

# 化学屋さん向けAI基礎講座

徳島大学デザイン型AI教育研究センター

徳島大学工学部（併任）

鳥井 浩平

# 自己紹介

- 徳島大学で学士・修士・博士を取得
- そのまま徳島大学デザイン型AI教育研究センターに着任
- AIとコンピュータのなんでも屋
- 専門は画像処理（特に医用画像）とニューラルネットワーク
- やりたい研究はニューラルネットワークの基礎研究
- その他、サーバー管理やWEBアプリケーション開発など、なんでもできると言うとう仕事が増えるので最近は言っていない

# 講演資料の公開

- 本日の講演資料は下記ウェブサイトで公開しています



<https://trklibrary.com/archives/>

# 目次

1. AIとは
2. AI技術の変遷
3. AIによる予測とは
4. 化学分野のAI
5. 実演

# AIとは

# AI（人工知能）とは

- 人が作り出した、人のような知覚や知性をもつ”何か”
- 特に、人のような知覚や知性をコンピュータを用いて再現したもの
- AIの定義は曖昧であり、AIという言葉の解釈は人によって異なる

## 世間一般のAIという言葉の解釈

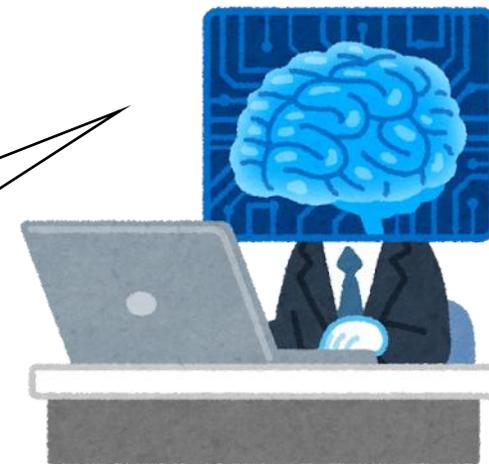
- もはやなんでも解決してくれる万能なシステム
- データを分類したり、データから数値を予測したりするシステム
- 機械学習、ディープラーニング（ニューラルネットワーク）

# 一般人目線のAI（映画やドラマの影響）



こんなデータが出たんだけど、  
このデータから何がわかるかな？  
あと、このデータから〇〇を  
予測するモデルを作ってもらえる？

わかりました。  
このデータにはXXといった法則があります。  
このデータを用いて〇〇を予測するモデルを  
生成しました。ご活用ください。



万能なAI

# 専門家目線のAI

ARIMA      Histogram of Gradients

Rule Base      Decision Tree      Local Binary Pattern      SIFT / SURF

Linear Regression      Random Forest

                         LASSO / Ridge Regression      XGBoost

Neural Network      k-means      Support Vector Machine

                         Auto Encoder      Principal Component Analysis

Transformer / Vision Transformer      Q Learning

Convolutional Neural Network      Diffusion Model      etc.

# AIに対する認識の摺り合わせ

- AIはなんでもできる万能システムではない
- AIは目的や環境によって適切な設計が必要な技術である
- AIは人と同じように考えたり学習したりするシステムではない
- AIは数学や計算機科学の延長線上にある技術である
- AIは結局のところ計算であることに注意する
- ただし、その振る舞いが少しずつ人に近づいているのは事実

# AI技術の変遷

# AIの変遷

## 第1世代

ルールベース…人が知識や法則を設計し、人らしい振る舞いを再現

## 第2世代

機械学習（統計モデル）

…回帰分析、ARIMA、決定木、k-means、主成分分析、etc.

## 第3世代

**ニューラルネットワーク（ディープラーニング）**

# ルールベース

## ルールベースのAIとは

- 人が持つ知見をもとに条件分岐等を駆使して構築したAI
- いわゆる網羅的なIF文のプログラミングによるAI

## メリット・デメリット

- 計算プロセスがAIの中で最も明瞭
  - 多様な環境の変化に対応できない、ノイズにととても弱い
- より柔軟な判断が可能なモデルが欲しい→機械学習が流行る

# 機械学習

## 機械学習とは

- データを用いた計算によってデータの性質や法則を見つける技術
- **学習** ≡ データを用いた計算

## 機械学習の真価

- コンピュータは人が一生かけても処理しきれないデータを処理できる  
→人には発見できない何かを機械学習で見つけられる可能性がある
- 物事の新しい性質や法則を発見できるかもしれない
  - 今までは困難であった予測問題を解けるかもしれない

# ニューラルネットワーク

- 生物の脳の神経網を模倣した数理モデル
- 機械学習の一種
- **ほかの機械学習手法で必要だった特徴抽出までも自動化可能**
- 今ではAIと言ったらほとんどこれ、一般的なAIのイメージもこれ
- ほかの機械学習手法とは一線を画す広範な応用が可能
- ただし、その複雑な計算過程はモデルを作った人ですら解釈不可能
  - 脳の中身を覗くようなもの、解釈しようとすること自体がナンセンスかも…
- データの解析にはとことん向かないが、予測精度はピカイチ
- 化学分野でも応用が進められているが...化学屋さんには受け入れ難い？

