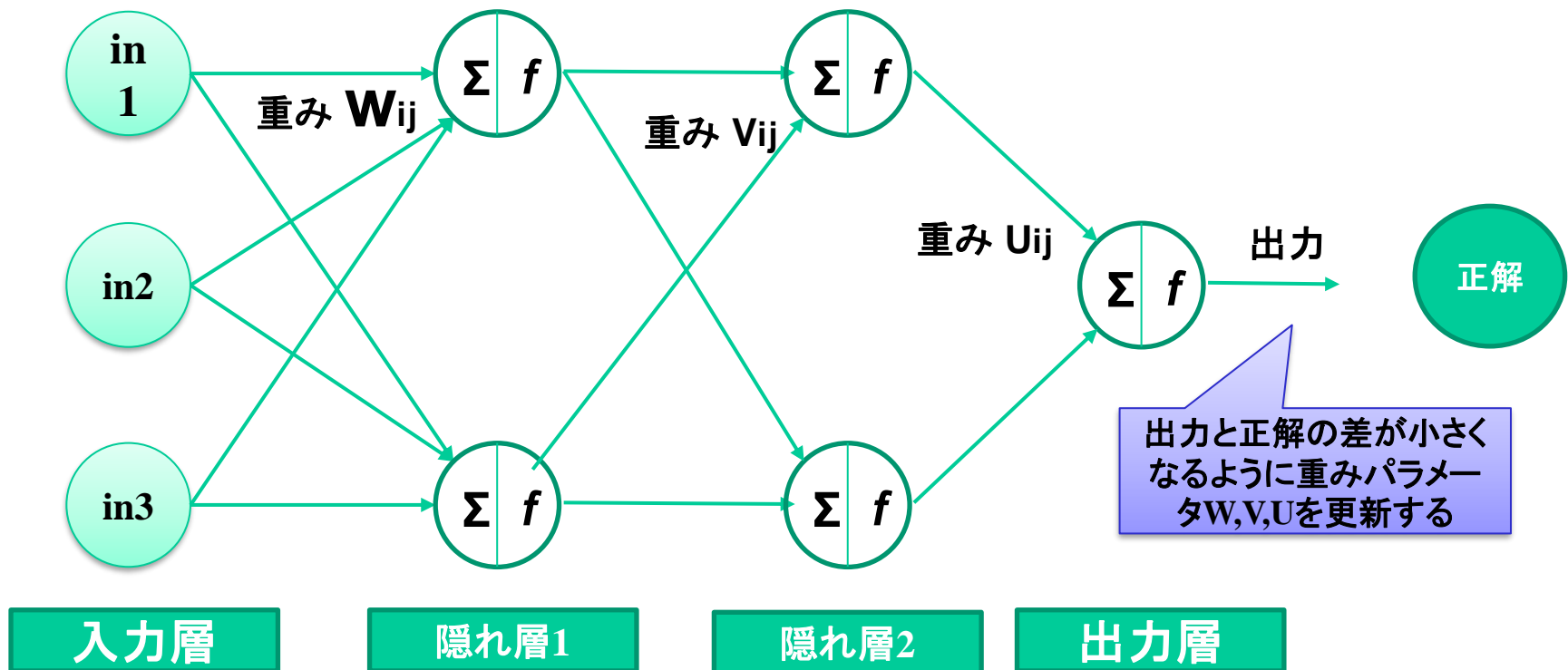
脳の神経回路の仕組みを真似たモデル．機械学習アルゴリズムの一種
入力と対応する正解出力の組からなる教師データを与える
出力データと正解データの差が小さくなるように重みパラメータを更新する



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

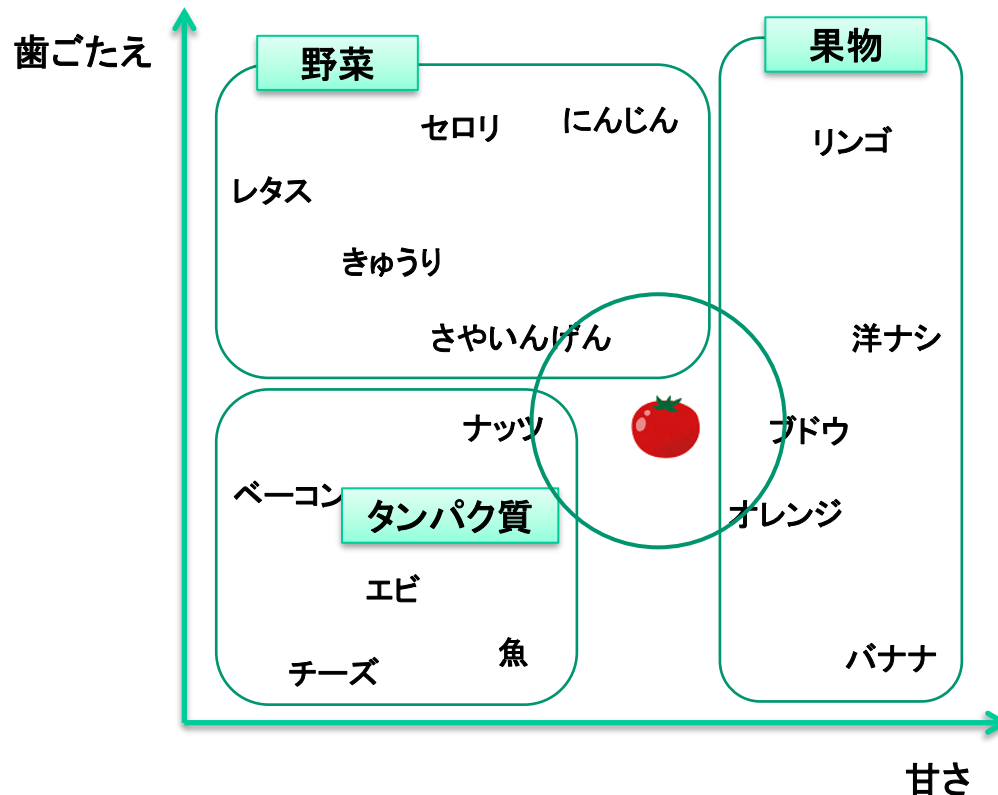
ex. 不動産価格の予測（回帰）、検査結果に基づく陽性判定（分類）

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!

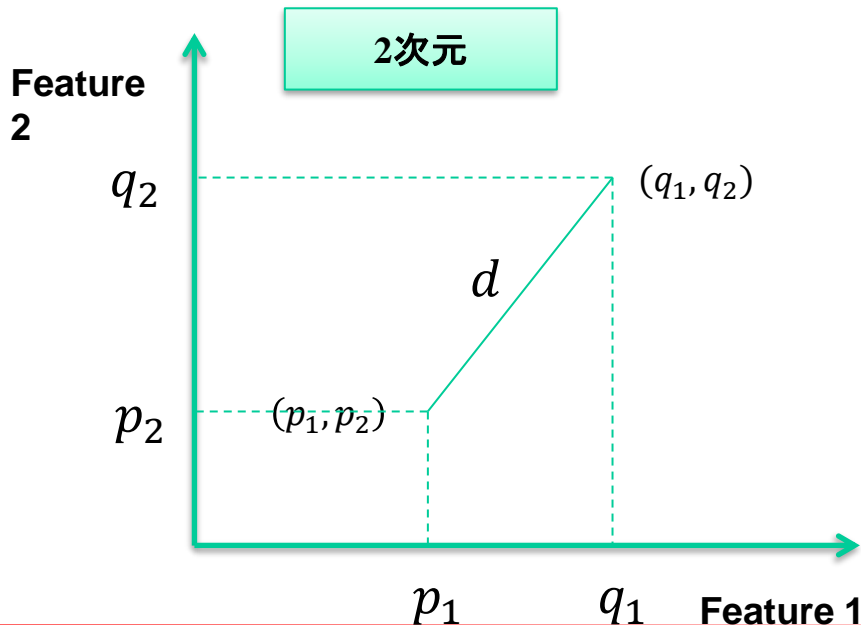
出典: Brett Lantz: 「Rによる機械学習」 翔泳社
(2017)を基に鈴木作成

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

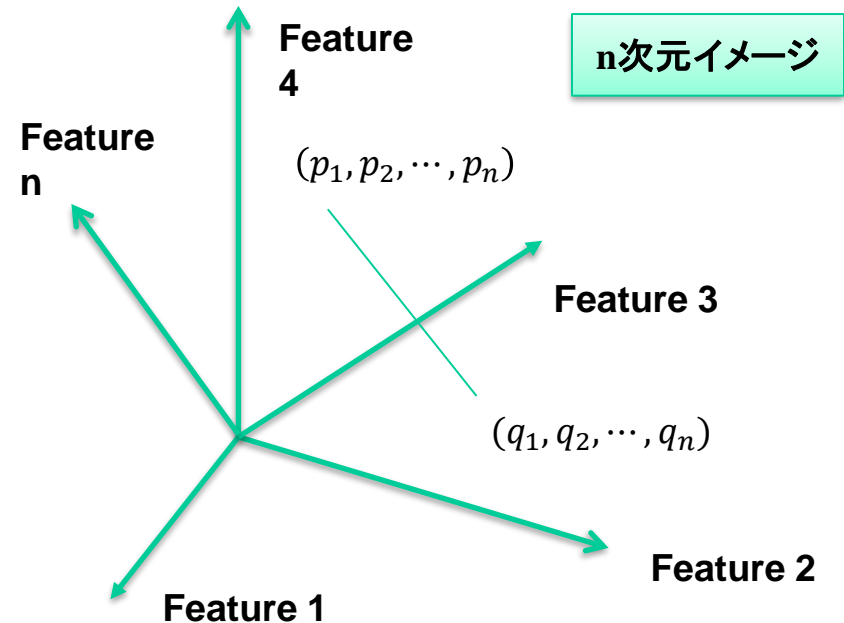
$$d = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2}$$



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 $(p_1, p_2, \dots, p_n), (q_1, q_2, \dots, q_n)$ 間の距離

$$d = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \dots + (p_n - q_n)^2}$$

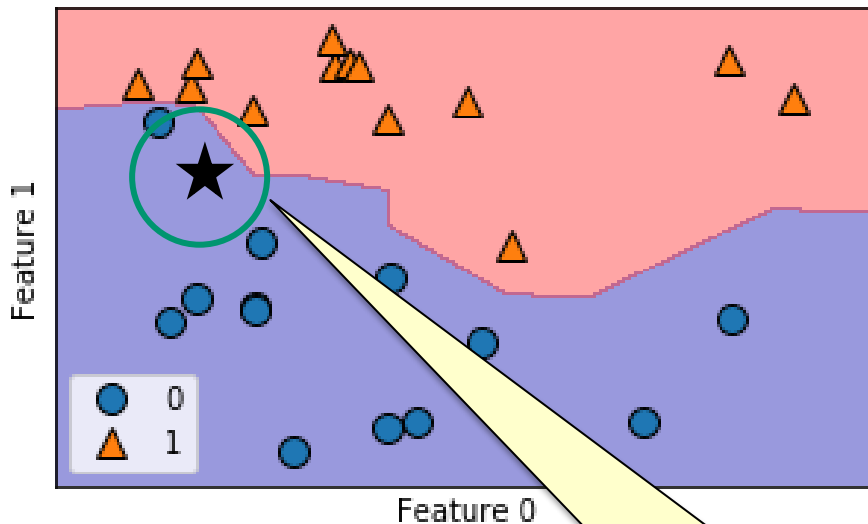


k-近傍法による予測方法とkの値による違い

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

1-NN

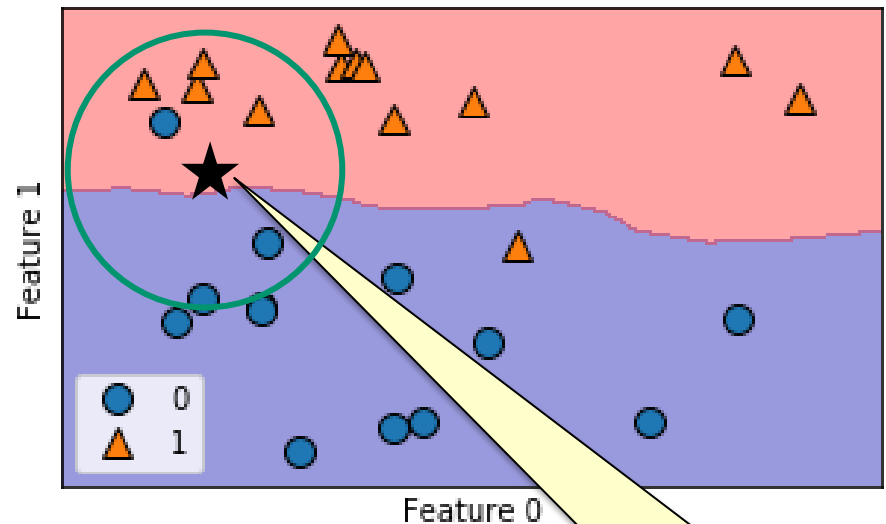
1 neighbor



★の最近傍点は●
⇒ ★は●と同じクラスと予測

7-NN

7 neighbor(s)



★の7つの近傍点は
▲が4、●が3
⇒ ★は▲と同じクラスと予測
(多数決)

