THE 24TH BEHAVIOR MODELING SUMMER SCHOOL

Statistical Estimation with Machine Learning

Junji Urata

September 22, 2025

University of Tsukuba

Outline

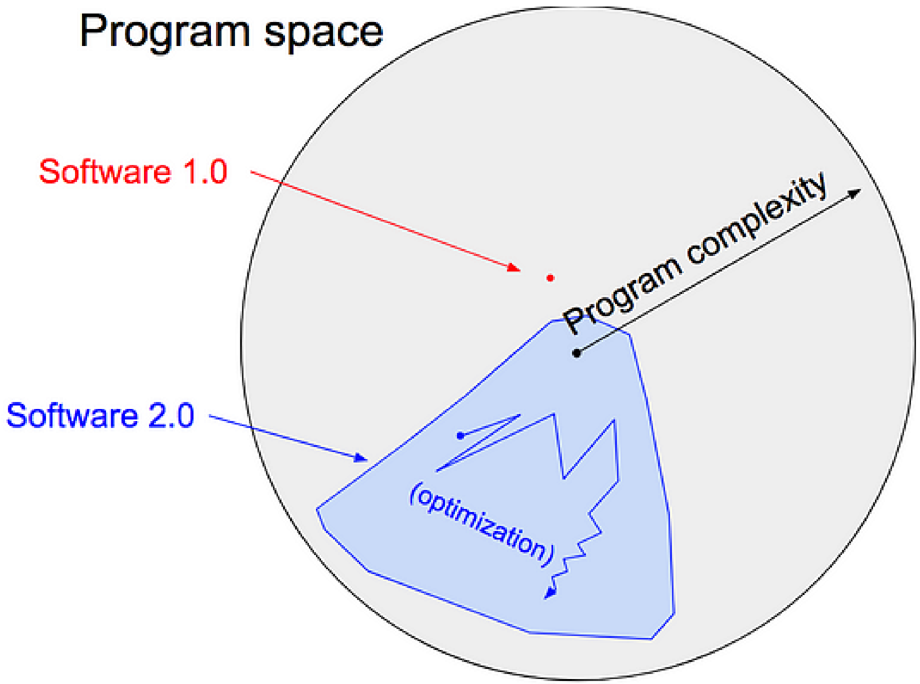
1. Introduction
2. Un-supervised learning
3. Supervised learning
    1. Data preparation
    2. Evaluation
    3. Estimation
    4. Model
        1. Neural Network
        2. Support Vector Machin

What is Machine Learning?

Arthur Samuel (1959)

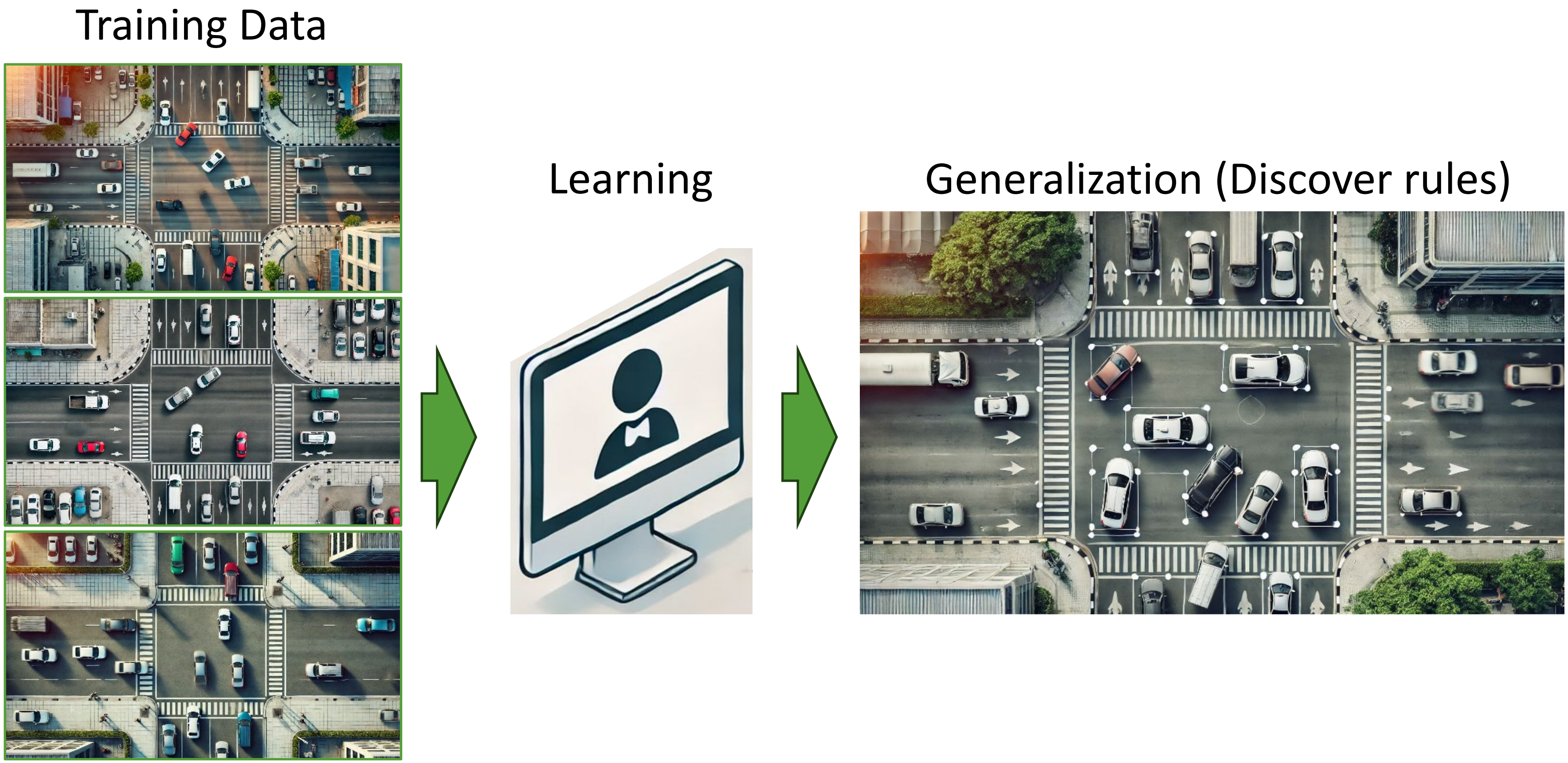
> "Field of study that gives computers the ability to learn without being explicitly programmed"

Software 2.0 *Andrej Karpathy (2017)*



What is Machine Learning?

- In Machine learning, we program the method of learning itself so that the computer can discover rules from accumulated data
- The goal of machine learning is to correctly predict results even for unknown data that is not in the training data



Training Data

Learning

Generalization (Discover rules)

* Images are made in ChatGPT 4o

Applying Machine learning for what?

Demand forecasting

- The power demand for the day is predicted based on the weather.
- Predicting road traffic demand of each link

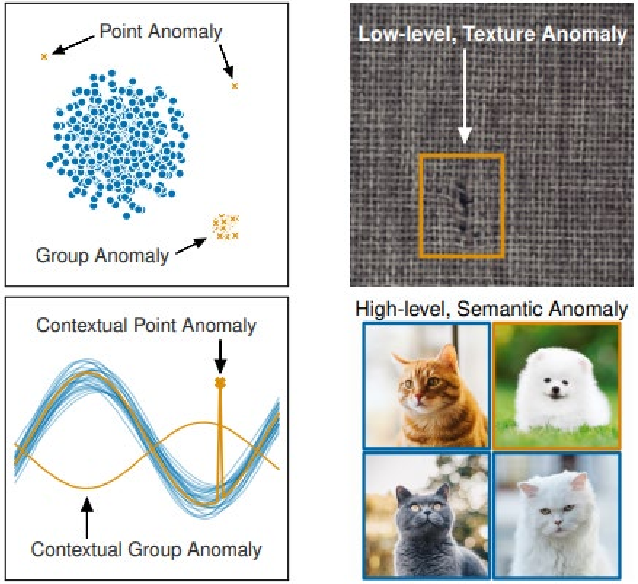
Product recommendations

- Displaying recommended products on Amazon based on past purchase data

-Recommending transportation options in Mobility as a Service (MaaS)