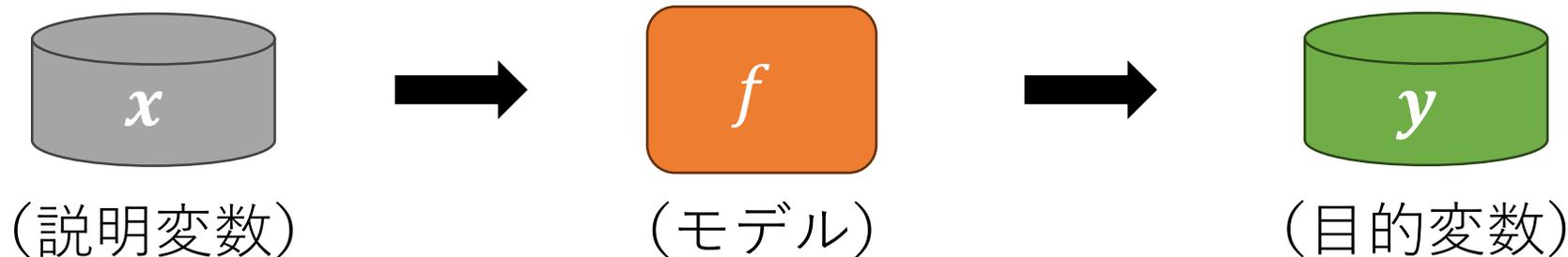
# AIによる予測とは

# AIによる予測の本質（回帰・分類）

## 学習フェーズ

データ $x$ とデータ $y$ を結ぶ関数 $f$ を計算によって求める

$$y = f(x)$$



連続値なら回帰  
カテゴリなら分類

## 予測

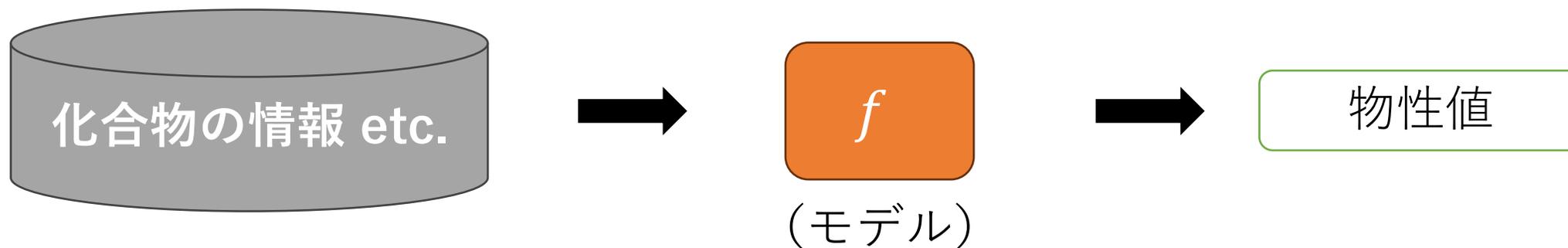
求まった関数 $f$ にデータ $x$ を入力してデータ $y$ を得ること

# 物性値予測のAI

## 学習フェーズ

化合物の情報から物性値を算出する関数 $f$ を計算によって求める

$$\text{物性値} = f(\text{化合物の情報})$$



## 予測

求まった関数 $f$ に化合物の情報を入力して物性値を得る

# 化学分野のAI

# 化学分野で使いやすい？機械学習手法

## 教師あり

- 線形回帰（単回帰、重回帰）
- Ridge / LASSO回帰
- PLS回帰（主成分分析 + 線形回帰）
- ロジスティック回帰
- 決定木

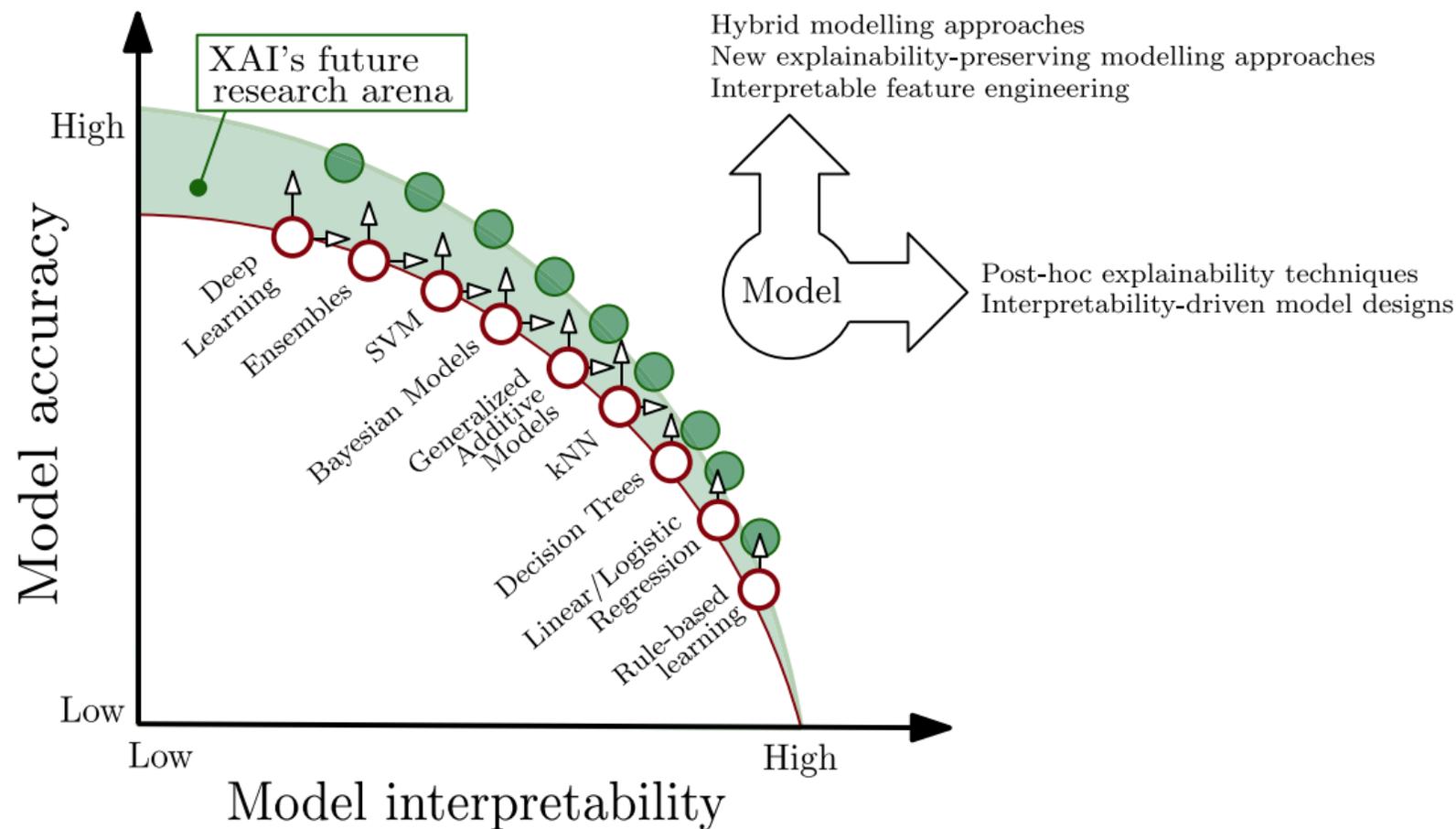
比較的簡単で中身が明瞭  
とりあえずこのあたりから  
手を付けてみると良いかも