- 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

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

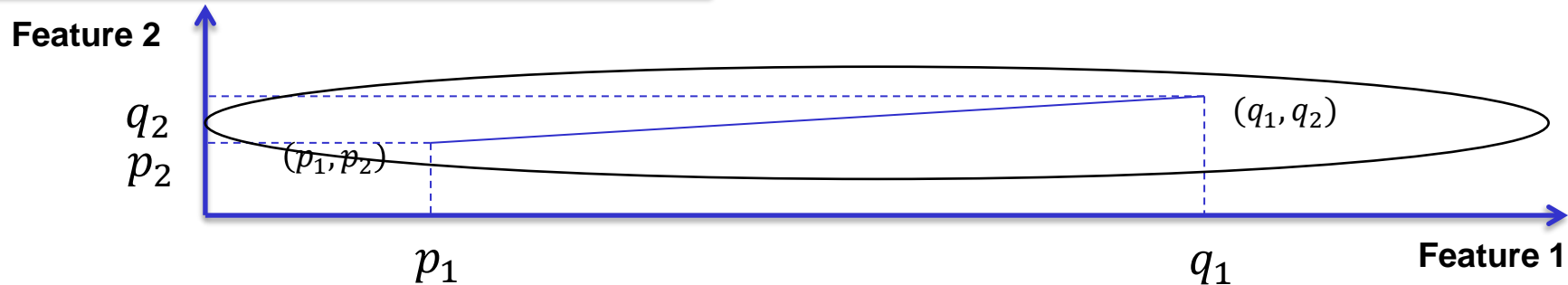
1. 未分類のデータ点1個に対して、分類済みの近傍データ点をk個見つける
2. 未分類データ点は、分類済み近傍データ点の多数派が所属するクラスに分類する

実装

- Rのclassパッケージに含まれるknn実装など

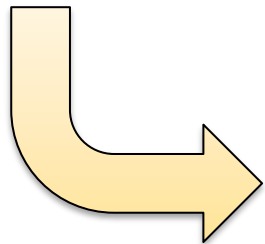
k近傍法で用いるデータの前処理の必要性

特徴量間で値の範囲が大きく異なる場合

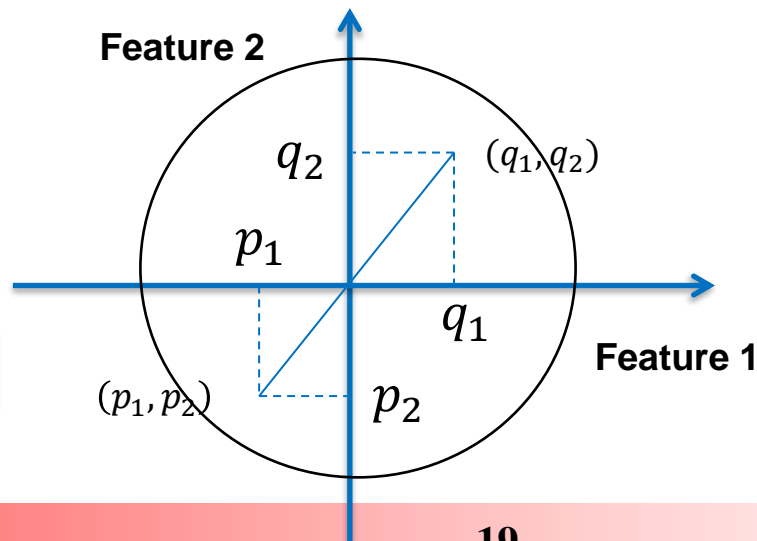


$$\sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2} \sim \sqrt{(p_1 - q_1)^2} = |p_1 - q_1|$$

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



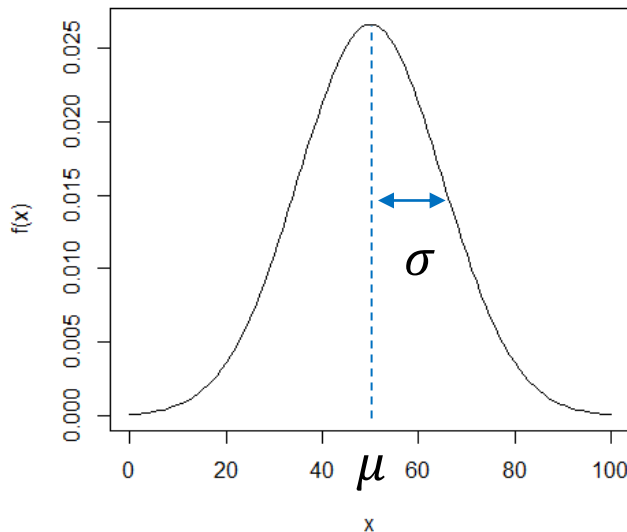
正規化・標準化




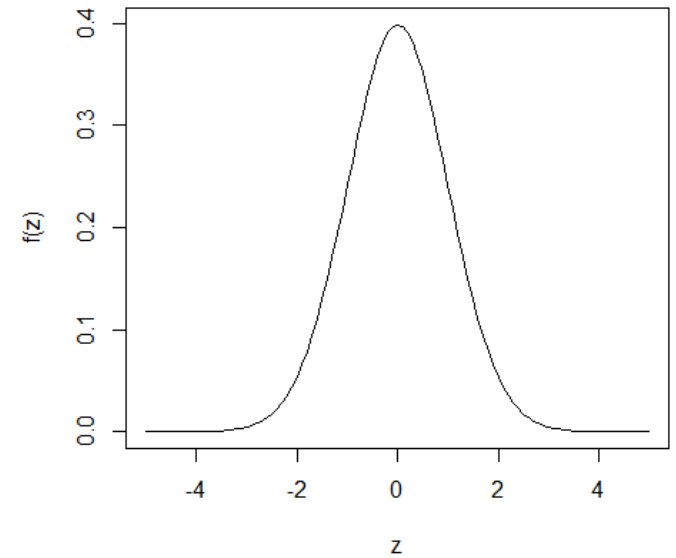
すべての特徴量について
正規化・標準化を行い、
どの特徴量も同等に評価
されるようにする必要がある

Zスコア標準化・最大最小正規化

Zスコア標準化



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


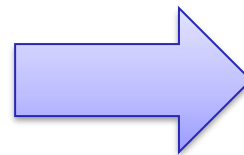


← 値の範囲が広すぎる →

← 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

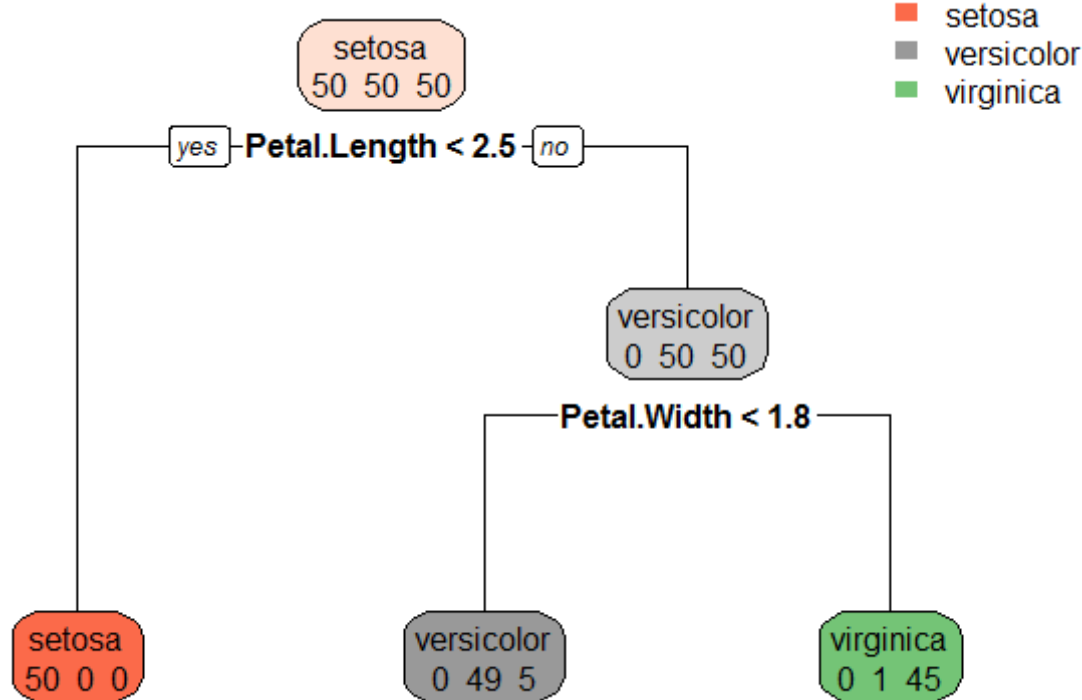
1. 目的変数の値が未知のデータ点1個に対して、特徴空間における近傍データ点をk個見つける
2. k個の近傍データ点の目的変数の平均を、注目データ点の目的変数の値とする

決定木

教師あり学習（分類）

決定木の概要と例

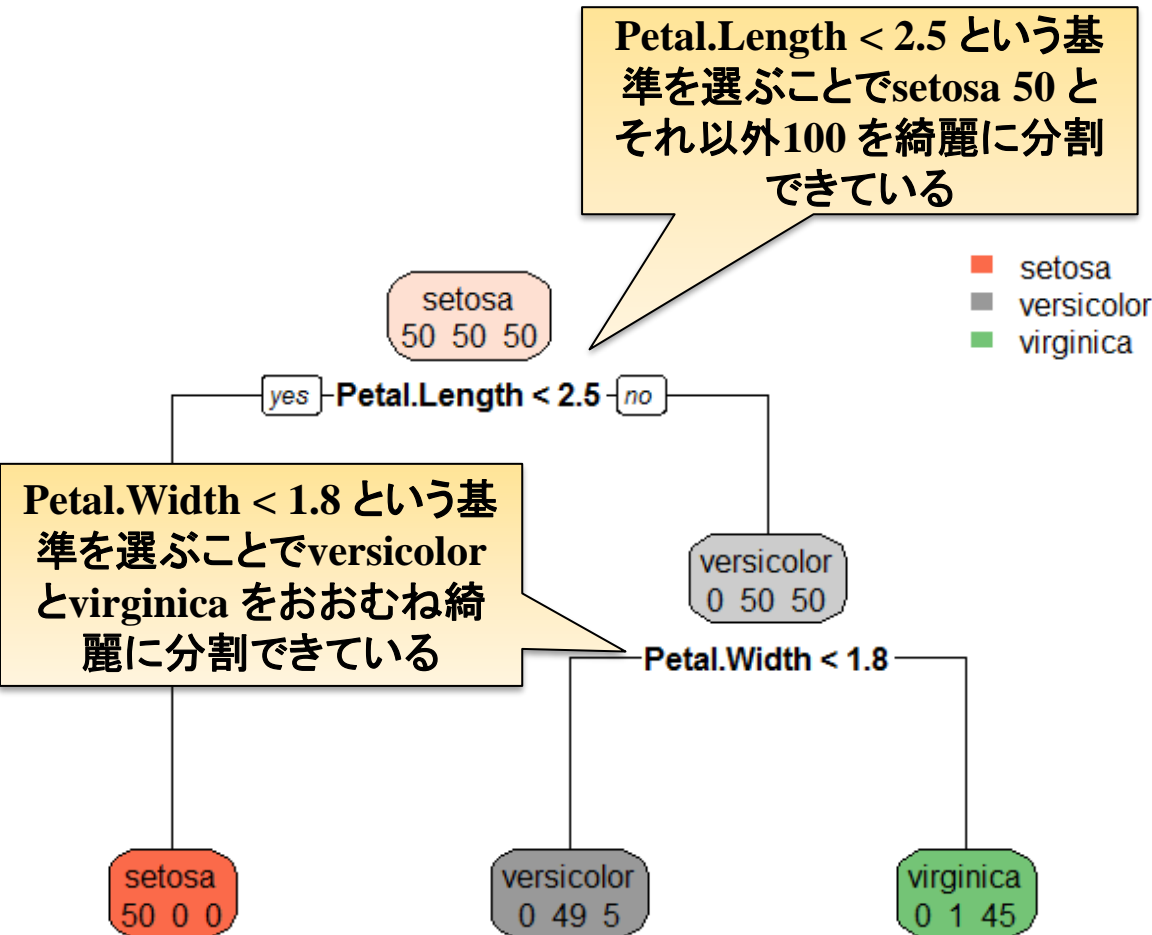
決定木：特徴量に関する条件分岐をたどると最終的に分類が決定する木構造



例：アヤメの種類を萼と花卉の長さと幅から決定する決定木

- Petal(花卉)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

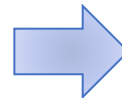
– ノード N_t のエントロピー

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

- 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



ノード N_t のエントロピー

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

ただし $\sum_{i=1}^n p_i = 1$

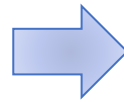
エントロピーの性質 (2クラス)

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

$n = 2$

ノード N_t

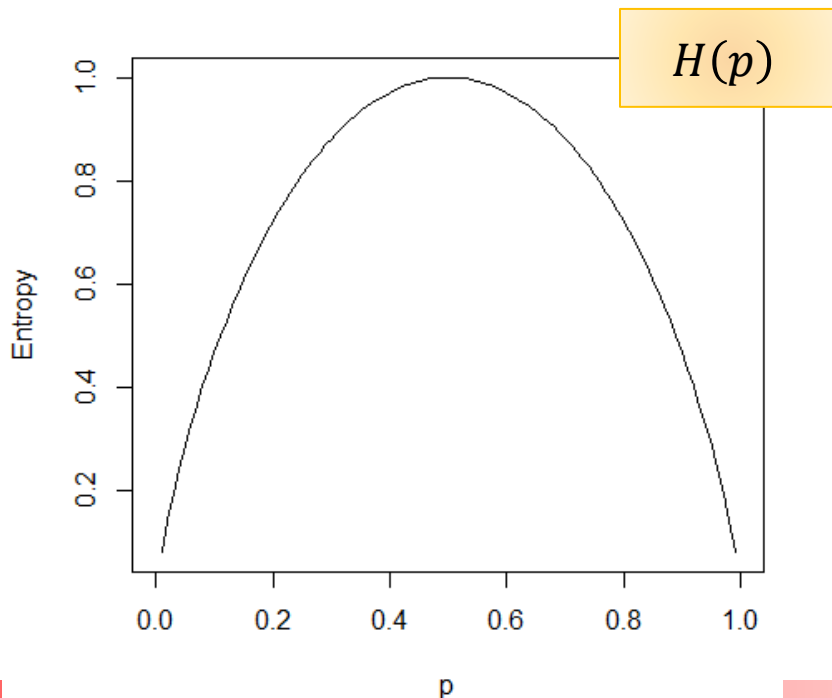
クラス1: $p_1 = p$
 クラス2: $p_2 = 1 - p$



ノード N_t のエントロピー

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

※一方のクラスの割合を p とおいた



エントロピーの性質 (2クラス)