Anomaly detection

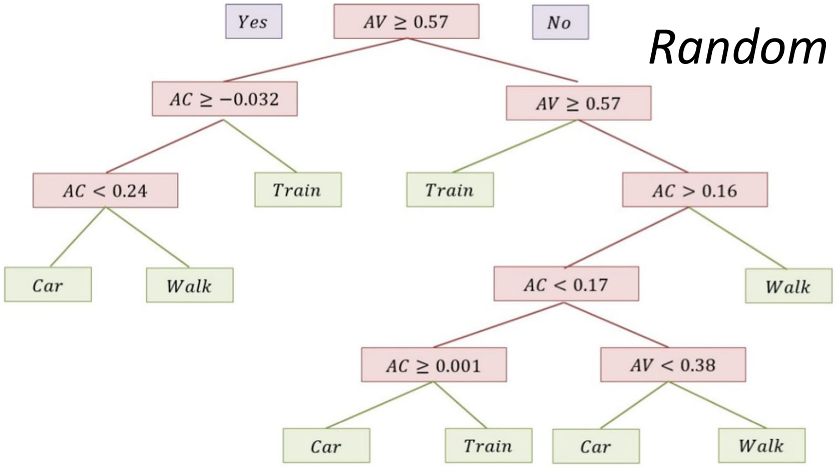
- Predicting printer failures in advance based on past failure data
- Detecting congestion or traffic accidents



Ruff, Lukas, et al. "A Unifying Review of Deep and Shallow Anomaly Detection." arXiv preprint arXiv:2009.11732 (2020).

Machine learning for what?

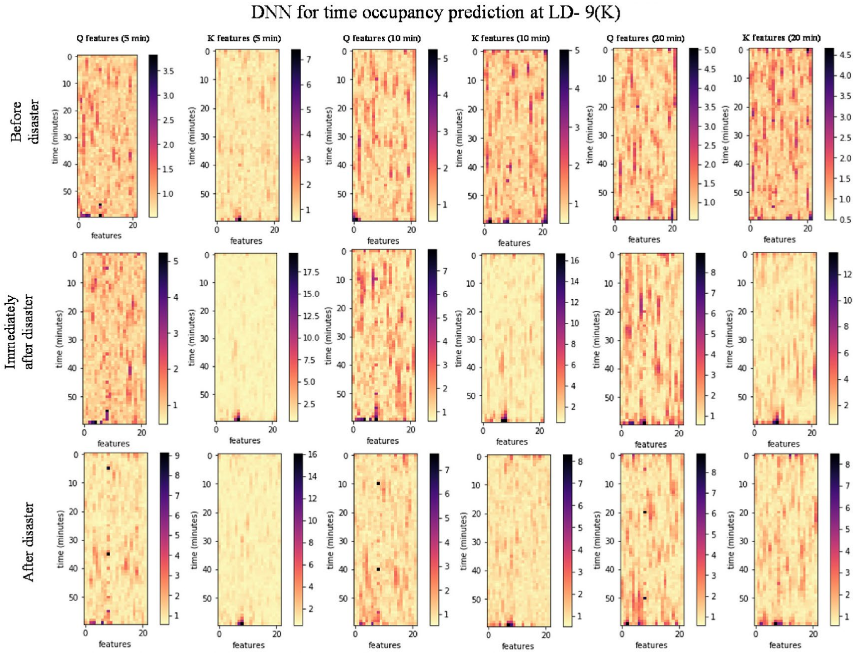
Shafique & Hato (2015): Traffic mode detection



AV = Average acceleration in vertical direction (G)
AC = Average acceleration in cross-wise direction (G)

Random Forest

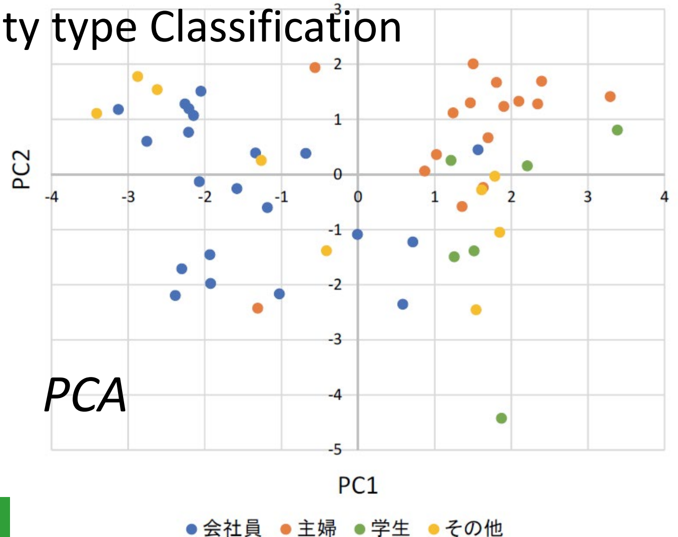
Chikaraishi, Varghsee, Urata et al. (2020):
Time Occupancy predition



- X-axis: 22 features; 1-22 represent either Q or K values of the respective loop detector
- Y- axis: Time in minutes (0-60)

Deep Neural Network

Sueki, Hara, Sasaki et al. (2018):
Activity type Classification



PCA

Machine learning Types

Supervised learning

- To learn the true relationship between input and output from the training data.
- The training data consists of pairs of input data and corresponding output values.

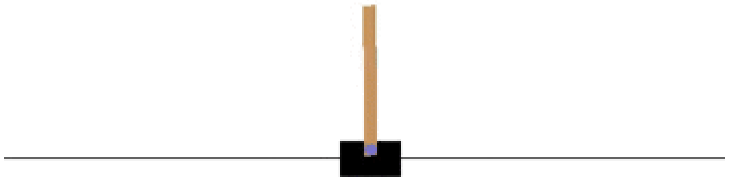
Unsupervised learning

- To discover useful knowledge from the data
- Using only input data in the training process

Reinforcement learning

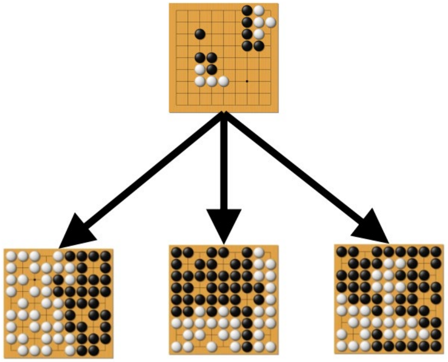
- Rewards are given for taking specific actions.
- To learn a set of actions that maximizes the cumulative reward over time.

CartPole problem



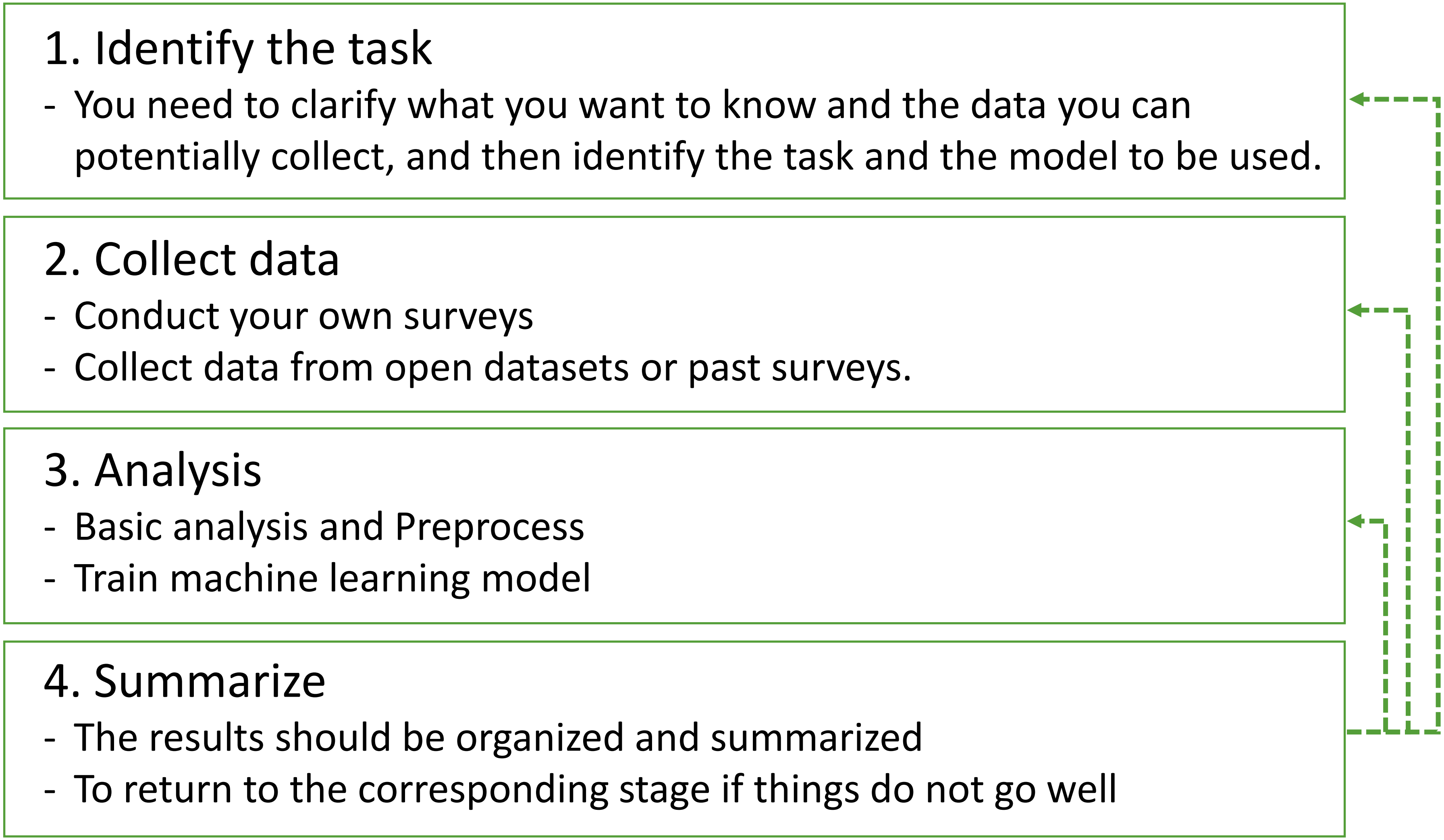
https://blog.brainpad.co.jp/entry/2017/02/24/121500