## 教師なし

- 主成分分析
- k-means

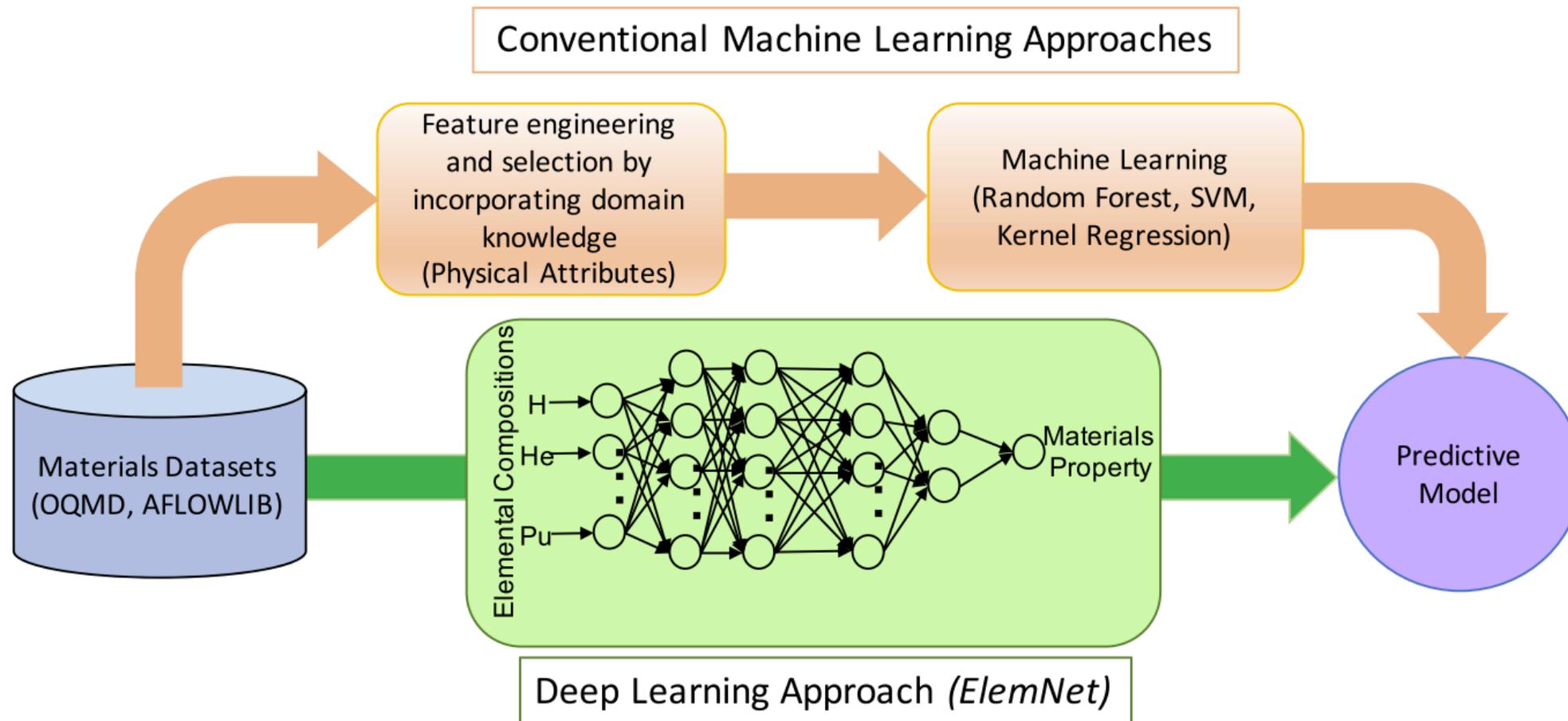
※後ほど実演します

# AIの解釈性と精度のトレードオフな関係



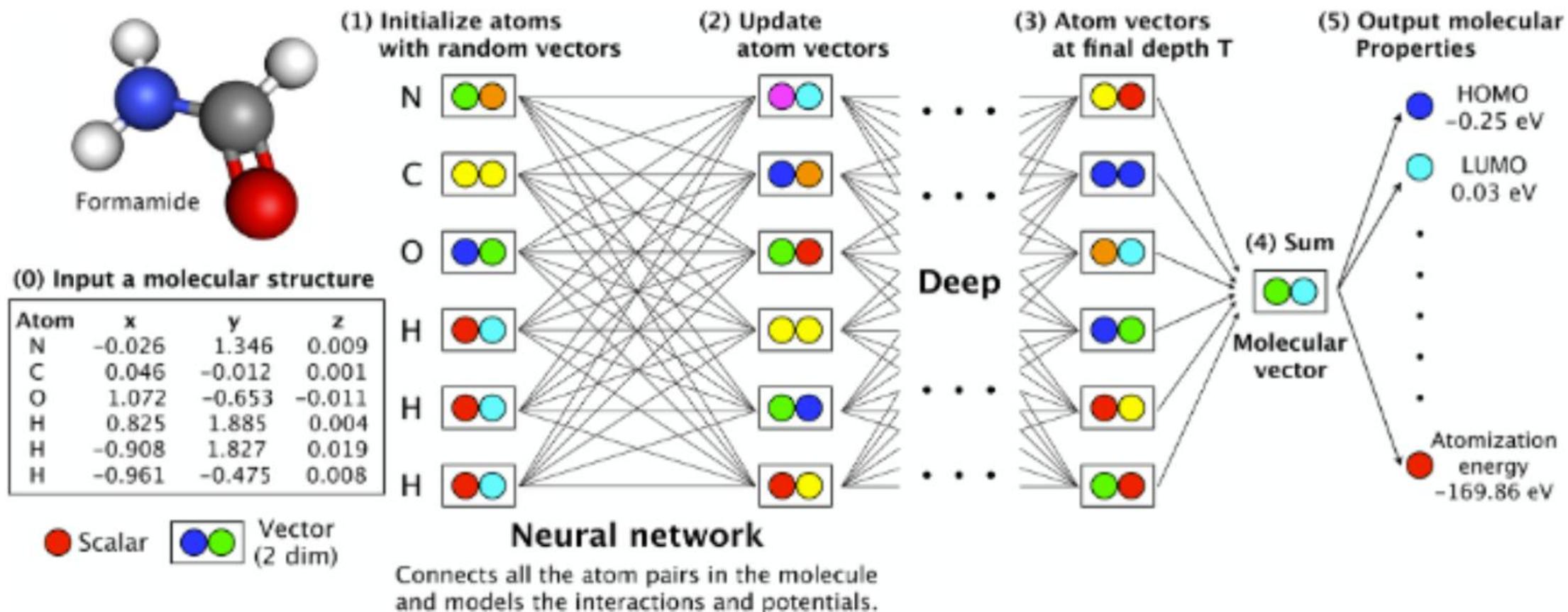
Arrieta AB et al. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI.

# Deepな研究 (1)



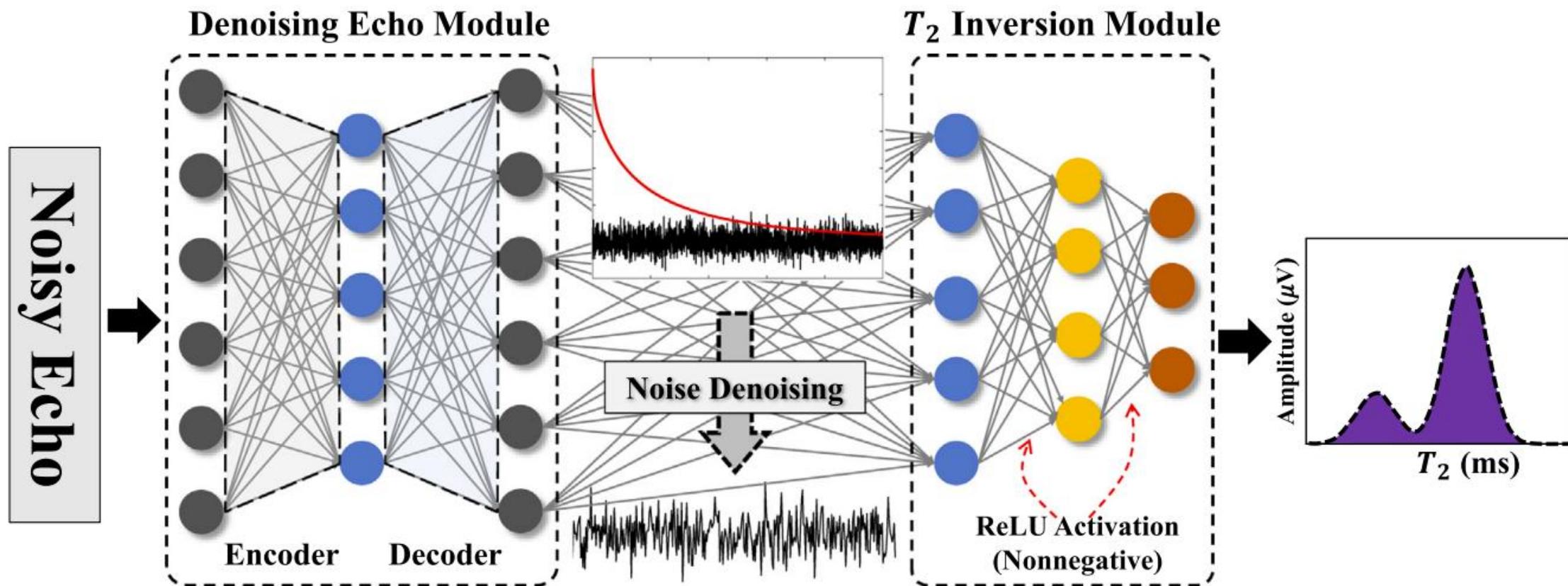
Jha D et al. (2018). Elemnet: Deep learning the chemistry of materials from only elemental composition.