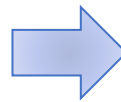
- $p = \frac{1}{2}$ で $H\left(\frac{1}{2}\right) = 1.0$ (最大値)
 - データの割合が50:50 のときエントロピー最大
- $p \rightarrow 0$ or $p \rightarrow 1$ で $H(p) \rightarrow 0$
 - いずれか一方のクラスしかない場合エントロピーは0

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



ノード N_t のエントロピー

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

ただし $\sum_{i=1}^n p_i = 1$

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

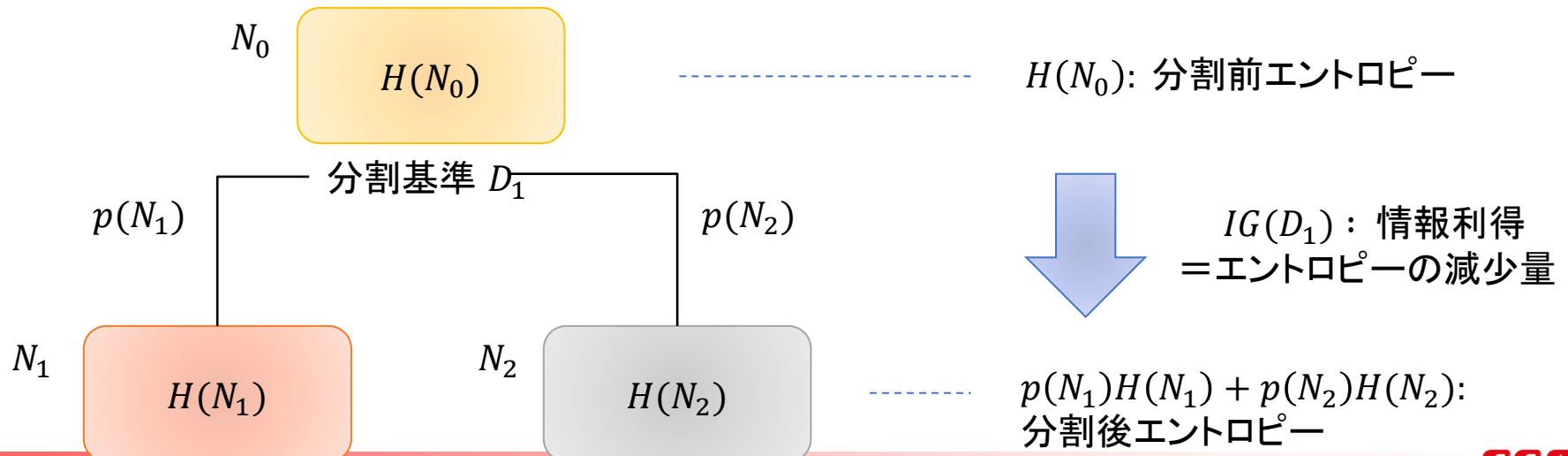
$$IG(D_1) = H(N_0) - \{ p(N_1)H(N_1) + p(N_2)H(N_2) \}$$

- H : エントロピー

分割前エントロピー

分割後エントロピー

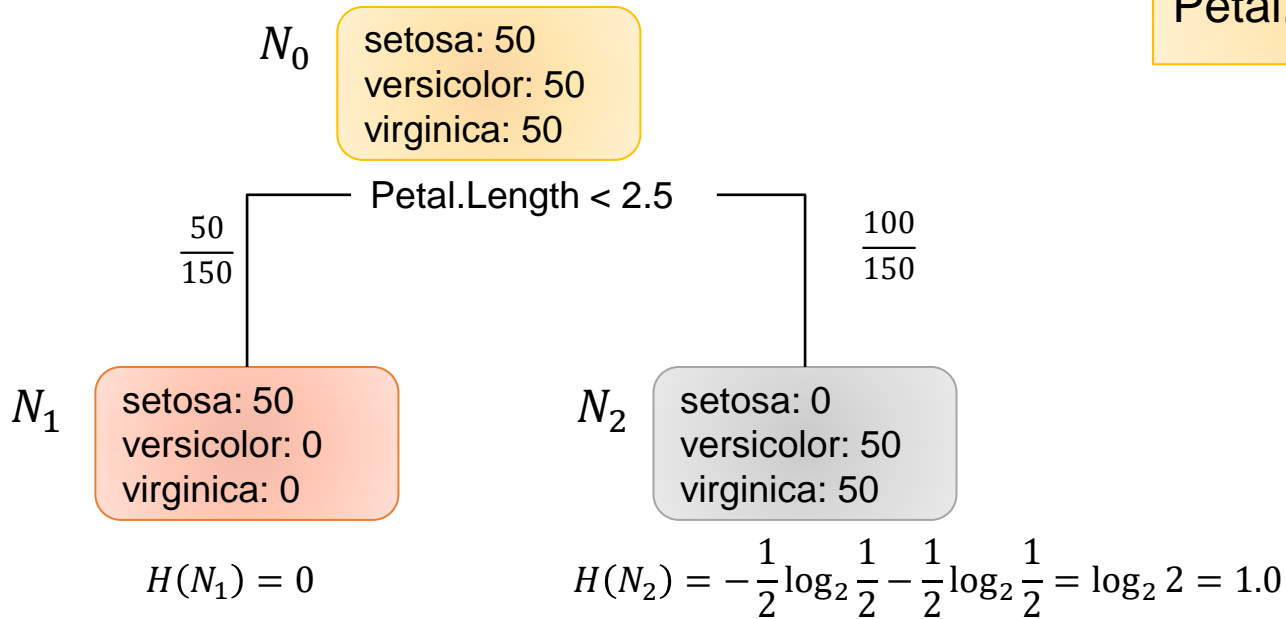
- p : 分割比率 ($p(N_1) + p(N_2) = 1$)



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



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

情報利得

分割前エントロピー

分割後エントロピー

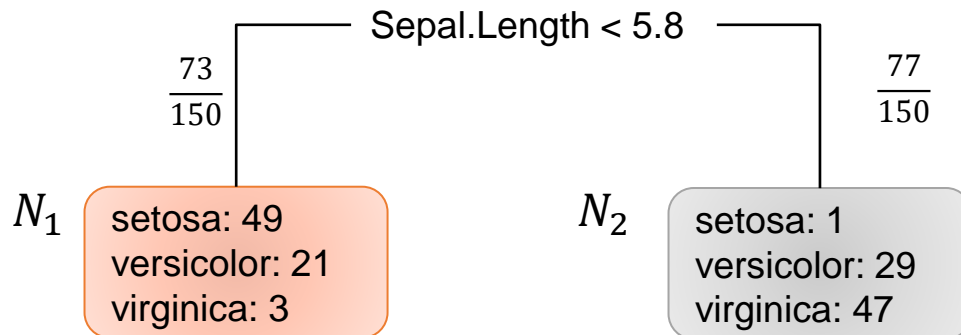
ある分割D2による情報利得の計算例

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

 N_0

setosa: 50
versicolor: 50
virginica: 50

分割D2:
Sepal.Length < 5.8



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

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

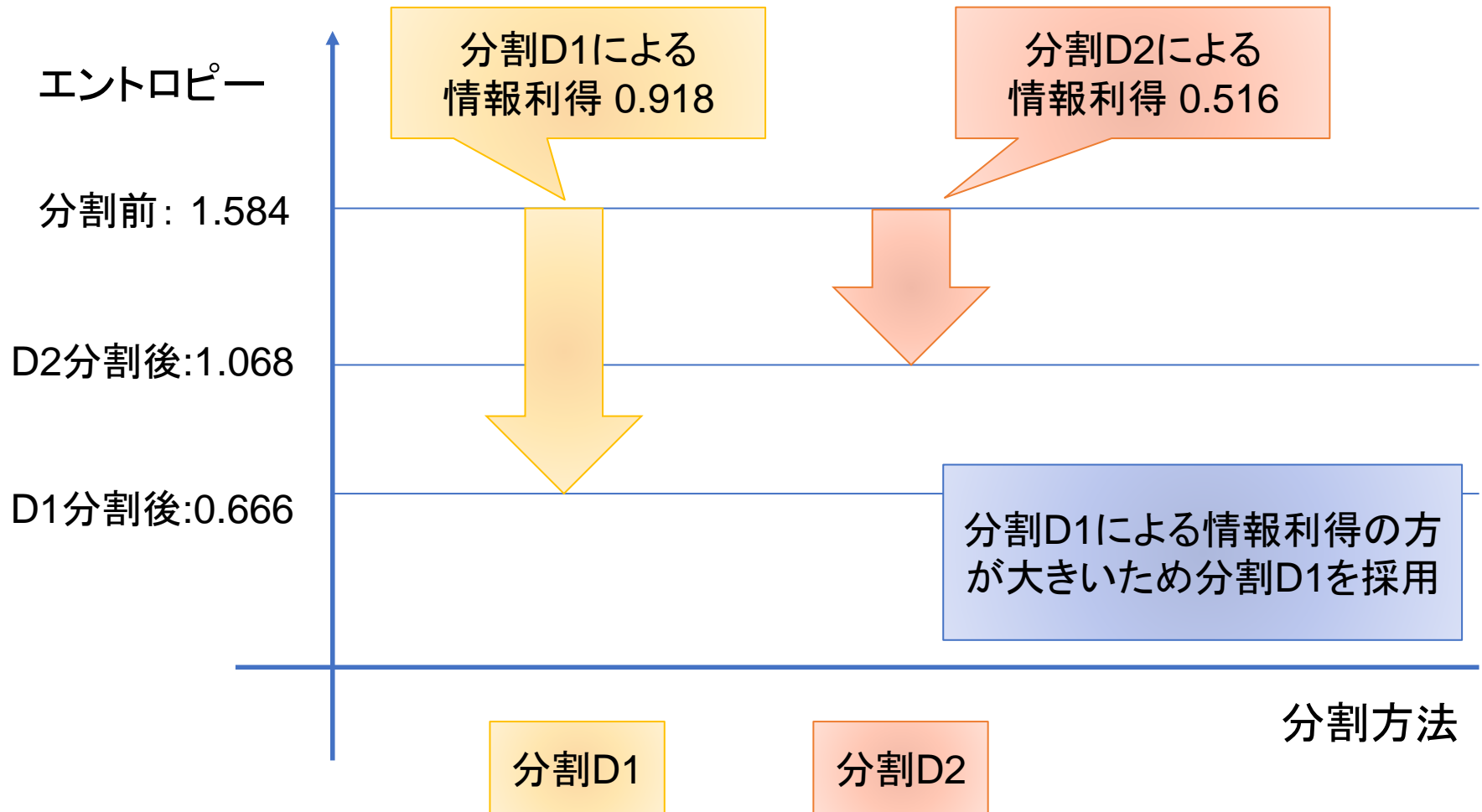
$$IG(D_2) = H(N_0) - \left\{ \frac{73}{150} * H(N_1) + \frac{77}{150} * H(N_2) \right\} = 1.584 - 1.068 = 0.516\text{bit}$$

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

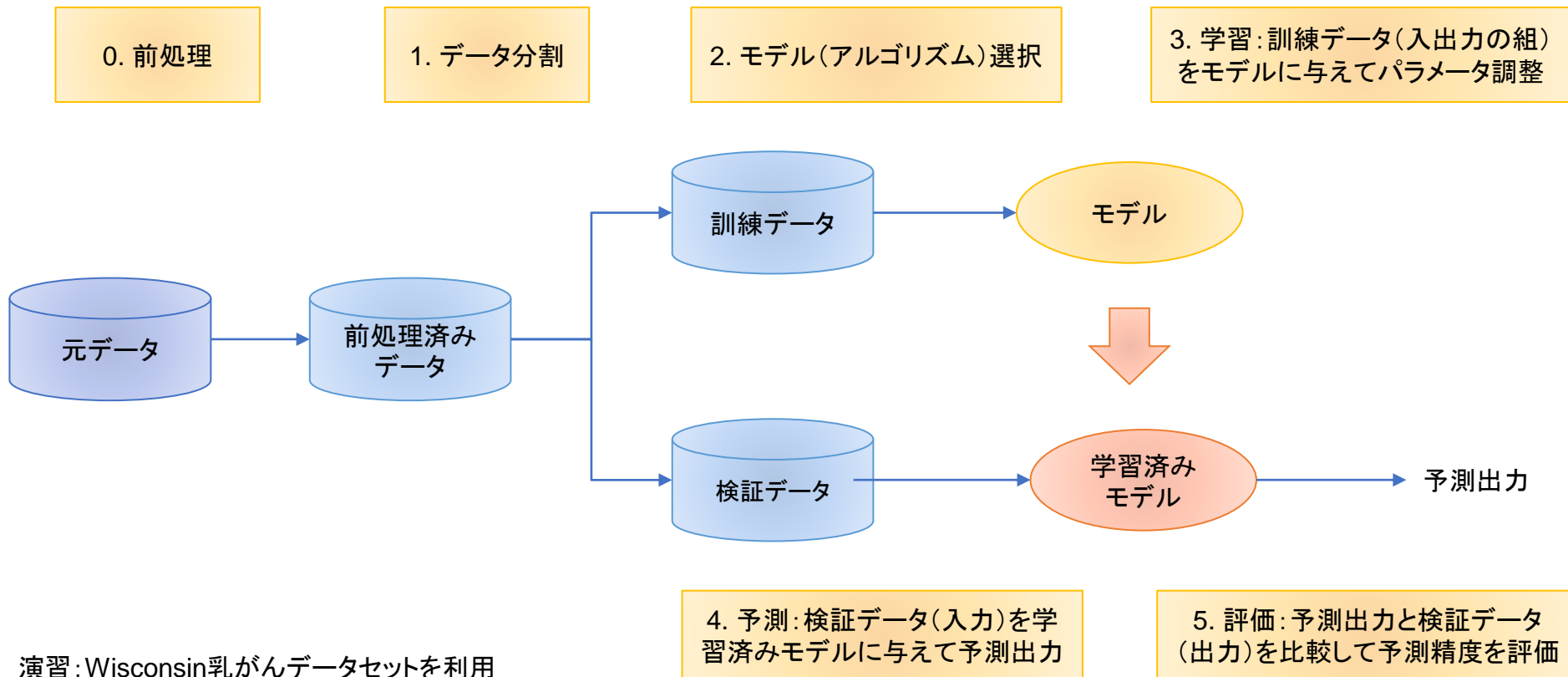
- 予測したい事例について、個々の決定木が予測した各クラスへの所属確率を平均し、確率が高いと判断されるクラスを予測結果とする