Igo game



美添（2019）

Flowchart of Machine Learning



1. Identify the task

- You need to clarify what you want to know and the data you can potentially collect, and then identify the task and the model to be used.

2. Collect data

- Conduct your own surveys
- Collect data from open datasets or past surveys.

3. Analysis

- Basic analysis and Preprocess
- Train machine learning model

4. Summarize

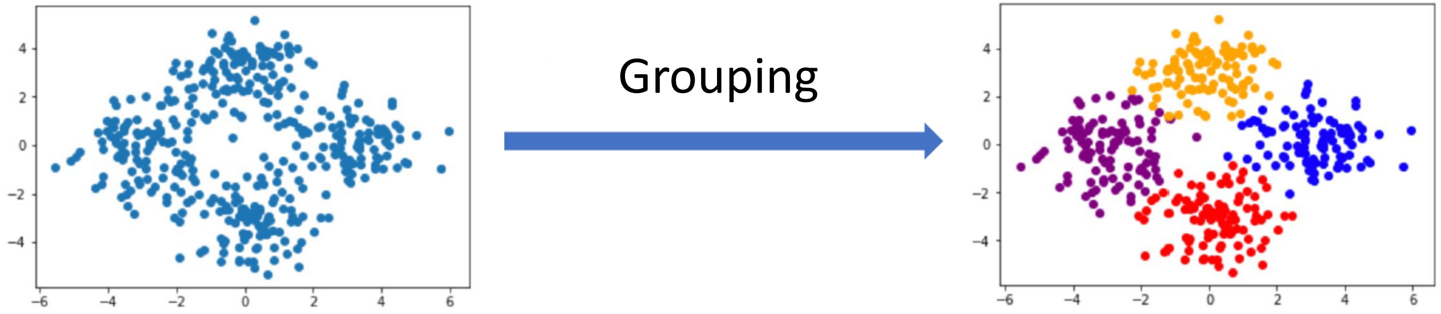
- The results should be organized and summarized
- To return to the corresponding stage if things do not go well

Un-supervised Learning

- Unsupervised learning aims to estimate the underlying structure or processes that generate the input data
- The input data consists of a collection of feature vectors

Ex.

Clustering is a method that groups data based on similarities



Grouping

you can cluster

- people based on previous behaviors and personal attributes
- zones based on attributes of the facilities and people who stay.

Outline

1. Introduction
2. Un-supervised learning
3. Supervised learning
    1. Data preparation
    2. Evaluation
    3. Estimation
    4. Model
        1. Neural Network
        2. Support Vector Machin

Supervised Learning

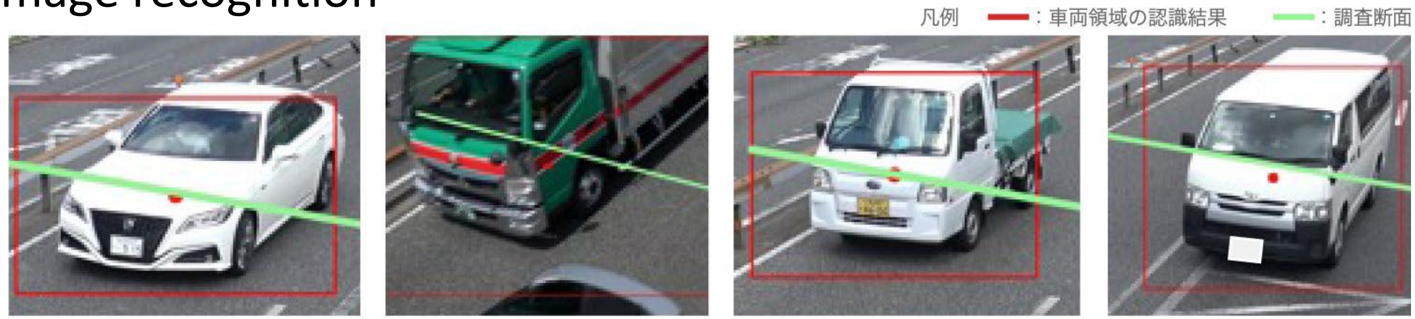
The goal is to build a model that can correctly predict y from **x** by learning from this input data.

The data in supervised learning consists of

- **x**: features of the data
- y: target value.

Ex. Image recognition

x



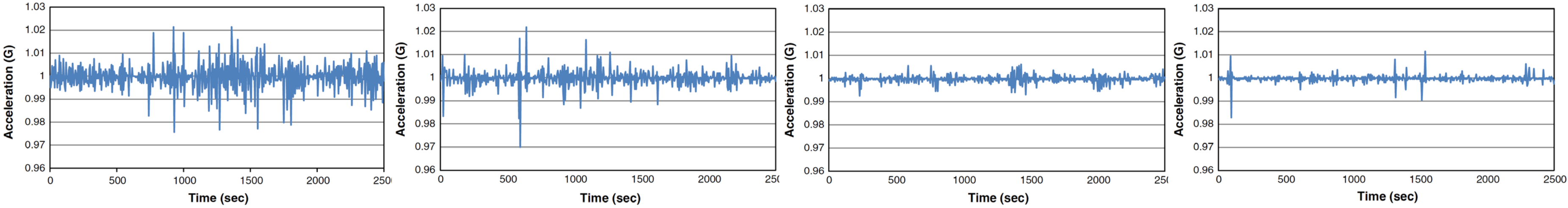
y　Small vehicle　Large vehicle　Large vehicle　Small vehicle

https://www.intelligentstyle.co.jp/product/aitraffic/

Ex. Traffic Mode detection

Shafique & Hato (2015)