# Deepな研究 (2)



Tsubaki M et al. (2018). Fast and accurate molecular property prediction: learning atomic interactions and potentials with neural networks.

# Deepな研究 (3)

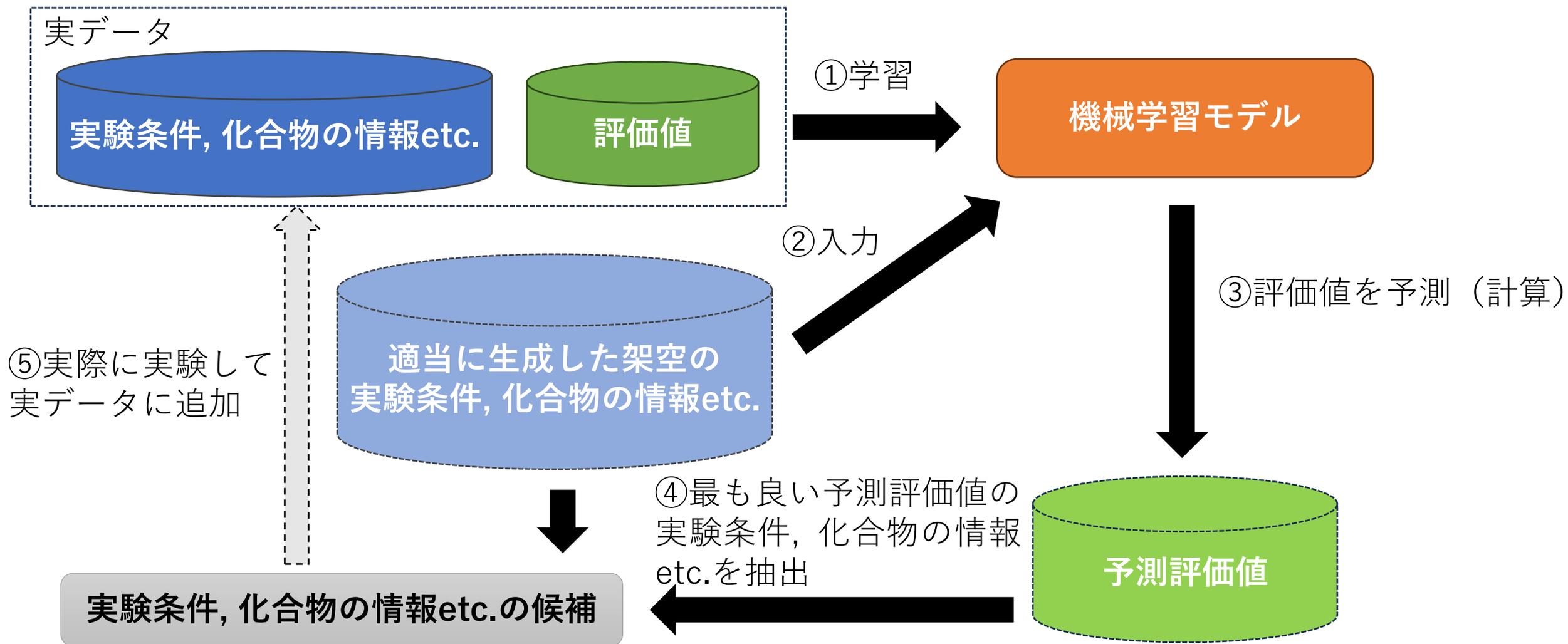


Luo G et al. (2023). A study on multi-exponential inversion of nuclear magnetic resonance relaxation data using deep learning.

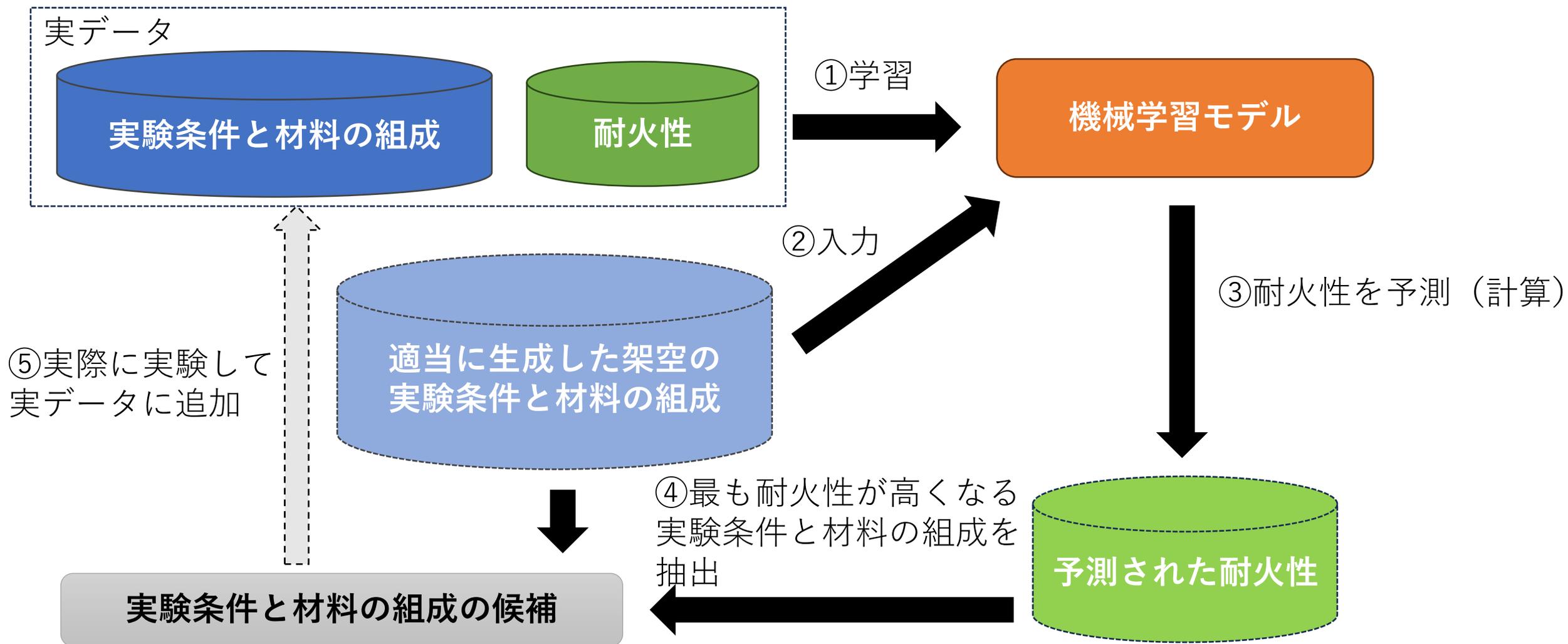
# 逆解析

- 最適な実験条件や化合物の組成などを探索する
  - 一般的な方法は意外とシンプルで…
1. まずはいくつかデータを集めて、機械学習モデルに学習させる
    - 機械学習モデルは別途選定する必要アリ
  2. 架空の実験条件や化合物などのデータを大量に生成する
  3. 学習済みモデルに生成したデータを入力し、予測値を取得する
  4. 一番良い予測値になるデータが次の実験候補！