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



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合で必要

R Studioのインストールと起動

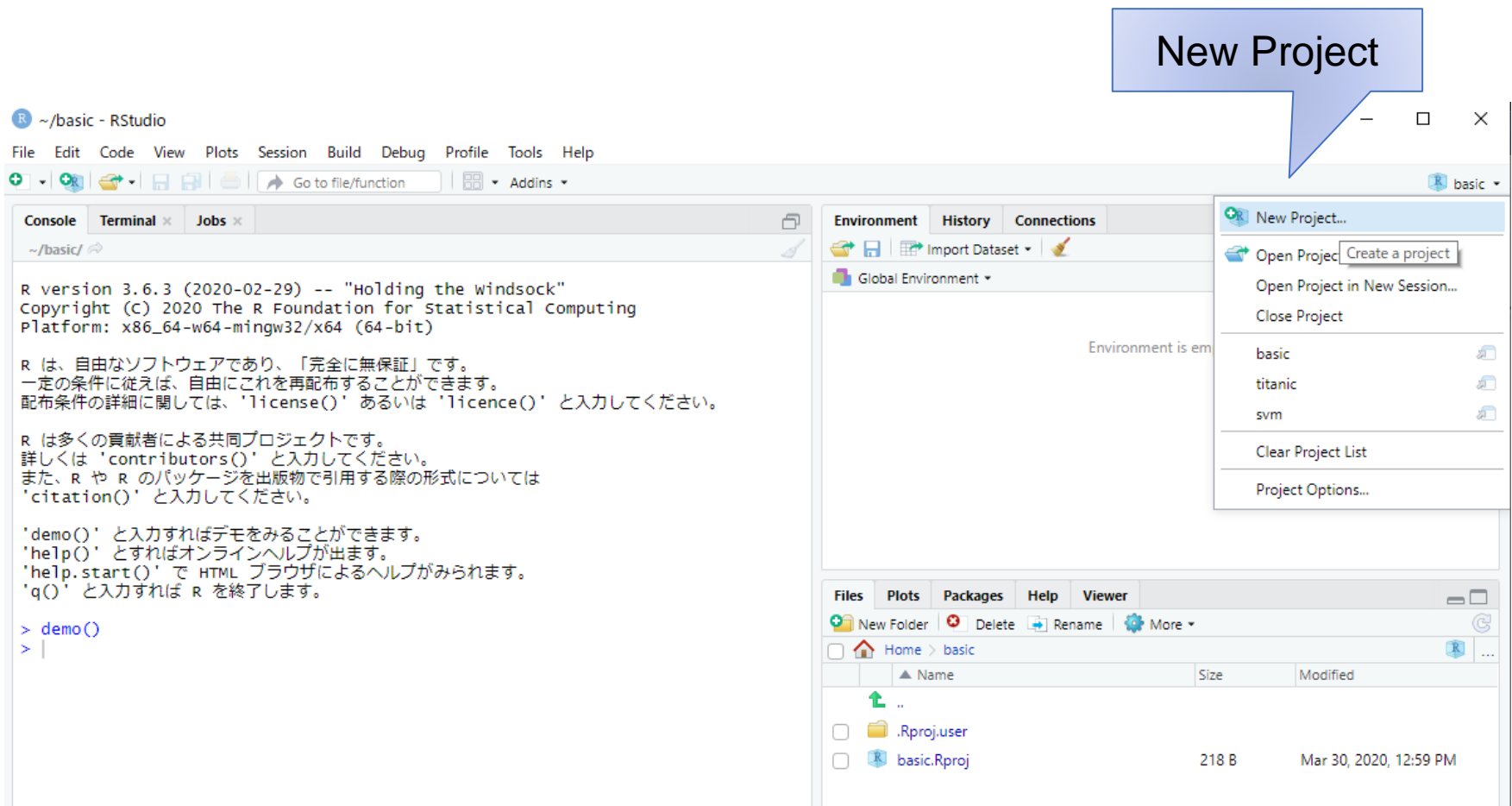
<https://rstudio.com/>にアクセスし、DOWNLOADからRStudio Desktop (Free)を選択

- 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

The image illustrates the steps to create a new project in RStudio:

- New Project** dialog box: Shows options to create a project in a new directory, an existing directory, or using version control. The **New Directory** option is selected.
- New Project** dialog box: Shows the **Project Type** selection screen. The **New Project** option is selected.
- Create New Project** dialog box: Shows the **Directory name** field with the text **first** entered. A callout bubble indicates that any directory name can be used.

Additional options visible in the third dialog box include:

- ☐ Open in new session
- Create Project** button
- Cancel** button

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

ソースペインにコマンド入力

The screenshot shows the RStudio interface. The Source pane (top left) contains a script with the command `str(iris)`. The Environment pane (top right) shows the Global Environment. The Console pane (bottom left) shows the output of the command, which is the structure of the `iris` data frame. A blue callout points to the Source pane, and another blue callout points to the Console pane. An orange callout points to the Console pane output.

- `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

データ読み込み・データ構造確認

~/first - RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

```
1 str(iris)
2 d <- iris
3
```

d <- iris でオブジェクトdにiris
データ読み込み

オブジェクト名の左にある矢印ボタン
でstr()同様データ構造が確認できる

Environment

Data

d 150 obs. of 5 variables

Sepal.Length:	num	5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
Sepal.Width :	num	3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
Petal.Length:	num	1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
Petal.Width :	num	0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
Species :	Factor w/ 3 levels	"setosa","versicolor",...: 1 1 ...

Files Plots Packages Help Viewer

New Folder Delete Rename More

Home > first

	Name	Size	Modified
	..		
	.Rproj.user		
	first.Rproj	218 B	Mar 30, 2020, 1:09 PM

3:1 (Top Level) R Script

Console Terminal Jobs

詳しくは `contributor()` と入力してください。
また、R や R のパッケージを出版物で引用する際の形式については
'`citation()`' と入力してください。

'`demo()`' と入力すればデモをみることができます。
'`help()`' とすればオンラインヘルプが出ます。
'`help.start()`' で HTML ブラウザによるヘルプがみられます。
'`q()`' と入力すれば R を終了します。

```
> str(iris)
'data.frame': 150 obs. of 5 variables:
 $ Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 $ Sepal.Width : num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ Petal.Length: num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 $ Petal.Width : num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 $ Species : Factor w/ 3 levels "setosa","versicolor",...: 1 1 1 1 1 1 1 1 1 1 ...
1 1 1 ...
> d <- iris
```

データフレーム型データの表示

~/first - RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Environment History Connections

Global Environment

Data

d 150 obs. of 5 variables

d (data.frame)

Files Plots Packages

New Folder Delete

Home > first

Name Size Modified

..

.Rhistory 64 B Apr 9, 2020, 6:26 PM

.Rproj.user 218 B Apr 9, 2020, 6:26 PM

first.Rproj

Console Terminal Jobs

~/first/

'demo()' と入力すればデモをみることができます。
'help()' とすればオンラインヘルプが出ます。
'help.start()' で HTML ブラウザによるヘルプがみられます。
'q()' と入力すれば R を終了します。

```
> str(iris)
'data.frame': 150 obs. of 5 variables:
 $ Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 $ Sepal.width : num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ Petal.Length: num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 $ Petal.width : num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 $ Species : Factor w/ 3 levels "setosa","versicolor",...: 1 1 1 1 1 1 1 1 1 1 ...
> d <- iris
> view(d)
> view(d)
> |
```

データフレーム型データの場合行の適当な場所をクリックするとデータ全体をソースペインに表示できる

データフレーム:
表形式のデータ構造

データ構造:ベクトル

The screenshot shows the RStudio interface. The script editor contains the following code:

```
1 str(iris)
2 d <- iris
3 x <- 1:5
4 y <- c(1.2, 2.5, 3.2, 4.9, 5.0)
5 x
6 y
7 z <- c("Benign", "Malignant")
8 z
9 |
```

A callout box highlights the following code snippets:

```
x <- 1:5
y <- c(1.2, 2.5, 3.2, 4.9, 5.0)
z <- c("Benign", "Malignant")
x
y
z
```

The Environment pane shows the data frame 'd' with 150 observations and 5 variables. The values are:

Variable	Class	Values
x	int [1:5]	1 2 3 4 5
y	num [1:5]	1.2 2.5 3.2 4.9 5
z	chr [1:2]	"Benign" "Malignant"

The Console shows the execution of the code and the resulting data frame:

```
> d <- iris
> view(d)
> y <- c(1.2, 2.5, 3.2, 4.9, 5.0)
> x <- 1:5
> x
[1] 1 2 3 4 5
> y
[1] 1.2 2.5 3.2 4.9 5.0
> z <- c("Benign", "Malignant")
> z
[1] "Benign" "Malignant"
>
```

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

The screenshot shows the RStudio interface with the following code in the editor:

```

1 str(iris)
2 d <- iris
3 x <- 1:5
4 y <- c(1.2, 2.5, 3.2, 4.9, 5.0)
5 x
6 y
7 z <- c("Benign", "Malignant")
8 z
9 A <- matrix(1:10, nrow = 5, ncol = 2)
10 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),
11             nrow = 4)
12 A
13 B
14

```

The Environment pane on the right shows the following objects:

Object	Class	Dimensions	Values
A	int	[1:5, 1:2]	1 2 3 4 5 6 7 8 9 10
B	num	[1:4, 1:3]	1.2 6.1 3.2 5.2 2.4 2.5 4.9 9.9...
d			150 obs. of 5 variables
x	int	[1:5]	1 2 3 4 5
y	num	[1:5]	1.2 2.5 3.2 4.9 5
z	chr	[1:2]	"Benign" "Malignant"

Callout 1 (pointing to line 9):

```

A <- matrix(1:10, nrow = 5, ncol = 2)

```

Callout 2 (pointing to line 10):

```

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)

```

Callout 3 (pointing to lines 12-13):

```

A
B

```

Callout 4 (pointing to the console output):

```

> 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
  [,1] [,2]
[1,]  1   6
[2,]  2   7
[3,]  3   8
[4,]  4   9
[5,]  5  10
> B
  [,1] [,2] [,3]
[1,] 1.2 2.4 1.3
[2,] 6.1 2.5 4.0
[3,] 3.2 4.9 5.0
[4,] 5.2 9.9 0.5
>

```

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

```

~/first - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function Addins
Untitled1* x d x
Source on Save Run Source
1 str(iris)
2 d <- iris
3 x <- 1:5
4 y <- c(1.2, 2.5, 3.2, 4.9, 5.0)
5 x
6 y
7 z <- c("Benign", "Malignant")
8 z
9 A <- matrix(1:10, nrow = 5, ncol = 2)
10 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),
11          nrow = 4)
12 A
13 B
14 l <- list(m = matrix(1:10, nrow = 2), v=1:100, df = iris)
15 l
16 l$m
17 l$v
18 l$df
19 str(l)
20

20:1 (Top Level)
Console Terminal Jobs
~/first/
146      6.7      3.0      5.2      2.3
147      6.3      2.5      5.0      1.9
148      6.5      3.0      5.2      2.0
149      6.2      3.4      5.4      2.3
150      5.9      3.0      5.1      1.8
> str(l)
List of 3
 $ m : int [1:2, 1:5] 1 2 3 4 5 6 7 8 9 10
 $ v : int [1:100] 1 2 3 4 5 6 7 8 9 10 ...
 $ df:'data.frame': 150 obs. of 5 variables:
 ..$ Sepal.Length: num [1:150] 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 ..$ Sepal.Width : num [1:150] 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 ..$ Petal.Length: num [1:150] 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 ..$ Petal.Width : num [1:150] 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 ..$ Species      : Factor w/ 3 levels "setosa","versicolor",...: 1 1 1 1 1 1 1
 1 1 1 ...
>

```

オブジェクト名の左にある
矢印ボタンでもデータ構造確認

```

l <- list(m = matrix(1:10, nrow = 2), v=1:100, df = iris)
l
l$m
l$v
l$df
str(l)