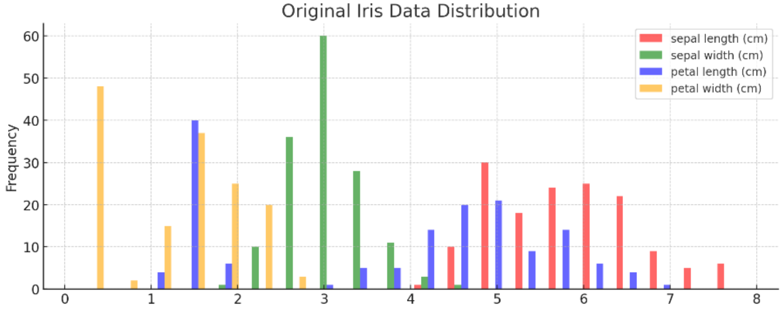
x



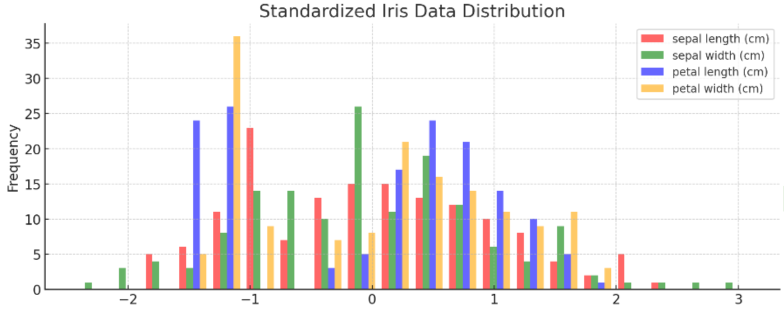
y　Walk　Bicycle　Car　Train

Preprocess

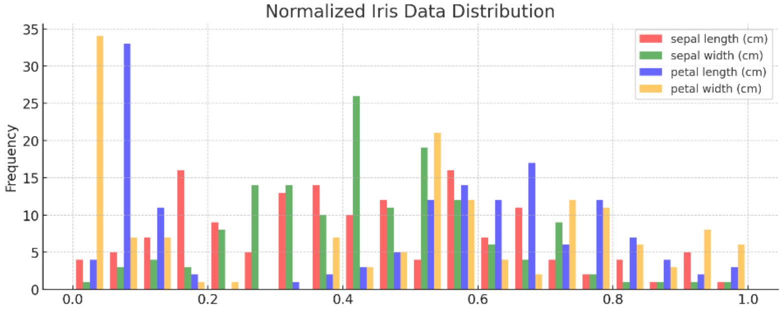
- The data may contain both continuous data and categorical data
  - Categorical data need to be transformed by creating dummy variable
- Major Preprocessing
  - Standardization: mean of 0 & variance of 1
  - Normalization: range [0, 1] or [-1, 1]
- The purpose is to make the absolute values and distributions of the various variables more comparable. By doing this, the model estimation process proceeds more smoothly



Standardization



Normalization



* Images are made in ChatGPT 4o

Model: Regression and Classification

- If the dependent variable y in the training data is continuous, a regression model is built.
- If y is discrete, a classification model is constructed.

Ex. Regression

- Sales forecast: y = Net sales of stores/day, x = Store square footage, type of business, weather, day of the week
- Bus demand: y = Number of passengers/h, x = Routes, temperatures, rain, school vacations, day of the week

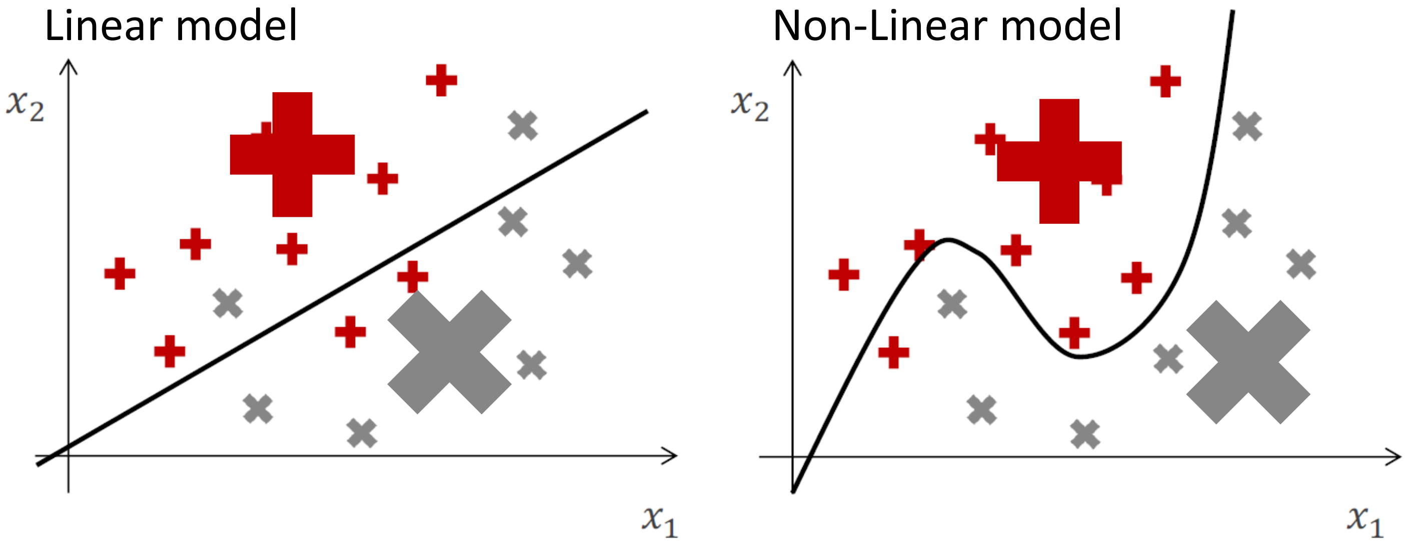
Ex. Classification

- Diseased: y = diseased or not, x = Blood test, saliva test, temperature, facial expression, radiographs
- Traffic mode: y = mode, x = Travel time, fares, access & egress time, temperature, driver license, accompany person

Classification

- Predictions are made using a model that outputs continuous values $f_w(\mathbf{x})$ based on the model's parameters **w**
- The parameter **w** are estimated via training the model using training data

Ex. Classification in Training data



Linear model

Non-Linear model

x_1, x_2: explanatory variable Labels(y): ✚ = 1, ✖ = -1

Objective function (Loss function)

The parameter estimation for the model evaluates how well the model fits the data

Loss function L(**w**)
= Distance between the model output f_w(**x**) and the label y from the training data

Obtain parameter **w** by $\min_{\boldsymbol{w}} \sum_i L(\boldsymbol{w}, (x, y)_i)$

Ex. Hinge loss

$$\text{Hinge Loss L} = \max(0, 1 - y_i \hat{y}_i)$$

y_i: (observed)Label, $\hat{y}_i$: model output
$y_i, \hat{y}_i = (-1, 1)$

Ex. Cross Entropy loss

$$\text{Cross Entropy Loss L} = -\left[y_i \log(\hat{p}_i) + (1 - y_i)\log(1 - \hat{p}_i)\right]$$

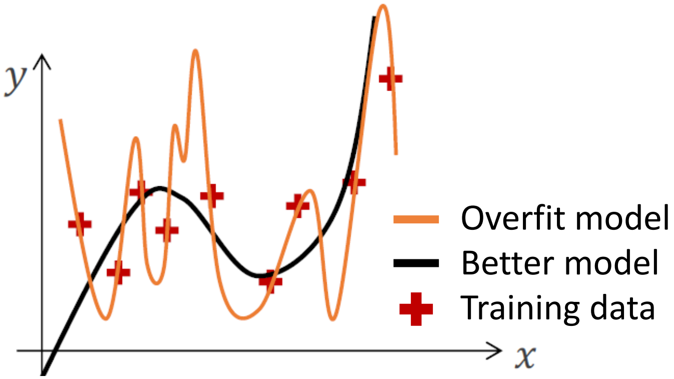
$\hat{p}_i$: model output (probability)
$\hat{p}_i \in (0, 1)$

Overfit and regularization

What is overfit?

- Overfitting occurs when the model becomes overly complex and fits too well to the training data

- Overfitting makes the model ineffective for prediction



To reduce the complexity

Regularization?

- We add the regularization term on the objective function

$$\min_{\boldsymbol{w}} \sum_i L(\boldsymbol{w}, (x, y)_i) + \lambda R(\boldsymbol{w})$$

L2 (Ridge regression): $R(w) = \sum_j w_j^2$

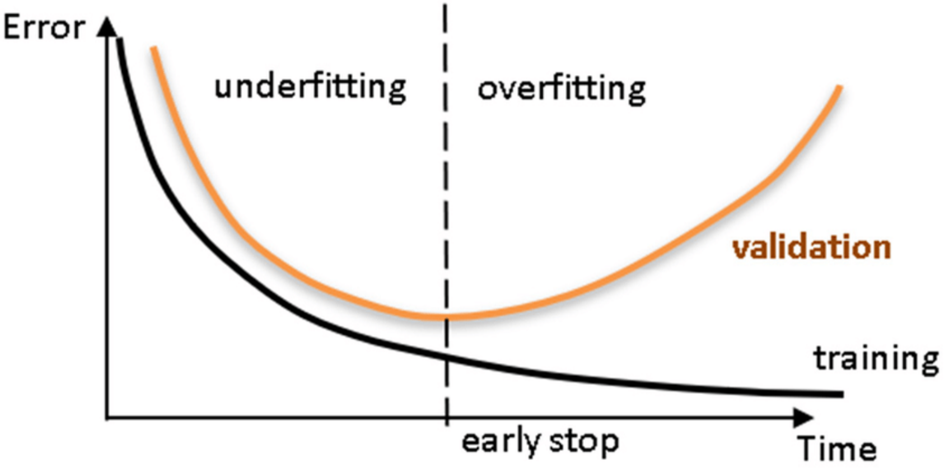
L1 (Lasso regression): $R(w) = \sum_j |w_j|$

Cross-Validation

Early Stopping

1. The data is split into training and validation sets during parameter estimation
2. The parameters are estimated using the training data
3. To calculate the loss function with the estimated parameters using the validation sets
4. Iterate 2 & 3, and then the final model is selected based on the parameters that perform well on the validation data

Training data	Validation data	Test data



Graph by Abambres, & Lantsoght, Eva. (2019). ANN-Based Fatigue Strength of Concrete under Compression. Materials. 12. 3787. 10.3390/ma12223787.