# 逆解析の基本イメージ



# 逆解析の基本イメージ（耐火物ver.）



# ベイズ最適化

- 逆解析の一種
- 機械学習モデルに**ガウス過程回帰モデル**を用いる
- さきほどの③④の部分にベイズ最適化ならではの工夫がある
- **データが少ないうちから始められる！**
  - とある論文では、たった10件程度の実データからスタート
  - とはいえ、うまく学習させるために最初の実データの品質は重要
- **モデルはどうしてもよくて、単に実験を効率よく進めたいときに！**
- 評価値（目的変数）が複数ある場合は**多目的ベイズ最適化（MOBO）**
  - 例）安定性&耐火性の両方に優れた材料を探したいときに

# Multi-Sigma

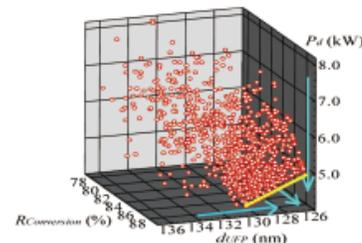
[https://www.lightstone.co.jp/multi\\_sigma/](https://www.lightstone.co.jp/multi_sigma/)

- まさに逆解析
- モデルは**NN**
- 最適化は**MOGA**
  - 多目的最適化の一種
- ついでに感度解析
- これらをWebで完結！

従来のPDCAサイクルにAIの手法を取り入れた  
**革新的実験計画法**で研究開発を超効率化

## 多目的遺伝的アルゴリズム (Multi-objective genetic algorithm : MOGA)

複数のアウトプットを同時に最適化  
最適な出力結果となるパラメータ条件の探索を行います。  
トレードオフの関係にある複数のアウトプットに対しても最適解の集合を取得できます。



## 感度解析

### 説明できるAI 要因分析

作成したAIモデルにおいて、各パラメータが各アウトプットにどの程度影響を与えているかを定量的に評価します。  
感度解析の手法を用いています。

Output1

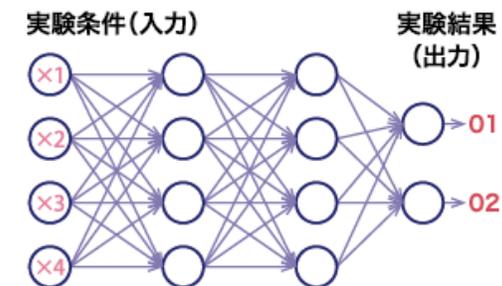
	貢献量[%]	正	負
Input1	45	40	5
Input2	35	10	25
Input3	20	20	0

## ニューラルネットワークモデル (Neural network model : NN)

### 20個のデータから高精度に予測できる理由

深層学習を使った解析は、学習時の条件(ハイパーパラメータ)を適切にチューニングしなければ高い精度を得られません。

**Multi-Sigmaは独自のオートチューニング機能を実装しており、小規模な実験データに対しても最適なハイパーパラメータを自動で探索し、過学習に陥ることなく高精度に予測します。☆国際特許取得済**



# 実演

# Python

- プログラミング言語の一種
  - プログラミング言語：C, C++, C#, Java, Java Script, Perl, Ruby, PHP, Swift, ...
- 比較的読みやすい言語
- プログラミングの面倒な部分は概ね自動化してくれている
- 初心者におすすめ
- 機械学習関係の既成プログラム（ライブラリ）が豊富

# 自動車の燃費予測とデータ解析

- Auto-mpg datasetを使って自動車の燃費予測とデータ解析を行う
- Google Colaboratory上で実行する

## Google Colaboratory

- ブラウザ上でPythonを実行することができるサービス
- Googleアカウントとインターネット環境があれば誰でも無料で使える
  - ※より高機能な有料プランもあります
- 一番面倒な環境構築をしなくていいことがメリット

# Auto-mpg dataset

- 1行目は変数名を羅列する
- 2行目以降は1行に1つのデータ

	A	B	C	D	E	F	G	H	I
1	mpg	cylinders	displacement	horsepower	weights	acceleration	year	origin	car_name
2	18	8	307	130	3504	12	70	USA	chevrolet chevelle malibu
3	15	8	350	165	3693	11.5	70	USA	buick skylark 320
4	18	8	318	150	3436	11	70	USA	plymouth satellite
5	16	8	304	150	3433	12	70	USA	amc rebel sst
6	17	8	302	140	3449	10.5	70	USA	ford torino
7	15	8	429	198	4341	10	70	USA	ford galaxie 500
8	14	8	454	220	4354	9	70	USA	chevrolet impala
9	14	8	440	215	4312	8.5	70	USA	plymouth fury iii
10	14	8	455	225	4425	10	70	USA	pontiac catalina
11	15	8	390	190	3850	8.5	70	USA	amc ambassador dpl
12	15	8	382	170	3562	10	70	USA	dodge challenger se