```

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

```

~/first - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
+ - Go to file/function Addins
Untitled1* x d x
Z <- c("Benign", "Malignant")
Z
A <- matrix(1:10, nrow = 5, ncol = 2)
10 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),
11          nrow = 4)
12 A
13 B
14 l <- list(m = matrix(1:10, nrow = 2), v=1:100, df = iris)
15 l
16 l$m
17 l$v
18 l$df
19 str(l)
20 sex <- c("F", "F", "M", "M", "M")
21 height <- c(158, 162, 177, 173, 166)
22 weight <- c(51, 55, 72, 57, 64)
23 medical <- data.frame(SEX = sex, HEIGHT = height, WEIGHT = weight)
24 str(medical)
25 medical$HEIGHT
26 medical[,2]
27 medical[4,]
28 summary(medical)
29
29:1 (Top Level)
Console Terminal Jobs
~/first/
$ WEIGHT: num 51 55 72 57 64
> medical$HEIGHT
[1] 158 162 177 173 166
> medical[,2]
[1] 158 162 177 173 166
> medical[4,]
  SEX HEIGHT WEIGHT
4   M    173     57
> summary(medical)
  SEX    HEIGHT    WEIGHT
F:2   Min.   :158.0   Min.   :51.0
M:3   1st Qu.:162.0   1st Qu.:55.0
      Median :166.0   Median :57.0
      Mean   :167.2   Mean   :59.8
      3rd Qu.:173.0   3rd Qu.:64.0
      Max.   :177.0   Max.   :72.0
>

```

```

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

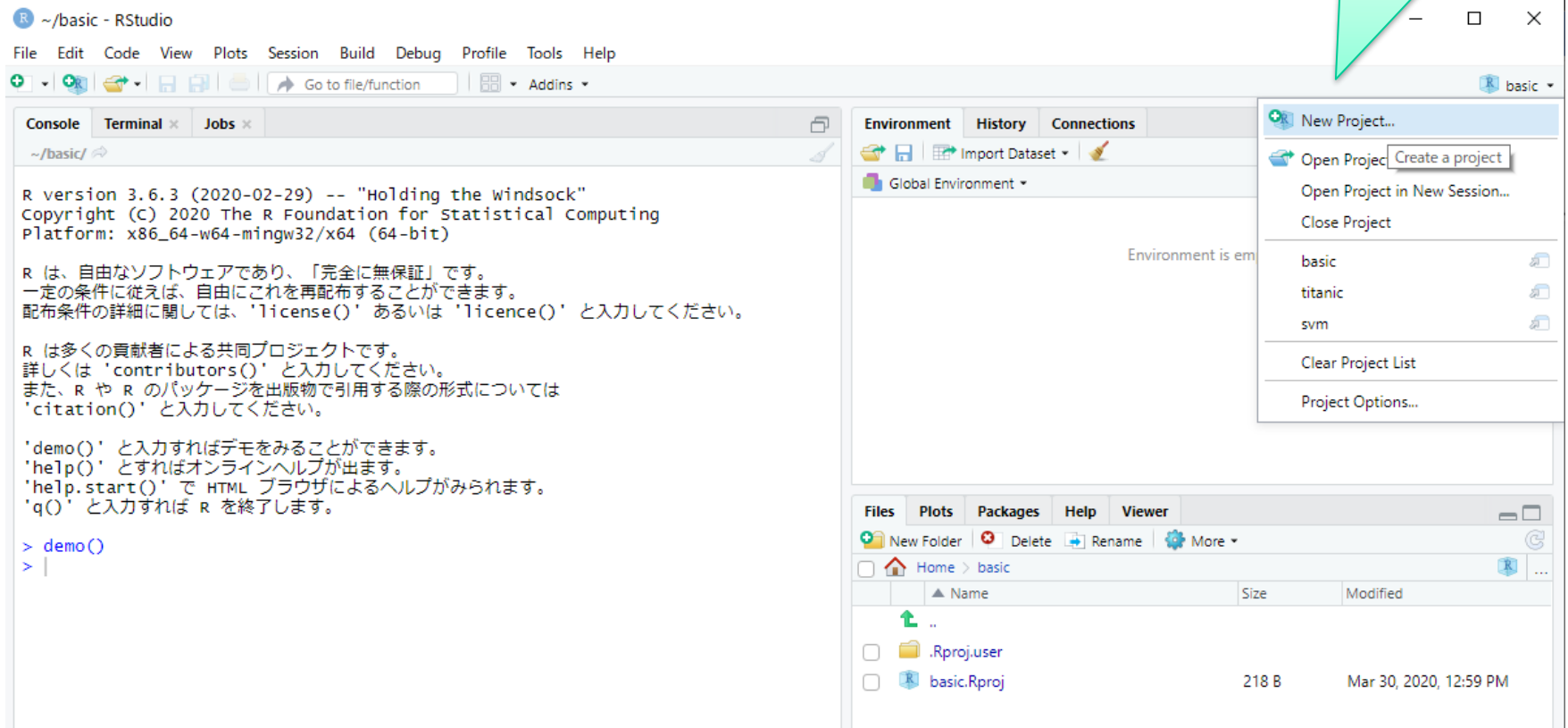
– Wisconsin Breast Cancer Diagnosticの意味

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

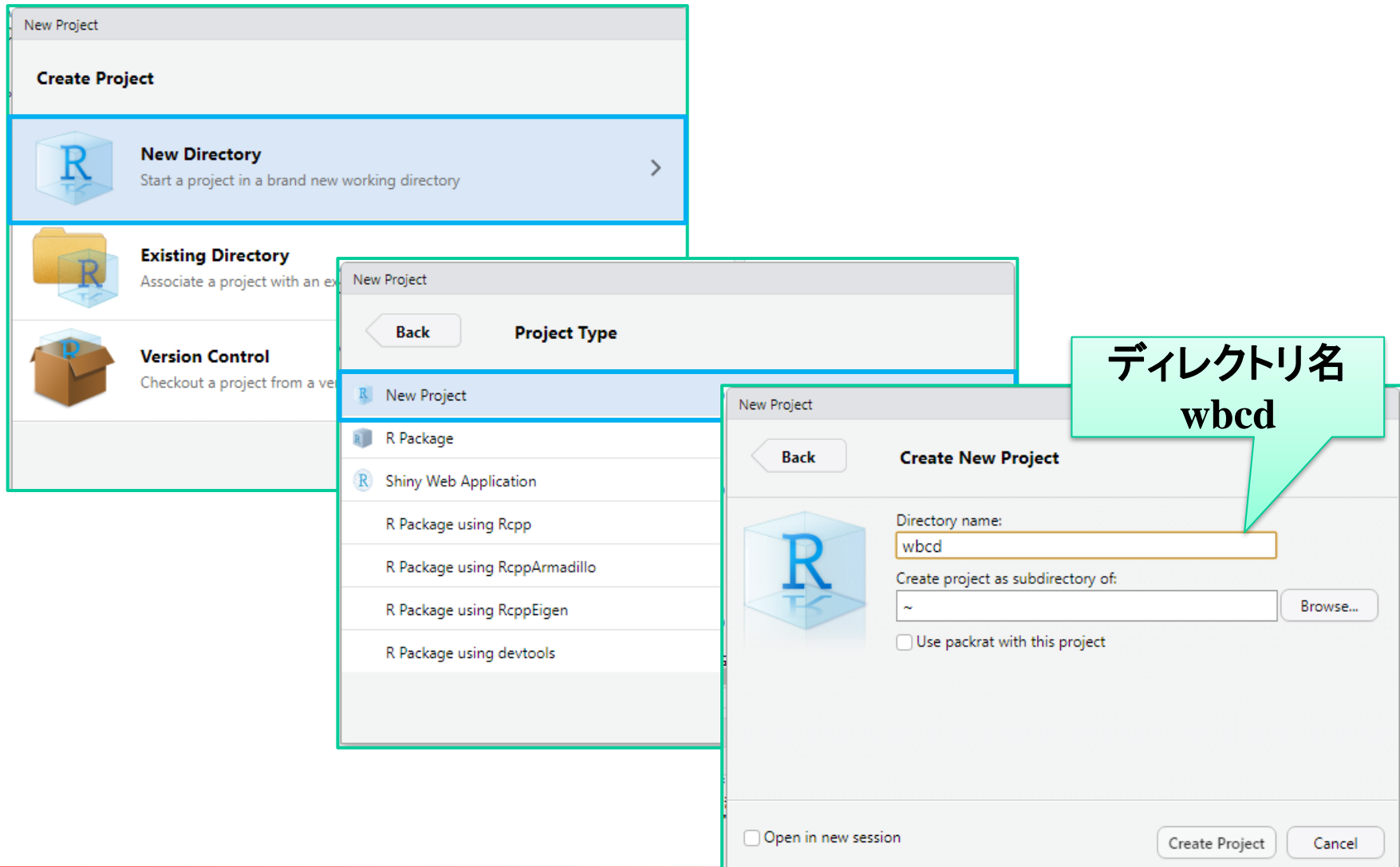
「新規プロジェクト作成」「スクリプト作成」の手順は覚えていると思うが、
忘れた人は次のページを参照のこと

R Studio起動後、右上からNew Project

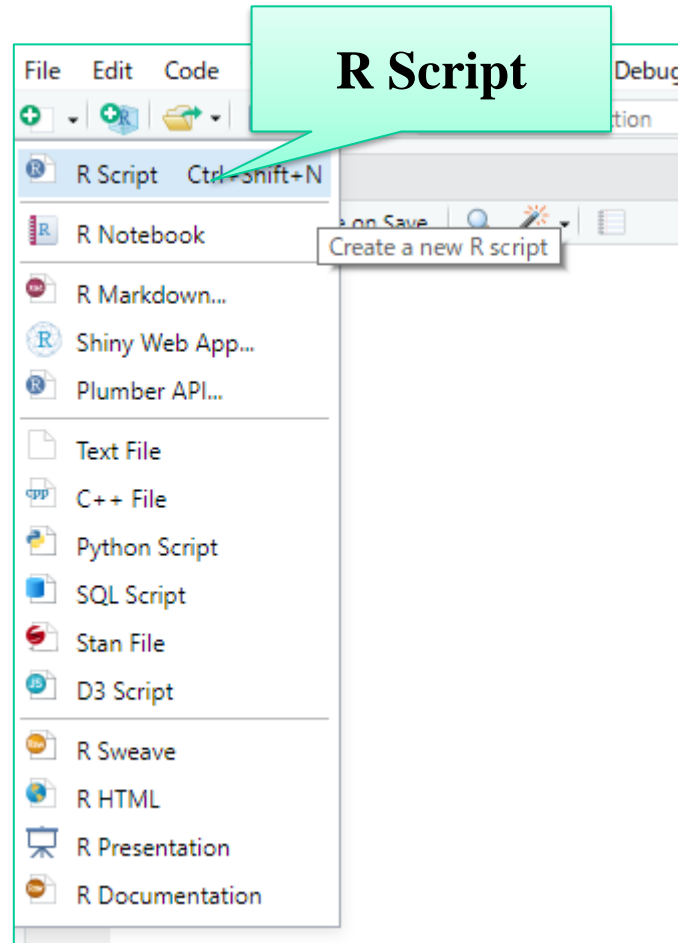
New Project



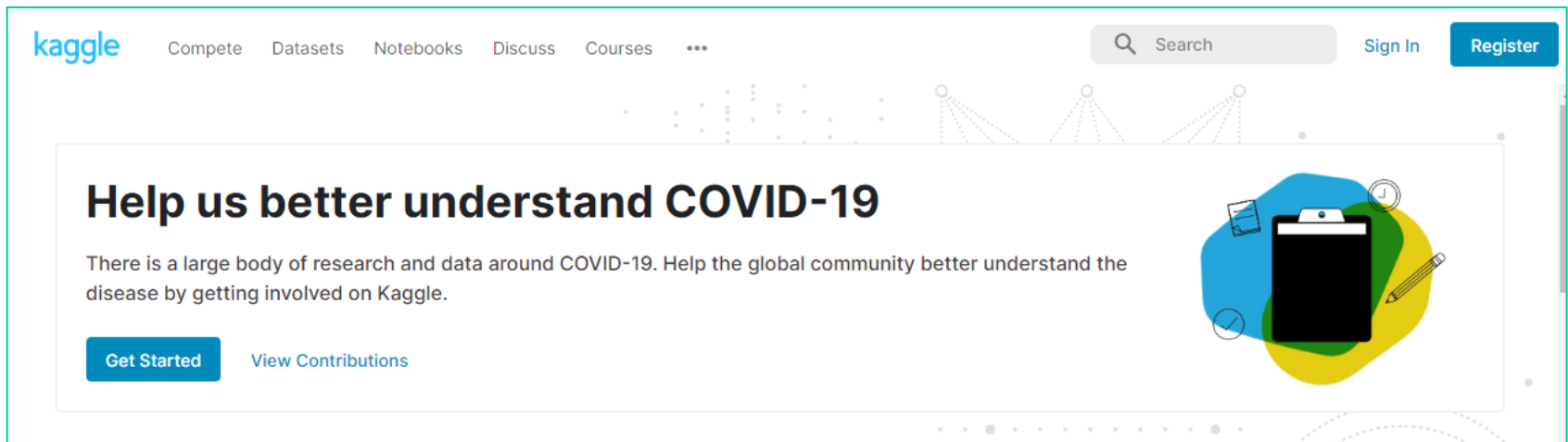
New Directory \Rightarrow New Project \Rightarrow ディレクトリ名 wbcd



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

The image shows the Kaggle website's registration page. At the top, the Kaggle logo is on the left, and navigation links for 'Compete', 'Datasets', 'Notebooks', 'Discuss', and 'Courses' are in the center. On the right, there is a search bar and 'Sign In' and 'Register' buttons. A green speech bubble points to the 'Register' button with the word 'Register'. Below the navigation bar, a banner for 'Help us better understand COVID-19' is visible. The main registration area has two tabs: 'Sign In' and 'Register'. Under the 'Register' tab, there are two options: 'Register with Google' (with a Google 'G' icon) and 'Register with your email' (with an email icon). A green speech bubble points to these two options with the text 'どちらでも好きな方法で登録' (Register with whichever method you prefer). Below these options, there is a link 'Have an account? Sign in.' and a paragraph of text about linking accounts.

kaggle Compete Datasets Notebooks Discuss Courses ...

Search Sign In Register

Help us better understand COVID-19

There is a large body of research and data on COVID-19 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

The screenshot shows the Kaggle dataset page for 'Breast Cancer Wisconsin (Diagnostic) Data Set'. The page header includes the 'Dataset' label, a yellow medal icon, and a view count of 1348. The main title is 'Breast Cancer Wisconsin (Diagnostic) Data Set' with the subtitle 'Predict whether the cancer is benign or malignant'. It is sourced from 'UCI Machine Learning' and was updated 4 years ago (Version 2). A green callout box on the right contains the text 'Wisconsin乳がん診断データセット'. Below the header, there are tabs for 'Data', 'Tasks (2)', 'Kernels (1,060)', 'Discussion (25)', 'Activity', and 'Metadata'. A 'Download (122 KB)' button is highlighted with a red rectangle. Below the tabs, the 'Usability' is 8.5, the 'License' is CC BY-NC-SA 4.0, and the 'Tags' are mathematics, cancer, and healthcare. The 'Description' section is visible at the bottom.

- <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

Console Terminal x Jobs x

~/wbcd/

R version 3.6.3 (2020-02-29) -- "Holding the Windsock"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R は、自由なソフトウェアであり、「完全に無保証」です。
一定の条件に従えば、自由にこれを再配布することができます。
配布条件の詳細に関しては、'license()' あるいは 'licence()' と入力してください。

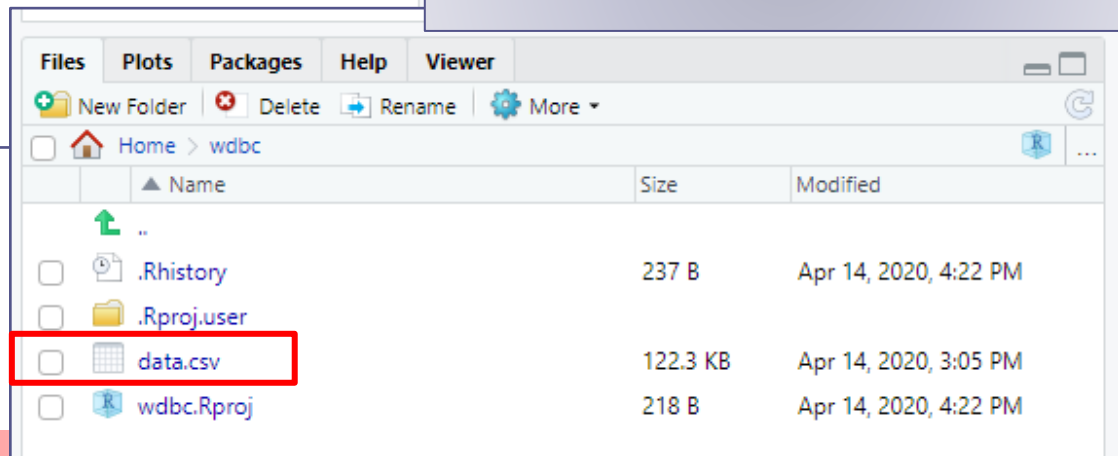
R は多くの貢献者による共同プロジェクトです。
詳しくは 'contributors()' と入力してください。
また、R や R のパッケージを出版物で引用する際の形式については
'citation()' と入力してください。

'demo()' と入力すればデモをみることができます。
'help()' とすればオンラインヘルプが出ます。
'help.start()' で HTML ブラウザによるヘルプがみられます。
'q()' と入力すれば R を終了します。