The model is ultimately evaluated based on its accuracy on the test data, which is completely independent of the data used for parameter estimation.

Model Assessment

We talked about using a loss function to estimate the model parameters. There are a number of approaches to assessing the model obtained.

1. Accuracy: The percentage of the test data that was correctly classified

$$Accuracy = \frac{\text{number of correctly classified data}}{\text{number of all samples}}$$

2. Confusion matrix: where predicted outcomes are evaluated in comparison to actual correct values

	予測：正 Prediction: positive	予測：負 Prediction: Negative
正解：正 Actual value: Positive	TP（真陽性）	FN（偽陰性）
正解：負 Actual value: Negative	FP（偽陽性）	TN（真陰性）

3. Precision：The percentage of instances predicted as positive that are truly positive

$$Precision = \frac{\text{TP（真陽性）}}{\text{TP(真陽性)} + FP\text{（偽陽性）}}$$

4. Recall：The proportion of true positives that are correctly identified

$$Recall = \frac{\text{TP（真陽性）}}{\text{TP(真陽性)} + FN\text{（偽陰性）}}$$

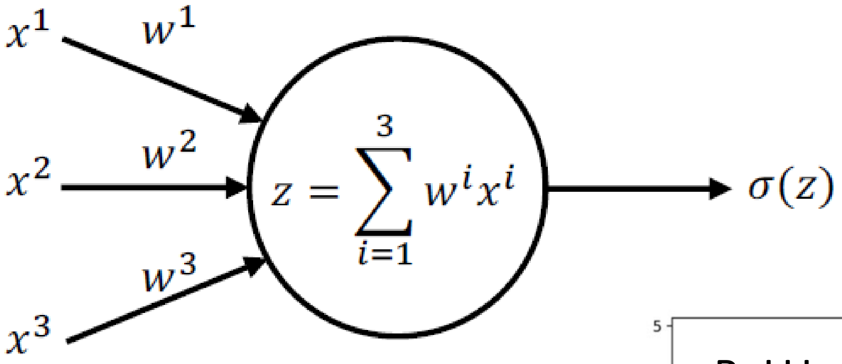
Models – Supervised learning

1. Neural Network (NN)

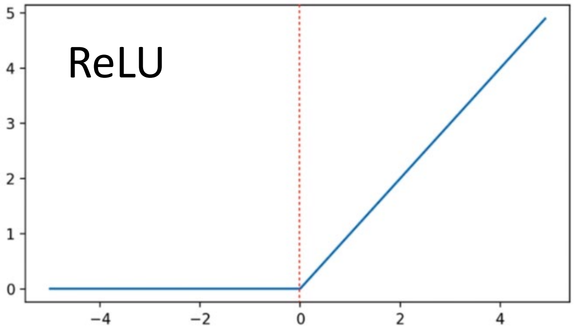
2. Support Vector Machine (SVM)

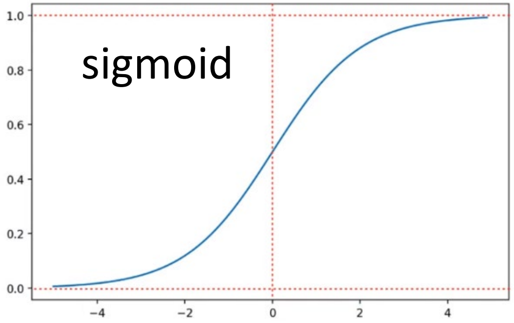
What is Neural Network?

- Neural networks are machine learning models inspired by the brain.
- Neurons (nodes) are connected through weighted edges, forming a network.
- Neurons receive signals z from other connected neurons according to the weights w on these edges, and then apply a transformation called an activation function σ to output a signal.
- Activation function – ReLU function, sigmoid function etc



$$\sigma(z) = \begin{cases} \max\{0, z\} & \text{(ReLU)}, \\ \dfrac{1}{1+e^{-z}} & \text{(sigmoid)}. \end{cases}$$



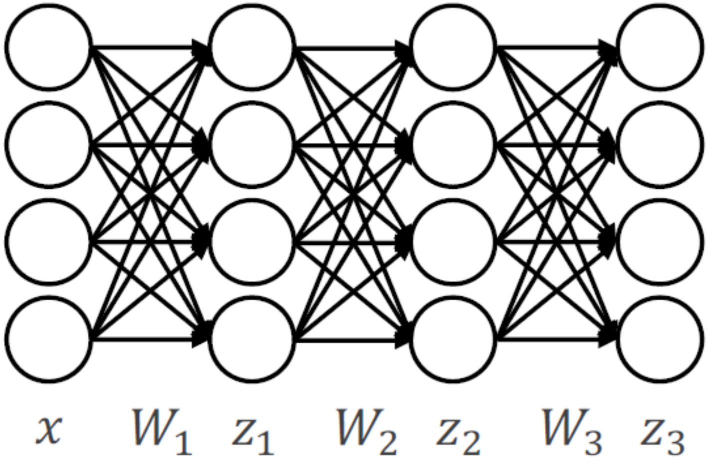


https://onl.tw/r3E6iRE

Deep Neural Network

- Neurons are stacked in parallel to form layers.
- By stacking multiple layers of neurons, a deep neural network (DNN) is created.
- By incorporating nonlinear activation functions such as ReLU and sigmoid, complex nonlinear transformations can be achieved.

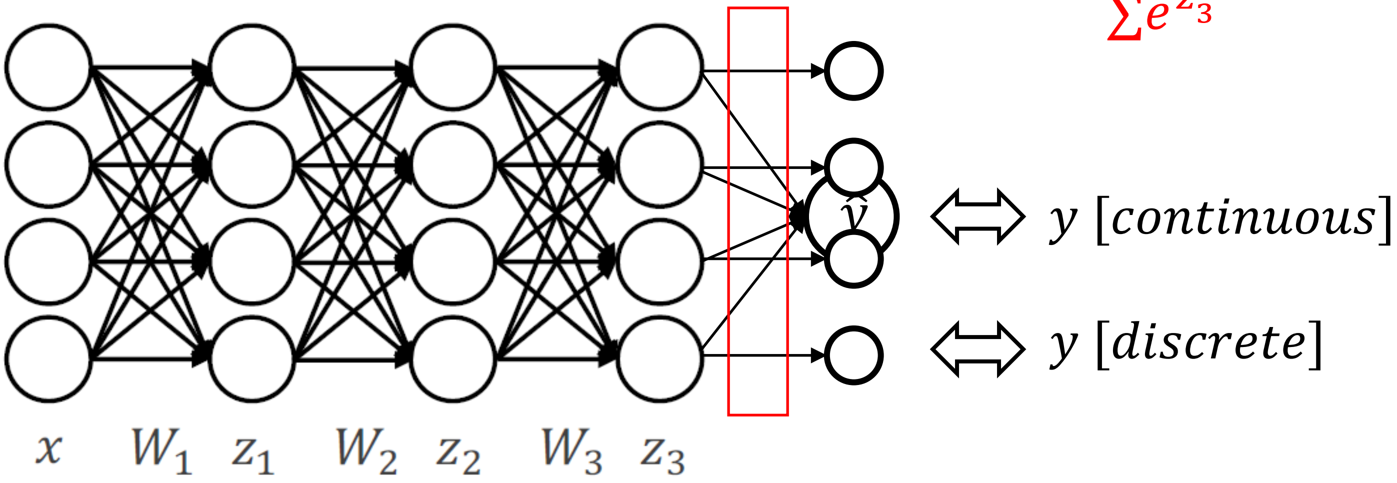
$$x \to W_1 x \to \sigma_1(W_1 x) \to W_2 \sigma_1(W_1 x) \to \sigma_2(W_2 \sigma_1(W_1 x))$$



Last layer for classification

- The final layer of a neural network is configured according to the task.
- For regression tasks, it is sufficient to apply a transformation that consolidates the output into a single node.
- In classification problems, the softmax function is commonly used to calculate the probability of belonging to each class.

$$softmax\ (z) = \frac{e^{z_3^j}}{\sum e^{z_3^j}}$$



$\hat{y}$
$\Leftrightarrow$ $y\ [continuous]$
$\Leftrightarrow$ $y\ [discrete]$
x
W_1
z_1
W_2
z_2
W_3
z_3

Why you can be DEEP?