<https://www.kaggle.com/datasets/uciml/automp-g-dataset>

# 重回帰分析とSHAP

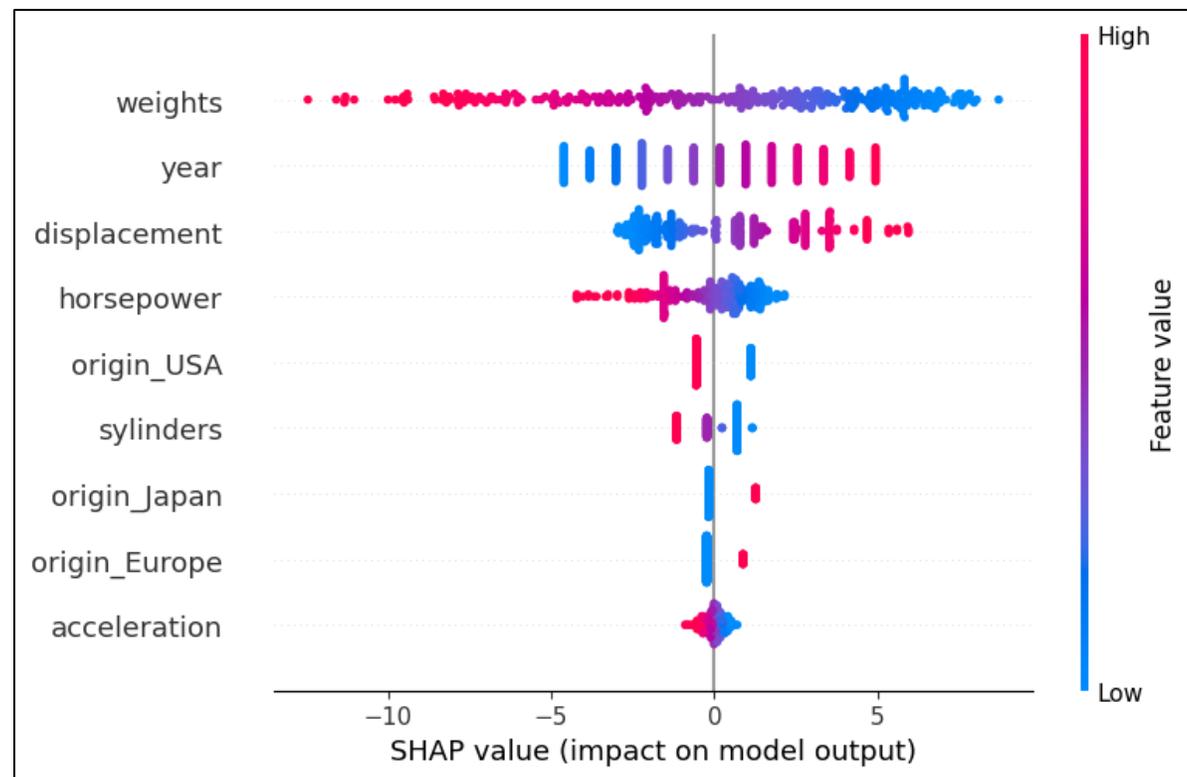
## 重回帰分析

変数間を繋ぐ数式を明らかに

```
mpg =
  -0.79 × cylinders
+ 2.40 × displacement
+ -1.35 × horsepower
+ -5.11 × weights
+ -0.26 × acceleration
+ 2.92 × year
+ 0.43 × origin_Europe
+ 0.57 × origin_Japan
+ -0.81 × origin_USA
+ 23.31
```