```
> getwd()
[1] "C:/Users/d_suz/OneDrive/ドキュメント/wbcd"
>
```

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



データの読み込みとデータ構造確認

The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains the R script:


```
1 wbcd <- read.csv("data.csv", stringsAsFactors = FALSE)
2 str(wbcd)
3
```
- Environment Pane:** Shows the variable `wbcd` with 569 observations and 33 variables.
- Files Pane:** Shows the directory structure, including `data.csv` (122.3 KB).
- Console:** Displays the output of the `str(wbcd)` command, showing the data types for each variable.


```
$ compactness_se : num 0.049 0.0131 0.0401 0.0746 0.0246 ...
$ concavity_se : num 0.0537 0.0186 0.0383 0.0566 0.0569 ...
$ concave.points_se : num 0.0159 0.0134 0.0206 0.0187 0.0188 ...
$ symmetry_se : num 0.03 0.0139 0.0225 0.0596 0.0176 ...
$ fractal_dimension_se : num 0.00619 0.00353 0.00457 0.00921 0.00511 ...
$ radius_worst : num 25.4 25 23.6 14.9 22.5 ...
$ texture_worst : num 17.3 23.4 25.5 26.5 16.7 ...
$ perimeter_worst : num 184.6 158.8 152.5 98.9 152.2 ...
$ area_worst : num 2019 1956 1709 568 1575 ...
$ smoothness_worst : num 0.162 0.124 0.144 0.21 0.137 ...
$ compactness_worst : num 0.666 0.187 0.424 0.866 0.205 ...
$ concavity_worst : num 0.712 0.242 0.45 0.687 0.4 ...
$ concave.points_worst : num 0.265 0.186 0.243 0.258 0.163 ...
$ symmetry_worst : num 0.46 0.275 0.361 0.664 0.236 ...
$ fractal_dimension_worst : num 0.1189 0.089 0.0876 0.173 0.0768 ...
$ X : logi NA NA NA NA NA NA ...
```

A callout box highlights the R code used to load the data:

```
wbcd <- read.csv("data.csv", stringAsFactors = FALSE)
str(wbcd)
```

データの意味

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

- 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 ($\text{perimeter}^2 / \text{area} - 1.0$)
 - 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

1. 変数の種類が33? おかしい.
クリックして確認

2. 意味のない変数Xが入っている

ss_worst	concavity_worst	concave.points_worst	symmetry_worst	fractal_dimension_worst	X
	0.711900	0.26540	0.4601	0.11890	NA
	0.241600	0.18600	0.2750	0.08902	NA
	0.450400	0.24300	0.3613	0.08758	NA
	0.686900	0.25750	0.6638	0.17300	NA
	0.400000	0.16250	0.2364	0.07678	NA
	0.535500	0.17410	0.3985	0.12440	NA
	0.378400	0.19320	0.3063	0.08368	NA
	0.267800	0.15560	0.3196	0.11510	NA
	0.539000	0.20600	0.4378	0.10720	NA
	1.105000	0.22100	0.4366	0.20750	NA
	0.145900	0.09975	0.2948	0.08452	NA

Showing 1 to 14 of 569 entries, 33 total columns

Environment History Connections
Data
wbcd 569 obs. of 33 variables

Files Plots Packages Help Viewer
New Folder Delete Rename More
Home > wbcd
Name Size Modified
.. 0 B Apr 14, 2020, 2:48 PM
.Rhistory
.Rproj.user

不要な変数xの削除

The screenshot shows the RStudio interface with the following components:

- Source Editor:** Contains the following R code:


```
1 wdbc <- read.csv("data.csv", stringsAsFactors = FALSE)
2 str(wdbc)
3 wdbc$x <- NULL
4 str(wdbc)
5
```
- Environment Pane:** Shows the variable 'wdbc' with 569 observations and 32 variables.
- Console:** Shows the output of the commands:


```
> wdbc$x <- NULL
> str(wdbc)
'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" "M" "M" "M"
 $ radius_mean : num  18 20.6 19.7 11.4 2...
 $ 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 ...
 $ texture_se : num  0.905 0.734 0.787 1.156 0.781 ...
```

Two callout boxes provide additional information:

- A box pointing to line 3 of the code contains the text: `wdbc$x <- NULL` and `str(wdbc)`.
- A box pointing to the console output contains the text: 変数の種類が32となり正しい (The number of variable types is 32, which is correct).

スクリプトの保存・ワークスペースの保存

The screenshot illustrates the process of saving a script and the workspace in RStudio. The 'File' menu is open, and 'Save As...' is highlighted. A callout points to this option, stating 'FileからSave As...'. The 'Save File - Untitled1' dialog box is shown, with the file name 'wbcd_script.R' entered. A callout explains that the file extension should be '.R', stating 'スクリプトファイル名の拡張子は .R とする'. The 'Quit R Session' dialog box is also visible, asking to save the workspace image to '~/.RData?'. A callout points to this dialog, stating 'R 終了時'.

FileからSave As...

Save File - Untitled1

スクリプトファイル名の拡張子は .R とする

R 終了時

Quit R Session

Save workspace image to ~/.RData?

Save Don't Save Cancel

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)

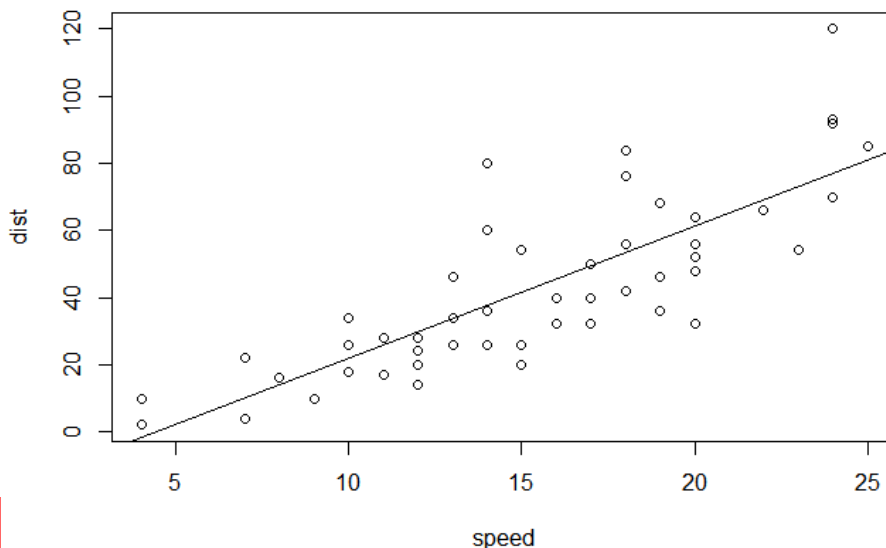
線形回帰

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, \dots, x_d として $y = w_0 + w_1x_1 + w_2x_2 + \dots + w_dx_d$ と表現される. w_1, w_2, \dots, 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 $y = 3.9x - 17.6$
- 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

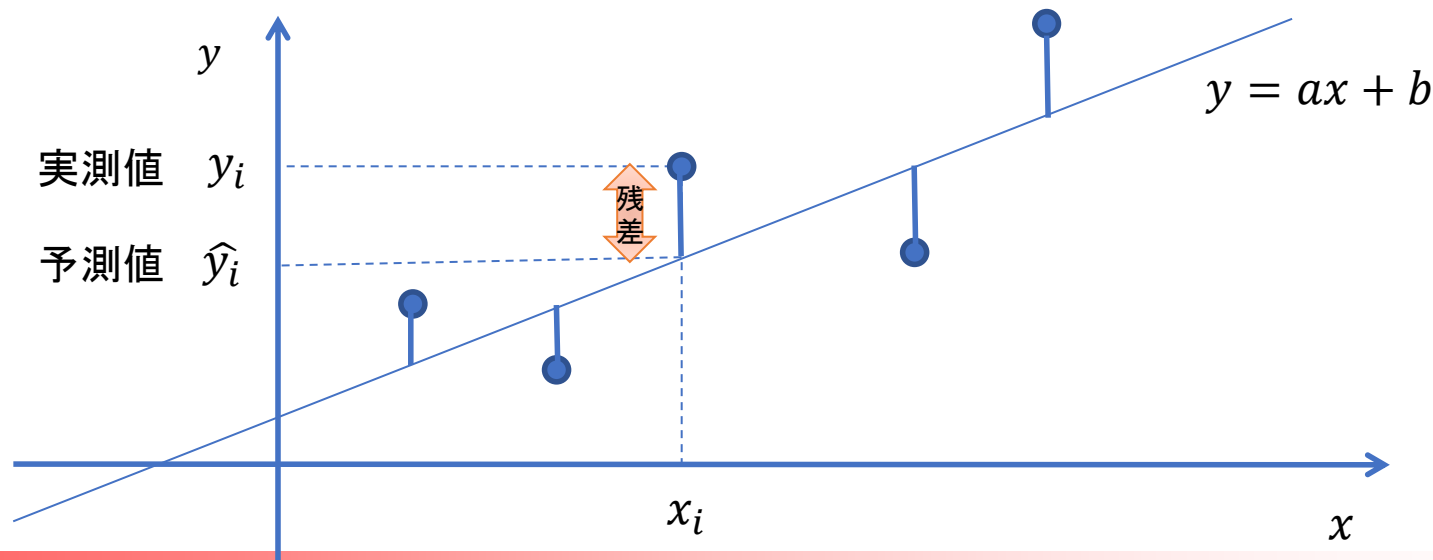
最小二乗法による回帰係数の導出(1/2)

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 $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ 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

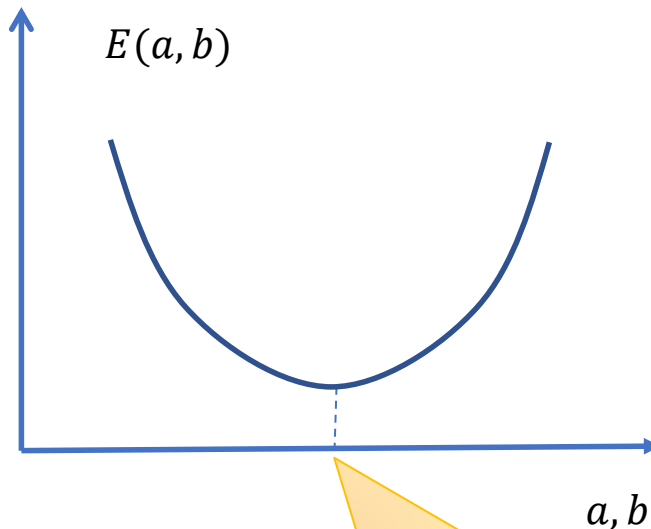
$$E = \sum_{i=1}^n (\hat{y}_i - y_i)^2 = \sum_{i=1}^n (ax_i + b - y_i)^2$$



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



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

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

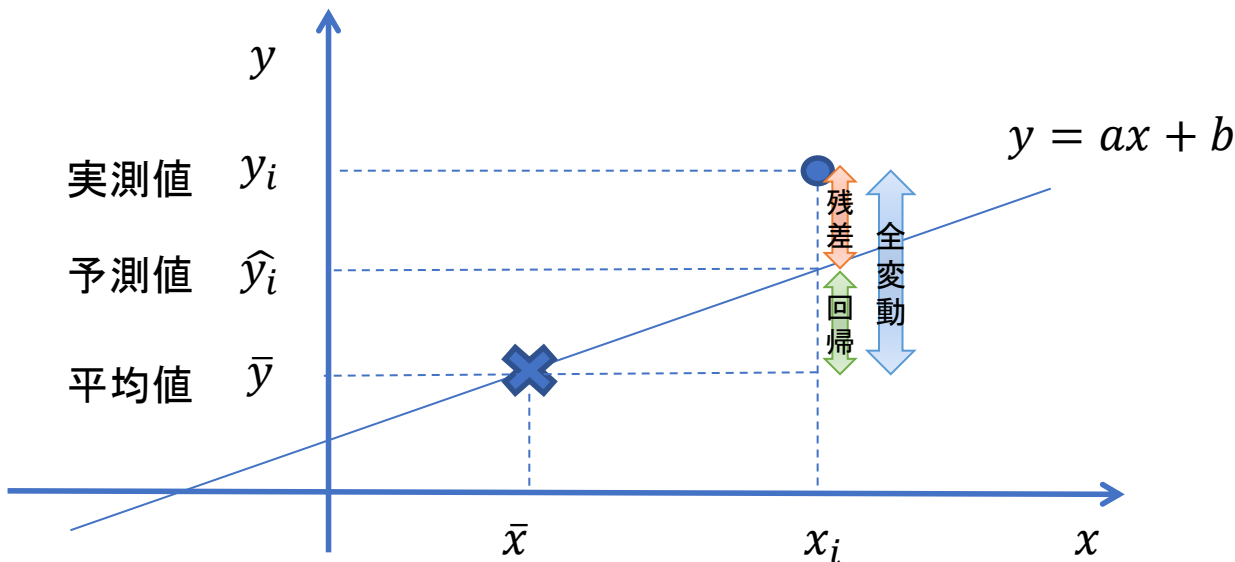
$$a = \frac{\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\frac{1}{n} \sum_{i=1}^n x_i^2 - \bar{x}^2} = \frac{\sigma_{xy}}{\sigma_x^2},$$

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

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

$$y = w_0 + w_1x_1 + w_2x_2 + \cdots + w_dx_d$$

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

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

重回帰式は

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

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

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

データ行列

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

をもとに

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

を定義すると、各サンプル行 (1~n) に対して回帰式をあてはめた結果は

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

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

$$\begin{aligned} E(w) &= |y - \tilde{X}w|^2 = (y - \tilde{X}w)^T (y - \tilde{X}w) \\ &= y^T y - w^T \tilde{X}^T y - y^T \tilde{X}w + w^T \tilde{X}^T \tilde{X}w \end{aligned}$$

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

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

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パッケージに含まれるlm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 $\lambda|\mathbf{w}|^2$ (L2ノルム)を付加

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

- 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 $\lambda|\mathbf{w}|_1$ (L1ノルム) を付加

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

- ここで $|\mathbf{w}|_1 = \sum_{i=1}^d |w_i|$
- 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 $\lambda|\mathbf{w}|^2$ (L2ノルム)を付加