Backpropagation

b: bias (parameter)
y_i: Observed Label
$\hat{y}_i$: model output
$y_i, \hat{y}_i = (0, 1)$

Loss function $F(\boldsymbol{w}, b) = \prod_i P(k_i|\boldsymbol{x}_i) = \prod_i \hat{y}_i^{y_i}(1 - \hat{y}_i)^{1-y_i}$

Log (loss function) $E(\boldsymbol{w}, b) = -\log F(\boldsymbol{w}, b) = -\sum_n \{y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)\}$

To minimize the loss function, parameters are updated in the direction of the gradient

$$\boldsymbol{w}^{t+1} = \boldsymbol{w}^t - \eta \frac{\partial E(\boldsymbol{w}, b)}{\partial \boldsymbol{w}} \qquad b^{t+1} = b^t - \eta \frac{\partial E(\boldsymbol{w}, b)}{\partial b}$$

η: learning rate (hyper parameter)

The derivative calculation is quite simple, allowing for quick parameter updates

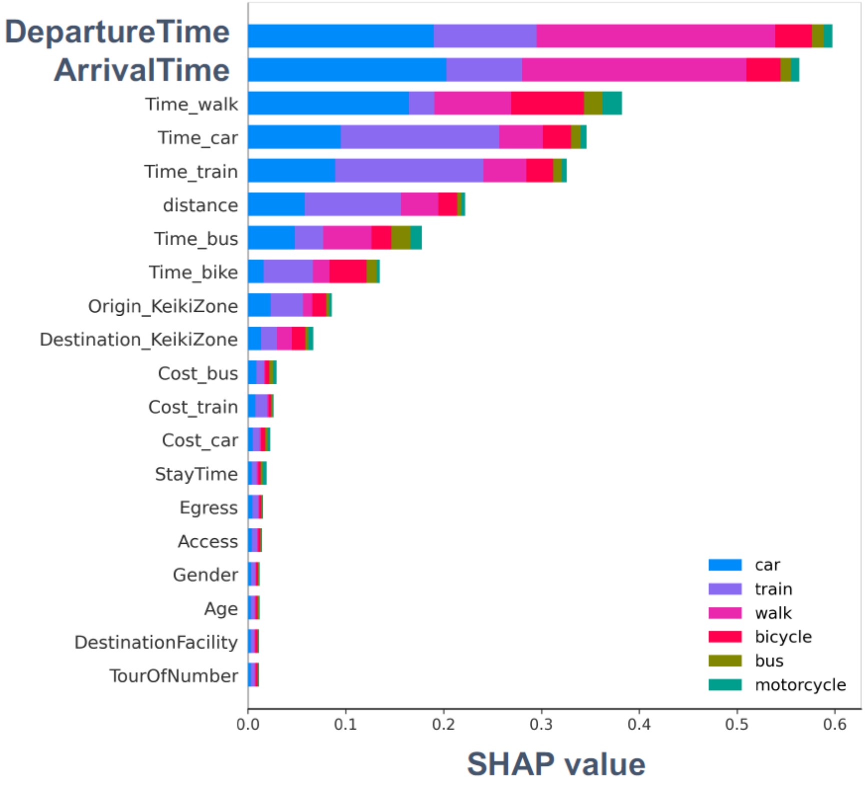
$$\frac{\partial E(\boldsymbol{w}, b)}{\partial \boldsymbol{w}} = \cdots = -\sum_n (y_i - \hat{y}_i)\boldsymbol{x}_i$$

(In sigmoid function)

In addition, we can use GPUs for parallel computing

SHAP (SHapley Additive exPlanations)

- SHAP is used to clarify how each feature contributes to the prediction results.
- It quantifies the impact of each feature on the prediction.
- SHAP makes it possible to interpret machine learning models, which often are regarded as black boxes.
- The values provided by SHAP are similar to elasticity in discrete choice models.



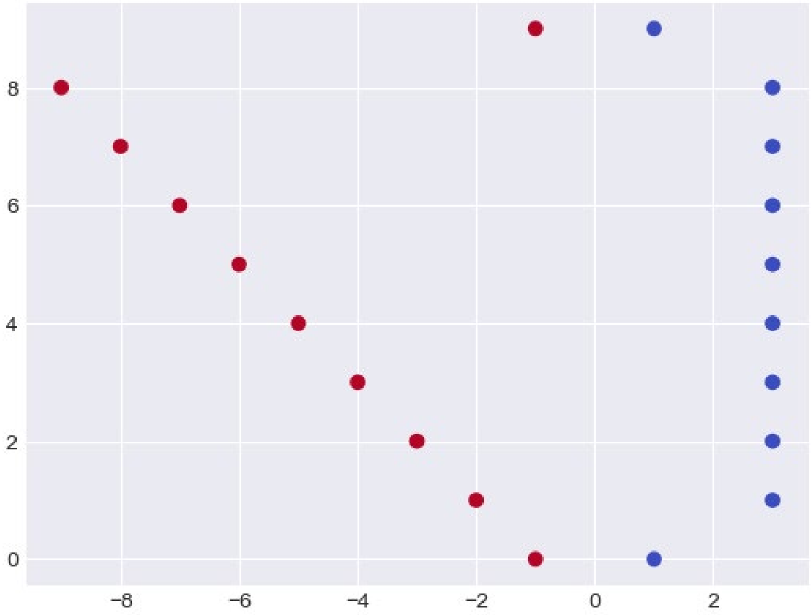
Mode choice model with NN (Yaginuma 2023)

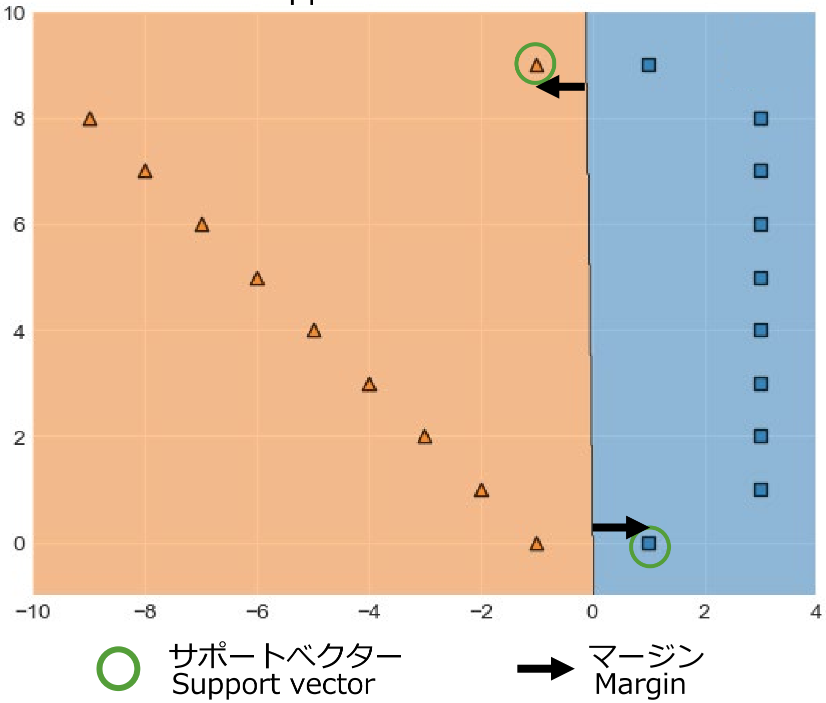
Support Vector Machine (SVM)

A common machine learning model used by the model to learn the boundary line for classifying data according to the rules governing margin maximization.