## SHAP

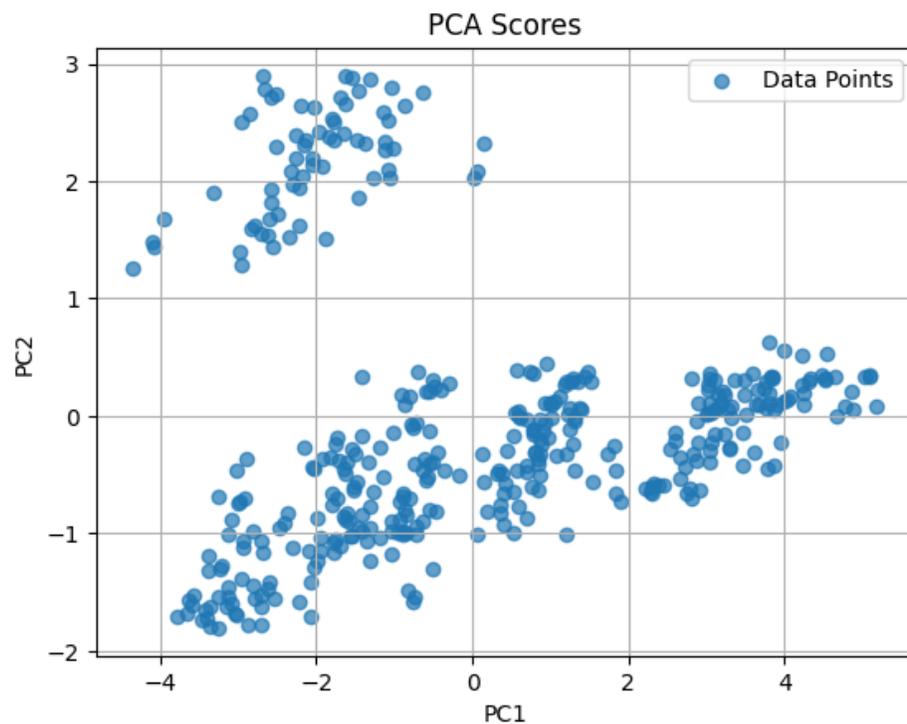
どの変数が重要なのかを明らかに



# 主成分分析とk-means

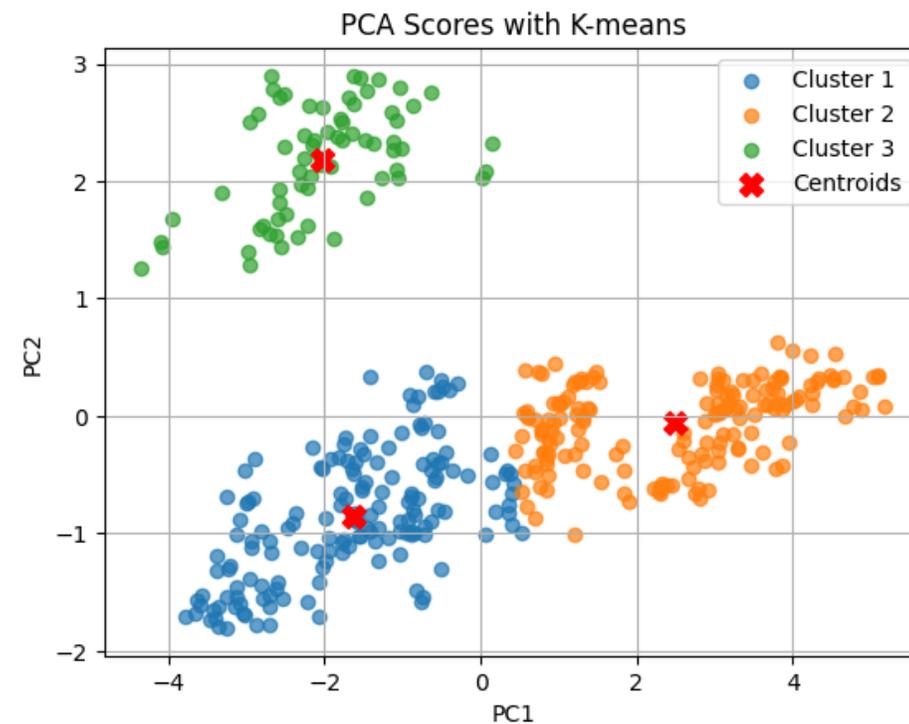
## 主成分分析

高次元のデータを低次元に



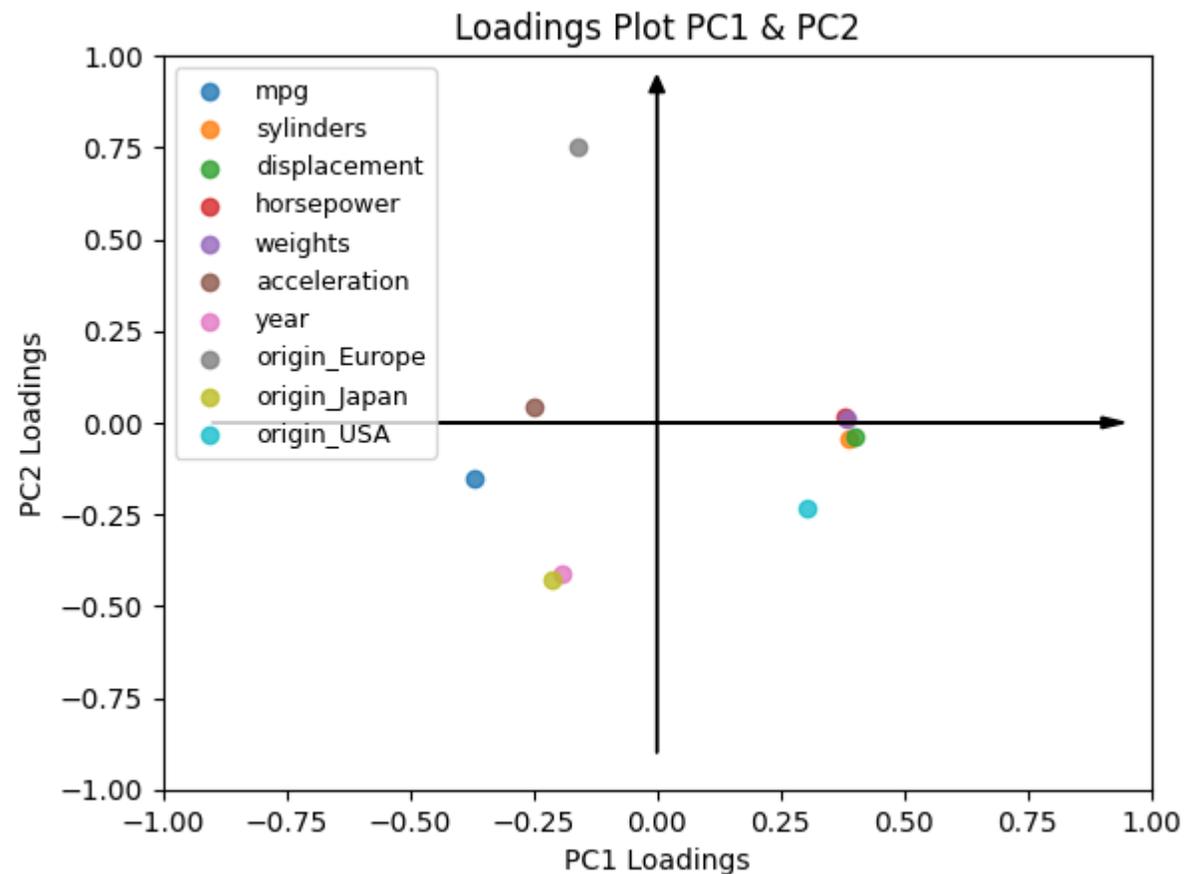
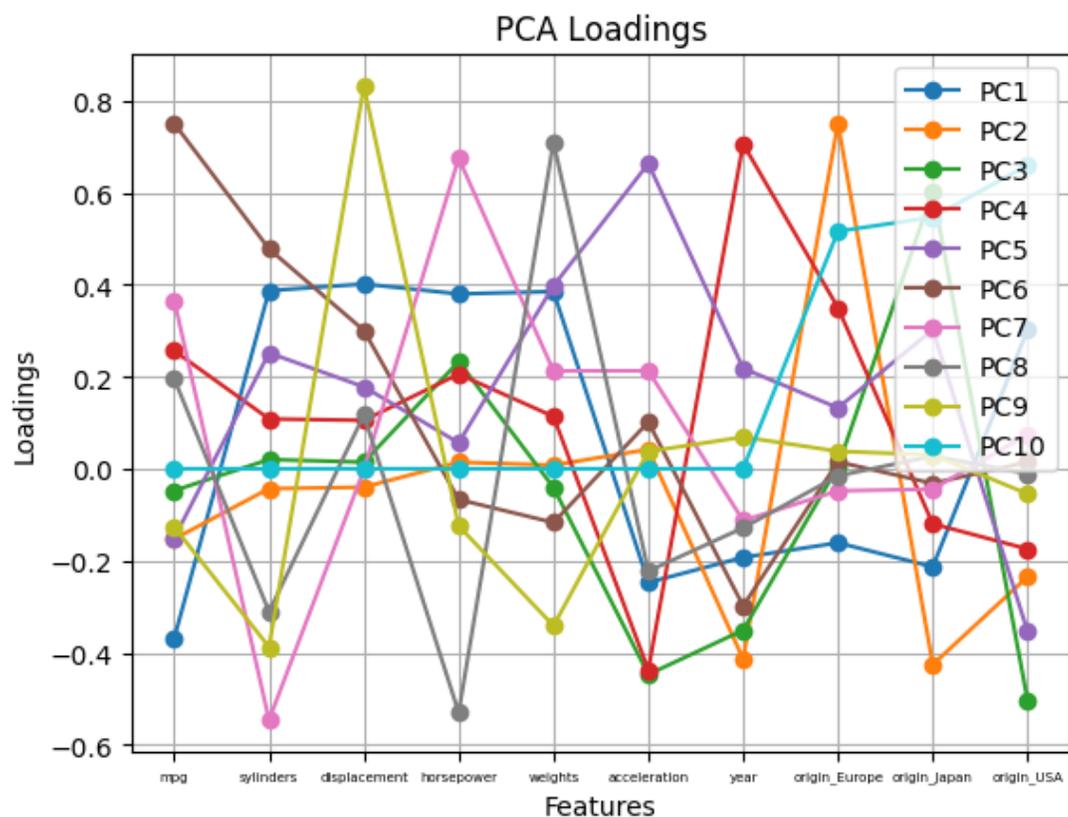
## k-means

データの集合をk個のグループに



# 主成分分析とローディング

データの分布と変数の関係を明らかに



# 実演

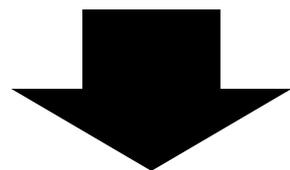
プログラムが欲しい方はこちら



[https://drive.google.com/drive/folders/15Gb3vbt3LqIKwMATF2lSkKe3\\_XXQbsDP?usp=sharing](https://drive.google.com/drive/folders/15Gb3vbt3LqIKwMATF2lSkKe3_XXQbsDP?usp=sharing)

# 生成AIを賢く使う

- プログラムを一から書くのは情報専門の人間でも大変です
- プログラムのひな型があるといいなあ...