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

- 係数があまりに大きくならないようにしている

Lasso回帰

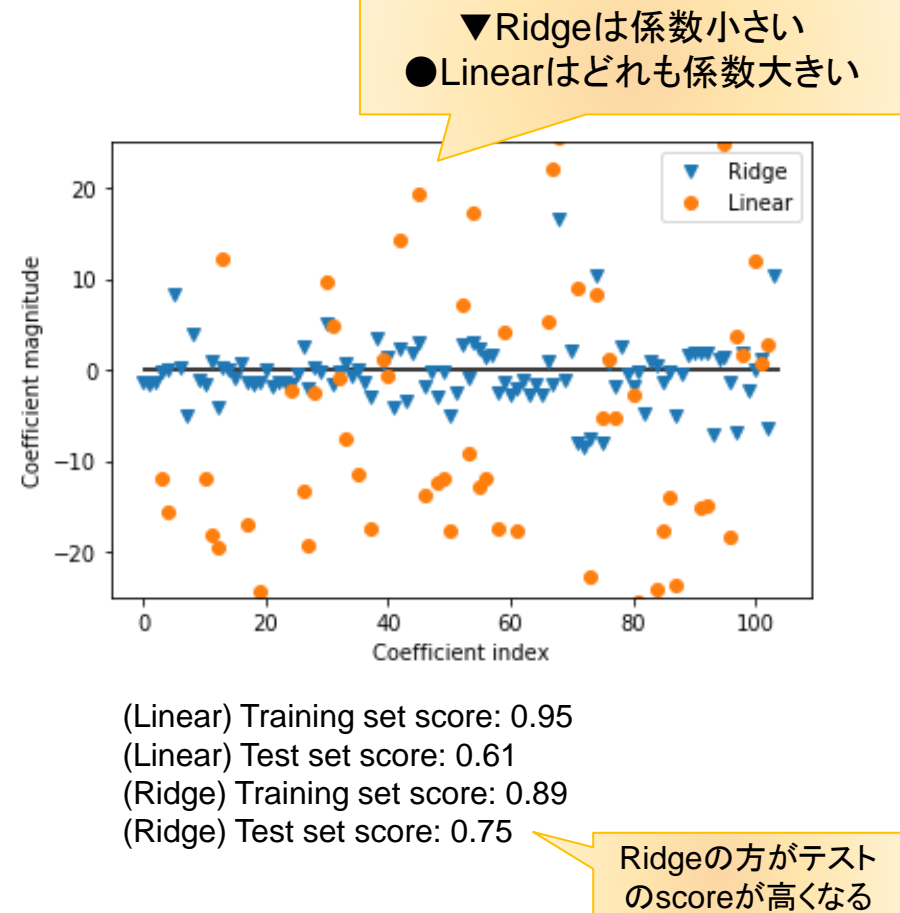
- 最小にすべき目的関数にペナルティ $\lambda|\mathbf{w}|_1$ (L1ノルム) を付加

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

- ここで $|\mathbf{w}|_1 = \sum_{i=1}^d |w_i|$
- 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

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

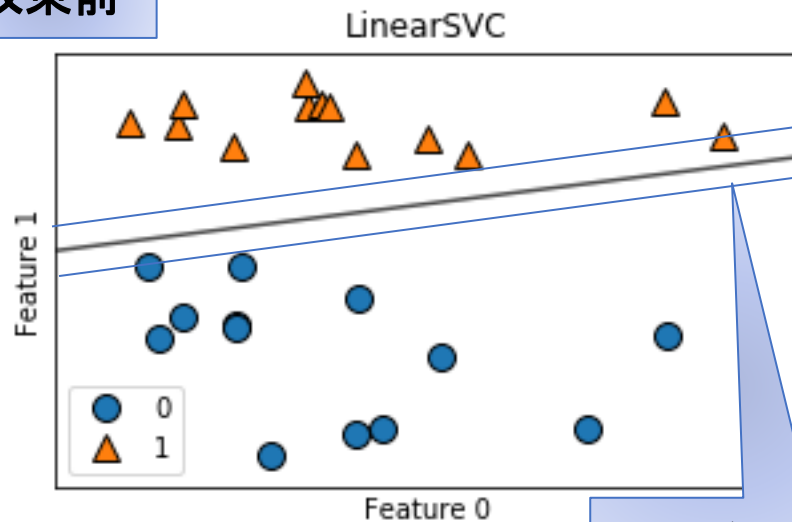
- ハードマージンSVM
 - 各クラスの手データが分類境界からなるべく離れるように分類境界を定める
- ソフトマージンSVM
 - 線形分離できない場合に、スラック変数 ξ を導入して、分類境界で分離できないデータを許容する
- カーネルSVM
 - カーネル法によって入力データをより高次元の特徴空間に写像し、特徴空間上で線形分離することで、実空間での非線形分離を可能にする
- SV回帰
 - SVMを回帰問題へ応用したもの

SVMの実装

- Rのkernlabパッケージなどで実装されている

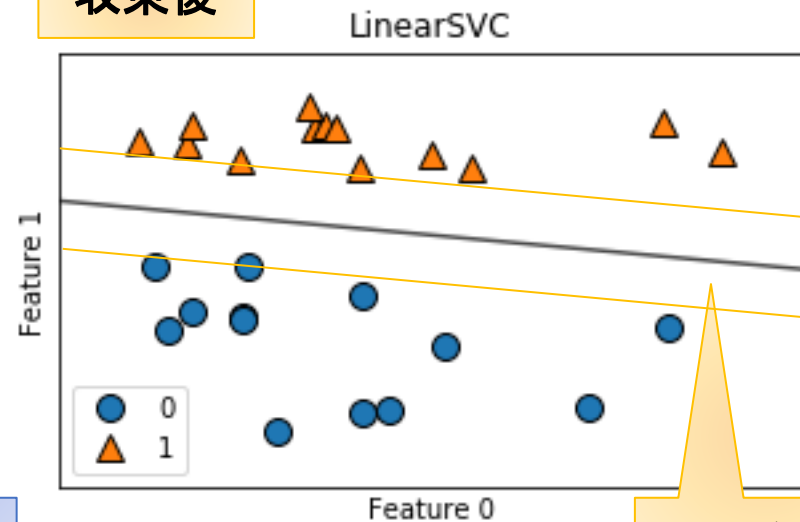
ハードマージンSVM

収束前



マージン小

収束後



マージン大

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

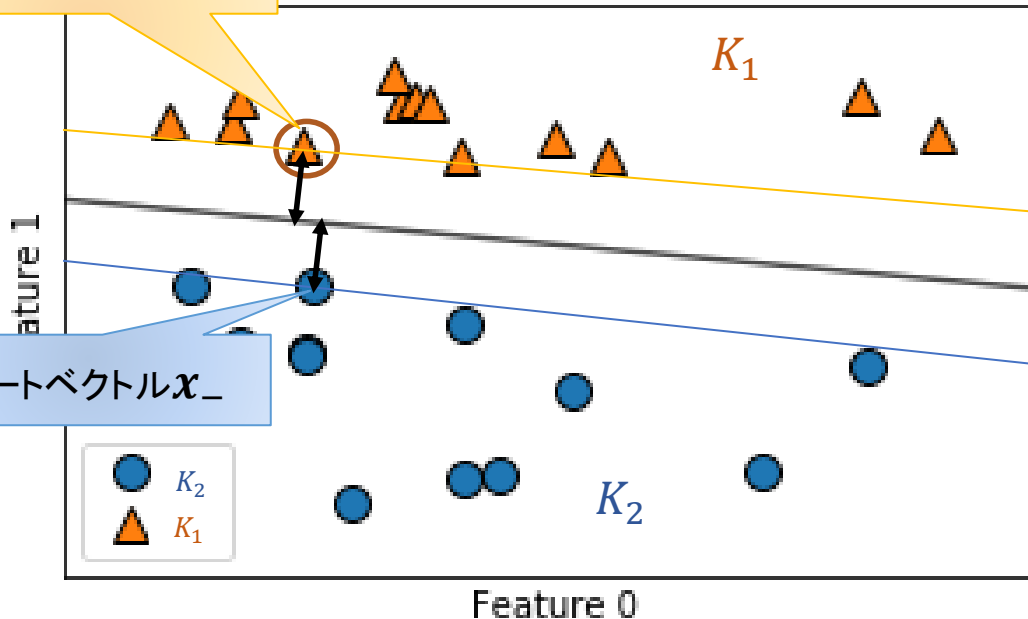
サポートベクトル: 分類境界に最も近いサンプル

マージン: サポートベクトルと分類境界の距離

マージン最大化による分類境界の導出(1/3)

サポートベクトル x_+

LinearSVC



サポートベクトル x_-

分類境界

$$\mathbf{w}^T \mathbf{x} + b = 0$$

$$\mathbf{w}^T \mathbf{x}_i + b > 0$$

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

$$\mathbf{w}^T \mathbf{x}_i + b < 0$$

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 $\frac{t_i(\mathbf{w}^T \mathbf{x}_i + b)}{|\mathbf{w}|}$ と表される. これより, マージン最大化は

$$\text{Maximize}_{\mathbf{w}, b} \min_i \frac{t_i(\mathbf{w}^T \mathbf{x}_i + b)}{|\mathbf{w}|}$$

マージン最大化による分類境界の導出(2/3)

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

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

$$\text{Maximize}_{\mathbf{w}, b} \frac{1}{|\mathbf{w}|} \quad \text{Subject to } t_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 \quad (i = 1, 2, \dots, n)$$

さらに簡単化して,

$$\text{Minimize}_{\mathbf{w}, b} \frac{1}{2} |\mathbf{w}|^2 \quad \text{Subject to } t_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 \quad (i = 1, 2, \dots, n)$$

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

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

L を b, \mathbf{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 \mathbf{a} だけの関数になる

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

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

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

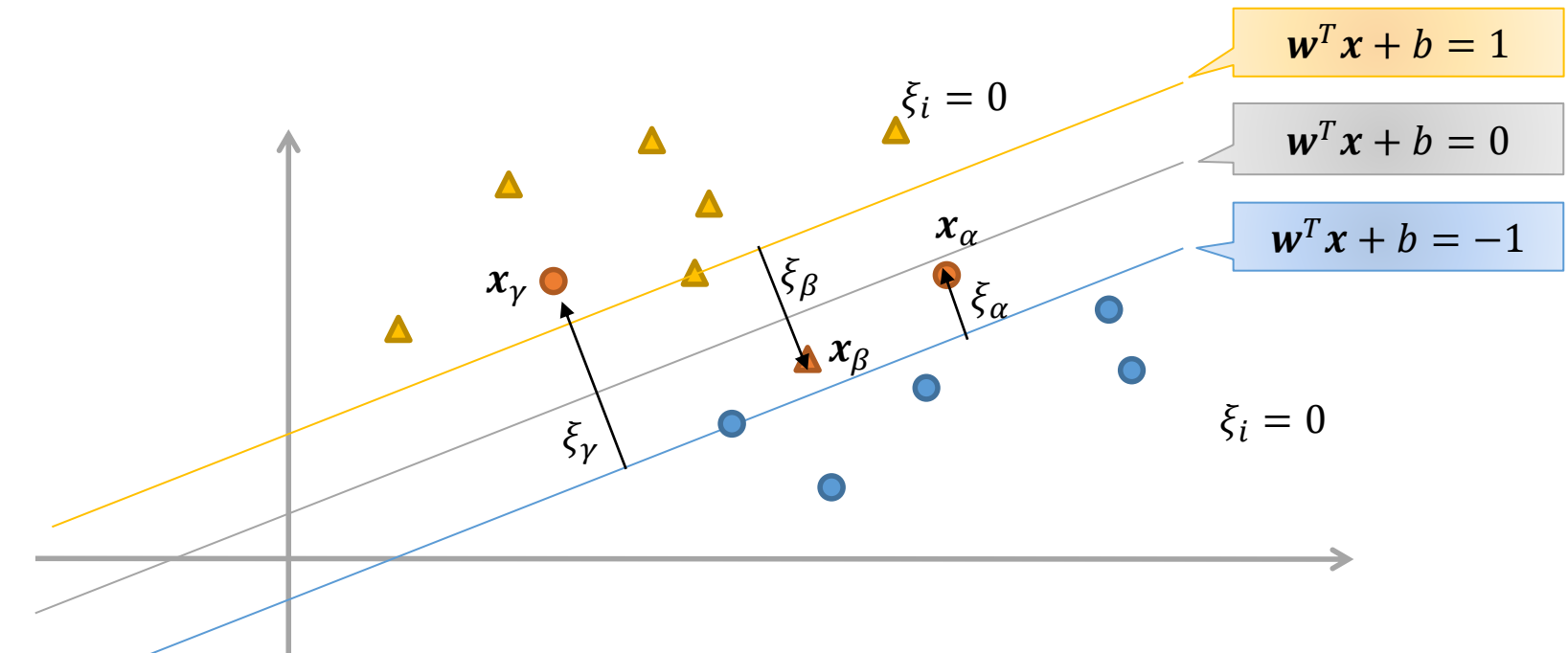
$$\text{Subject to } \sum_{i=1}^n a_i t_i = 0$$

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

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



$$\text{Minimize}_{w,b,\xi} \quad \frac{1}{2} |w|^2 + C \sum_{i=1}^n \xi_i \quad \text{Subject to } t_i(w^T x_i + b) \geq 1 - \xi_i \quad (i = 1, 2, \dots, n)$$

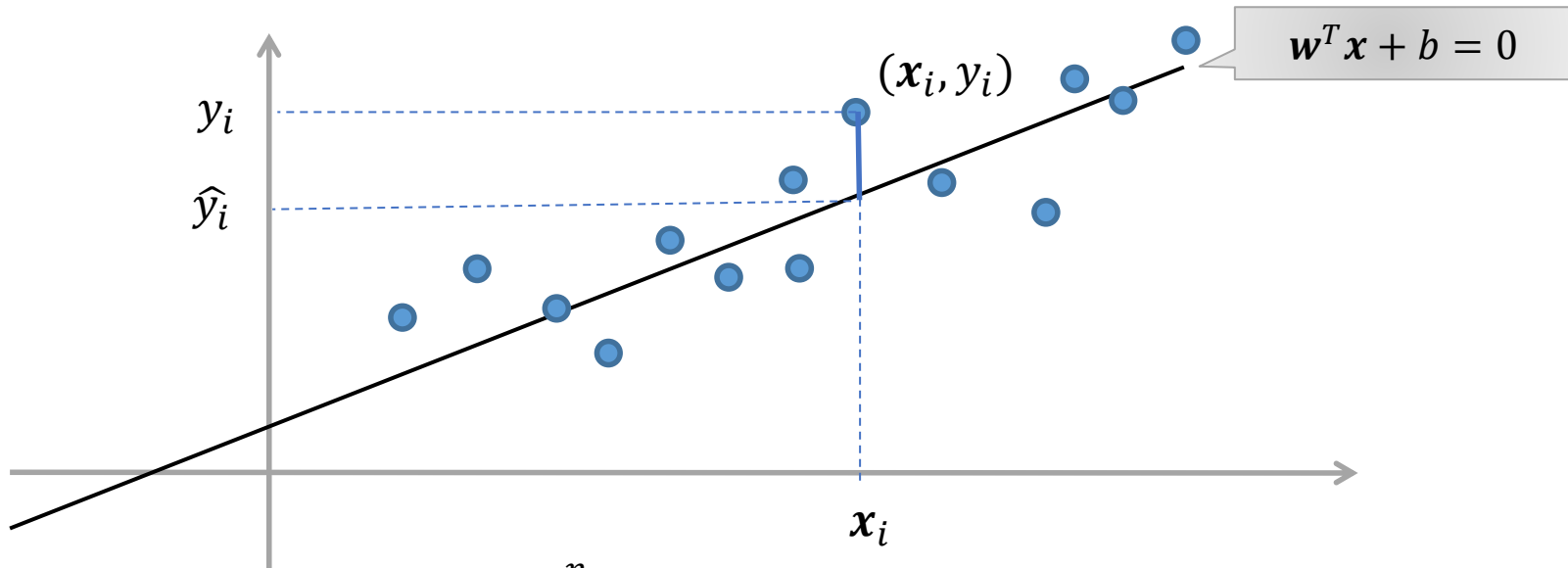
$$\xi_i \geq 0 \quad (i = 1, 2, \dots, n)$$

C: 小さい \Rightarrow 大きい ξ を許容

C: $\rightarrow \infty \Rightarrow \xi = 0$ のみ許容 (ハードマージン)

SV回帰

- SVMの回帰問題への応用



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

$\epsilon > 0$: 不感度係数

$C > 0$: 正則化係数

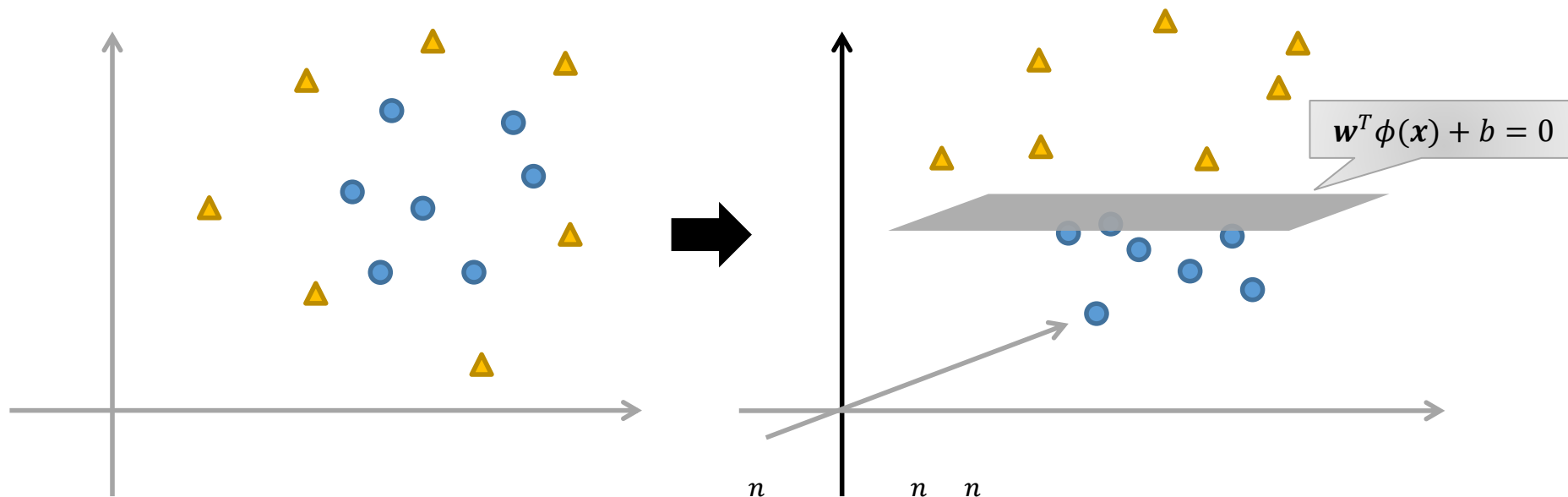
不感度: 残差が ϵ 以下の時は損失0

正則化: 小さい \Rightarrow 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



$$\text{Maximize } L(\mathbf{a}) = \sum_{i=1}^n a_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n a_i a_j t_i t_j \phi(x_i)^T \phi(x_j)$$

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

カーネル法

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

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

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 $\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ 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

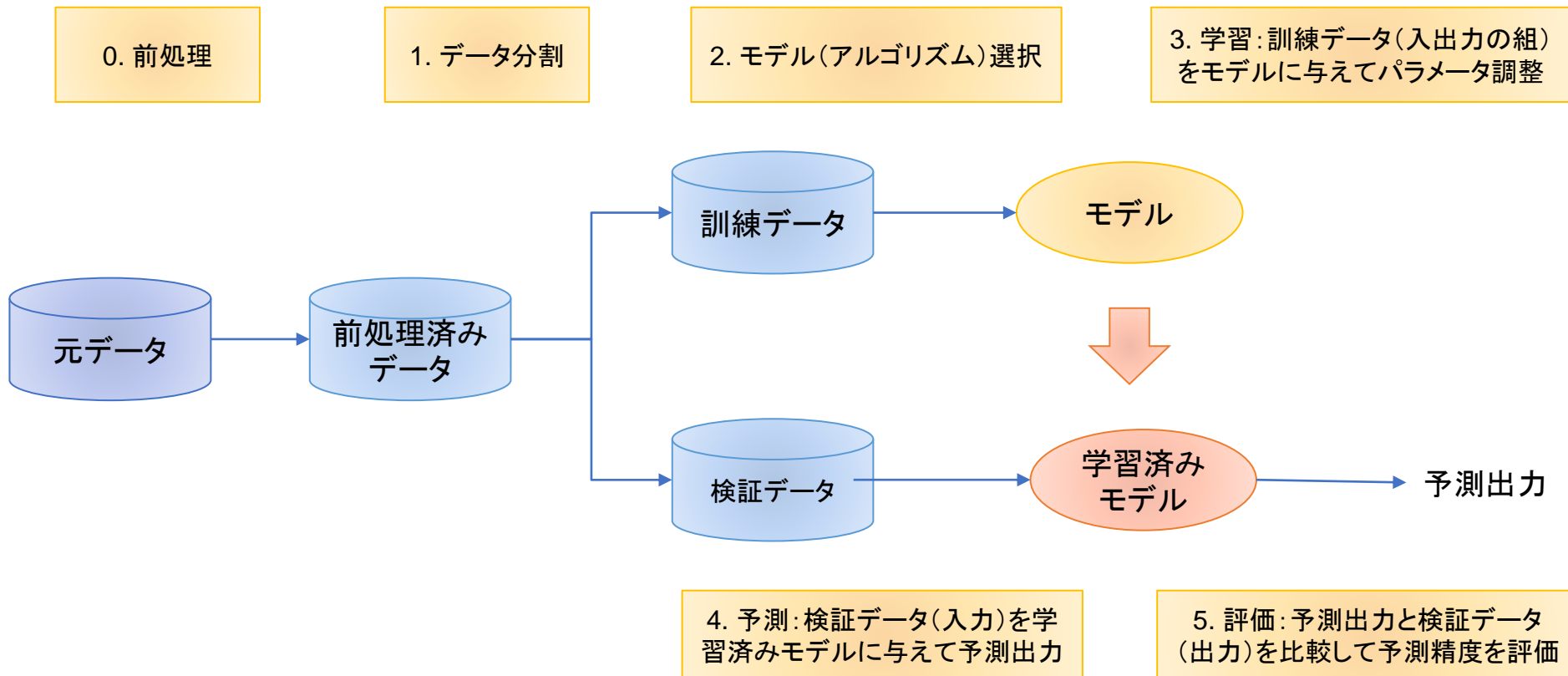
カーネル関数の一例

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

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

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