- “Optimal” refers to the situation where the margin (the distance between the decision boundary and the closest data points) is at its maximum.
- Maximizing this distance helps improve the generalization ability of the model.

※ Generalization: how applicable the model is to unknown data

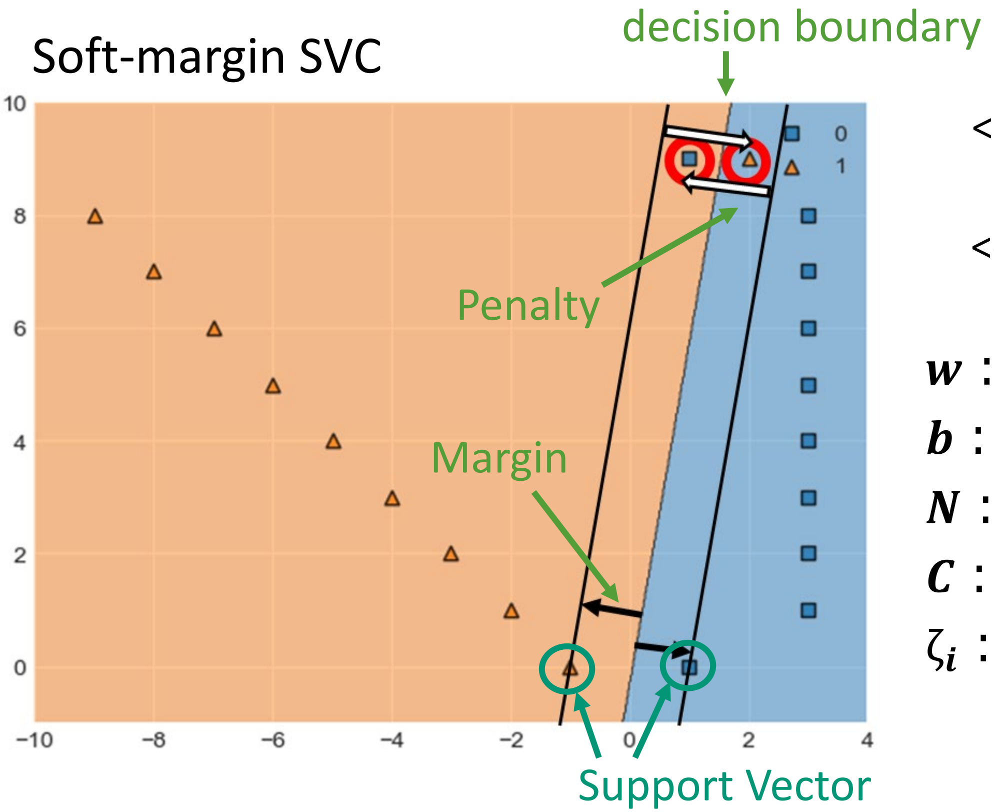




Support Vector Machine Classification (SVC)

- Under the rule of margin maximization, the decision boundary that classifies the data is learned.
- The goal is to find the decision boundary that maximizes the margin. However, in some cases, the data cannot be perfectly separated by the boundary.
- In such cases, an objective function is set to minimize a penalty term.

Soft-margin SVC



Inverse of margin

<Objective function> $\min_{\boldsymbol{w},\boldsymbol{b},\zeta} \frac{\mathbf{1}}{\mathbf{2}} \|\boldsymbol{w}\|^{\mathbf{2}} + \boldsymbol{C} \sum_{\boldsymbol{i}=\mathbf{1}}^{\boldsymbol{N}} \zeta_{\boldsymbol{i}}$

<constraint> $\boldsymbol{y}_{\boldsymbol{i}}\left(\boldsymbol{w}^{\boldsymbol{T}} \cdot \boldsymbol{x}_{\boldsymbol{i}} + \boldsymbol{b}\right) \geq \mathbf{1} - \zeta_{\boldsymbol{i}} \quad \forall \boldsymbol{i}$

$\boldsymbol{w}$: Normal vector to the decision boundary

$\boldsymbol{b}$: bias term of the decision boundary

$\boldsymbol{N}$: Number of data

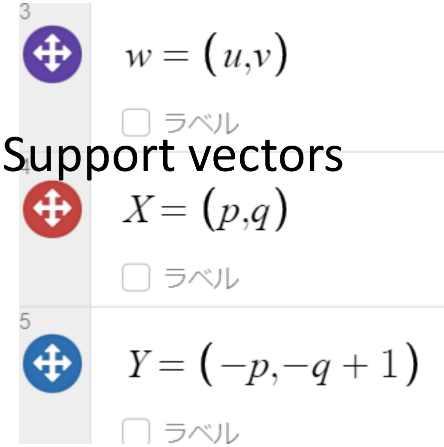
$\boldsymbol{C}$: Weight of parameter

$\zeta_{\boldsymbol{i}}$: Penalty term (usually, apply Hinge function)

$$\zeta_{\boldsymbol{i}} = \boldsymbol{f}(\mathbf{1} - \boldsymbol{y}_{\boldsymbol{i}}(\boldsymbol{w}^{\boldsymbol{t}} \boldsymbol{x}_{\boldsymbol{i}} + \boldsymbol{b}))$$

Support Vector Machine

Normal vector



3

$w = (u,v)$

□ ラベル

Support vectors

$X = (p,q)$

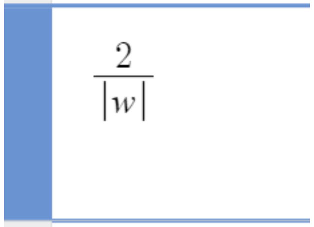
□ ラベル

5

$Y = (-p,-q+1)$

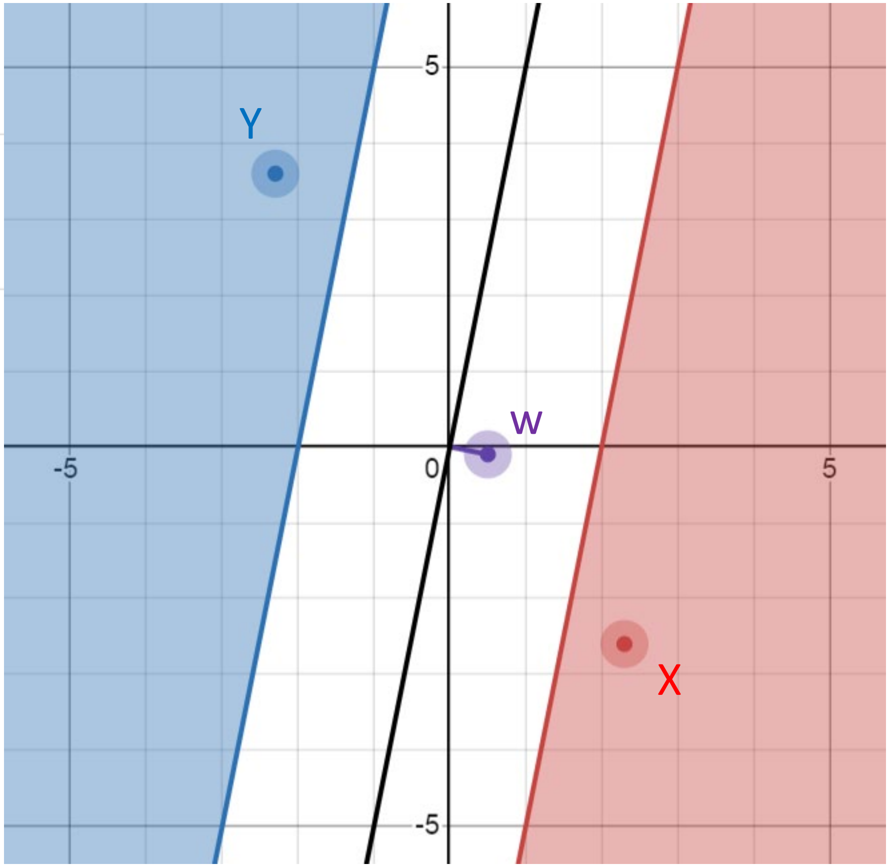
□ ラベル

Margin distance



$\frac{2}{|w|}$

Hard-margin SVC (without penalty function)



decision boundary

おまけ：様々な深層学習モデルの発展
Bonus: Development of Various Deep Learning Models

適用対象とモデル　Applicable subjects and models

画像認識　Image recognition：
　畳み込みニューラルネットワーク（CNN）
　Convolutional Neural Network

自然言語処理　Natural language processing：
　再帰型ニューラルネットワーク（RNN）、Transformer
　Recurrent Neural Network