**ChatGPT（生成AI）の出番！**

<https://chatgpt.com/>

Pythonで物性値予測を行うための重回帰分析のプログラムを書いて。



以下は、Pythonを使用して物性値予測を行うための重回帰分析プログラムのサンプルコードです。このコードでは、`pandas` を使ってデータを操作し、`scikit-learn` を使って重回帰分析を実行します。

## 必要なライブラリのインストール

以下のライブラリが必要です。まだインストールしていない場合は、次のコマンドでインストールしてください。

```
bash
```

 コードをコピーする

```
pip install pandas scikit-learn matplotlib
```

## 重回帰分析のコード

## 重回帰分析のコード

python

 コードをコピーする

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt

# サンプルデータの読み込みまたは生成
# CSVファイルから読み込む場合:
# df = pd.read_csv('data.csv')
# 以下は仮のデータフレーム例
data = {
    'Feature1': [1.0, 2.0, 3.0, 4.0, 5.0],
    'Feature2': [2.0, 4.0, 6.0, 8.0, 10.0],
    'Property': [1.1, 2.0, 2.9, 4.1, 5.0] # 物性値
}
df = pd.DataFrame(data)
```

```
# 説明変数（特徴量）と目的変数（物性値）の分割
X = df[['Feature1', 'Feature2']] # 説明変数
y = df['Property']              # 目的変数

# データの分割（80%を学習データ、20%をテストデータ）
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# モデルの作成と学習
model = LinearRegression()
model.fit(X_train, y_train)

# テストデータで予測
y_pred = model.predict(X_test)

# モデル評価
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

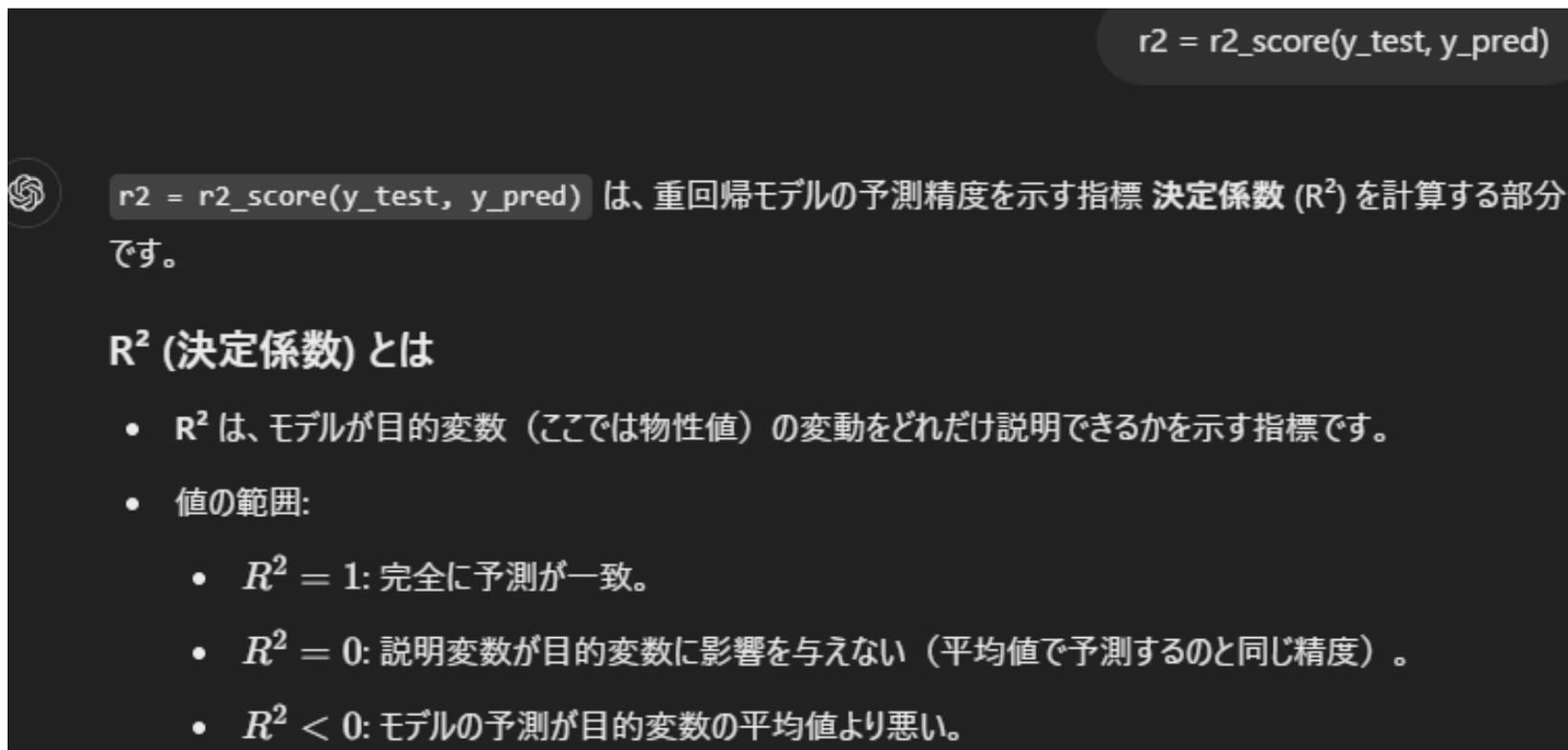
```
print("Mean Squared Error (MSE):", mse)
print("R-squared (R2):", r2)

# 回帰係数と切片
print("Coefficients:", model.coef_)
print("Intercept:", model.intercept_)

# 結果の可視化 (テストデータの実測値 vs 予測値)
plt.scatter(y_test, y_pred)
plt.xlabel("Actual Property Values")
plt.ylabel("Predicted Property Values")
plt.title("Actual vs Predicted Property Values")
plt.grid()
plt.show()
```

# その他生成AIの使い方

- エラーが出たプログラムを修正してもらう
- よくわからない部分を解説してもらう etc.



`r2 = r2_score(y_test, y_pred)`

`r2 = r2_score(y_test, y_pred)` は、重回帰モデルの予測精度を示す指標 **決定係数 ( $R^2$ )** を計算する部分です。

**$R^2$  (決定係数) とは**

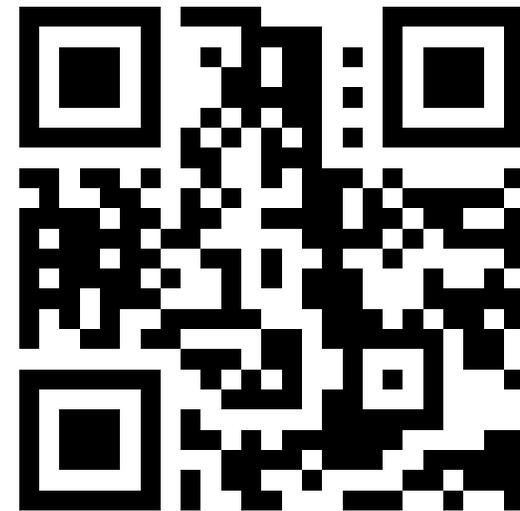
- $R^2$  は、モデルが目的変数（ここでは物性値）の変動をどれだけ説明できるかを示す指標です。
- 値の範囲:
  - $R^2 = 1$ : 完全に予測が一致。
  - $R^2 = 0$ : 説明変数が目的変数に影響を与えない（平均値で予測するのと同じ精度）。
  - $R^2 < 0$ : モデルの予測が目的変数の平均値より悪い。

# FreeDSのご紹介

- Excelファイル (.xlsx) をアップロードするだけで機械学習ができる！
  - ちょうど今日の実演内容みたいなことを、ほぼ自動でできます
- 完全無料
- 機械学習の感触を確かめたい方向け
  - ヘビーな使い方は想定しておりません
- サンプルデータセットもあるのでお試しください



**FreeDS**



<https://trklibrary.com/ds/>

# さいごに

AIに関する質問を何でも受け付けています  
お気軽にご連絡ください

torii@tokushima-u.ac.jp

本講演がみなさんの研究の一助となれることを願っています  
ありがとうございました