画像生成　Image generation：
　敵対的生成ネットワーク（GAN）
　Generative Adversarial Networks

音声生成　Voice generation：
　WaveNet

モデルの予測精度向上のアルゴリズム
Algorithms for improving the model's predictive accuracy

・逆誤差伝搬法	Backpropagation
・確率的勾配降下法	Stochastic Gradient Descent, SGD
・ドロップアウト	Drop-out

非ブラックボックス化（説明可能性）Non-black box (explainability)

・SHAP

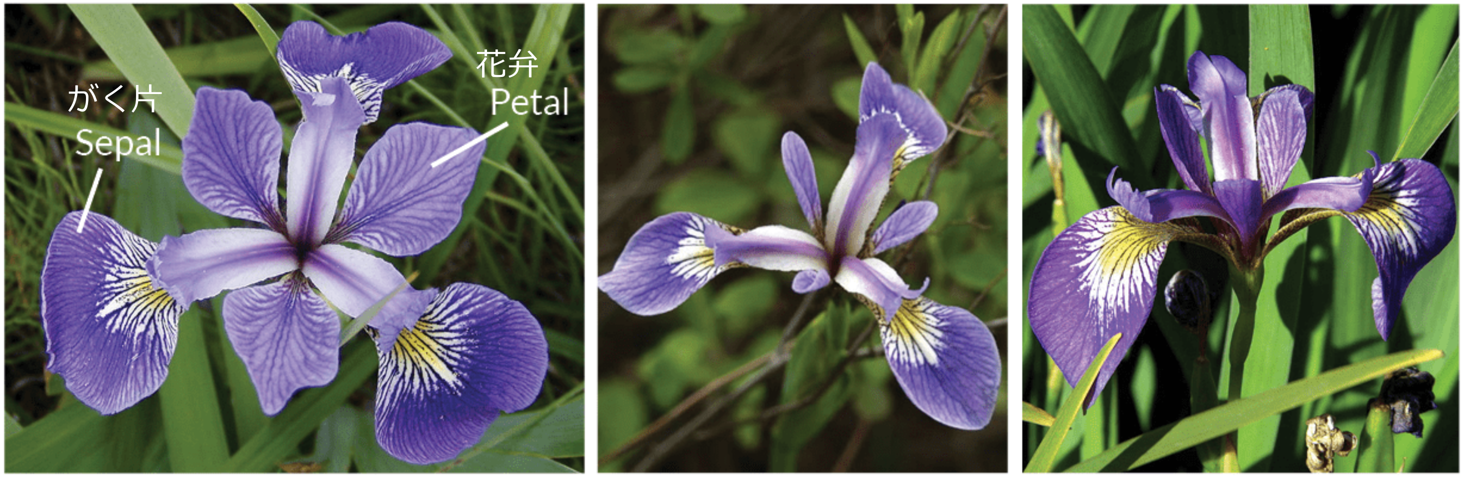
Thank you for your listening.

Email: urata.junji.gf@u.tsukuba.ac.jp

Major References:
東京大学数理・情報教育研究センター二反田篤史2021
3-3 機械学習の基礎と展望 [Link]
3-4 深層学習の基礎と展望 [Link]

IRISデータへのSVM & NNの適用
Application of SVM & Neural Neural model to IRIS Data

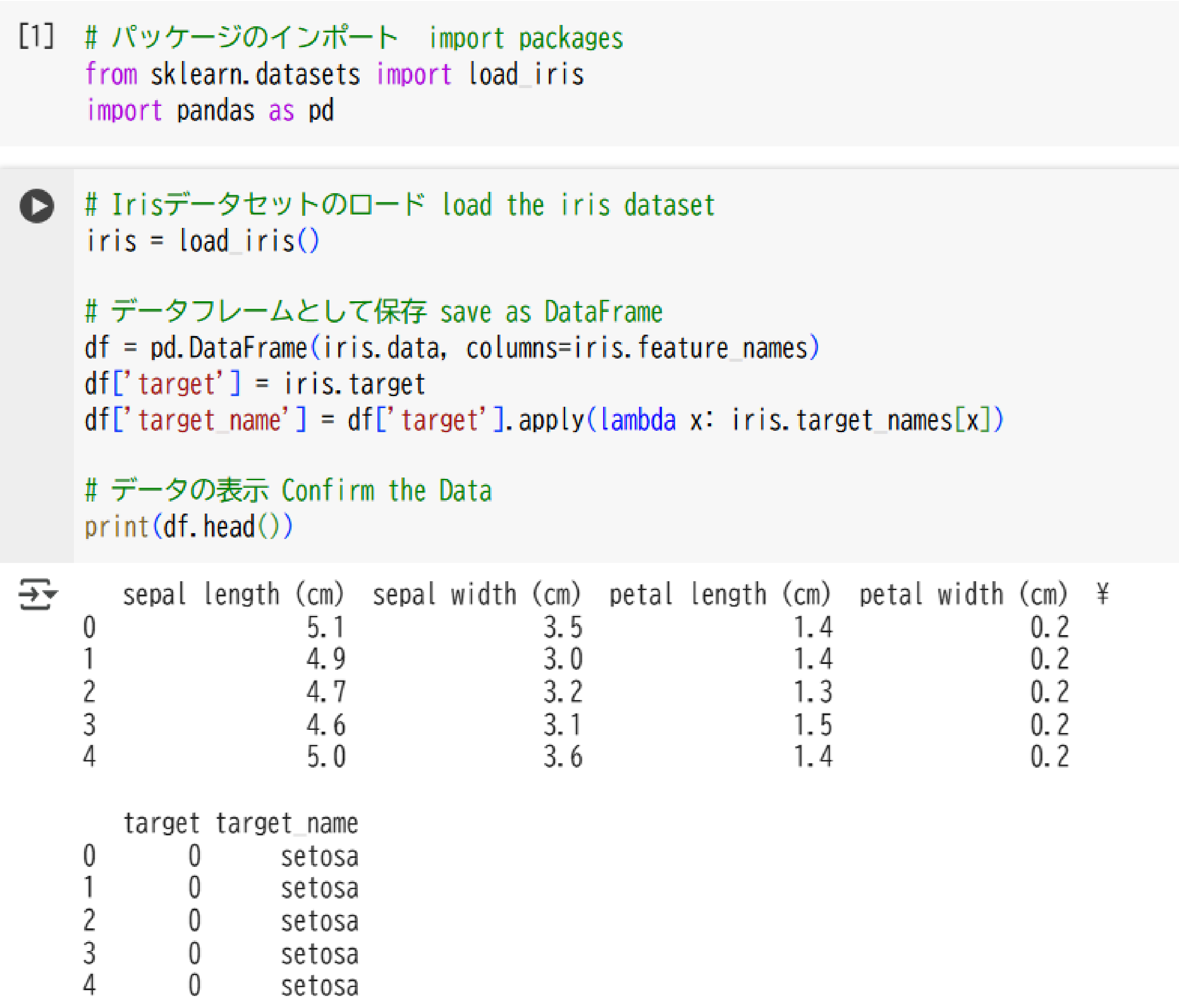
Data Science Example – Iris dataset
http://www.lac.inpe.br/~rafael.santos/Docs/CAP394/WholeStory-Iris.html



Iris Versicolor **Iris Setosa** **Iris Virginica**

Share the code in Google Colab

SVMーIRISデータへの適用①
SVM – Application to IRIS Data①

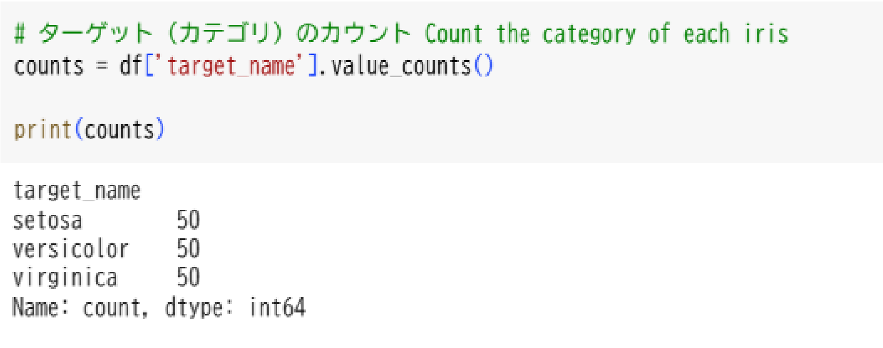


```
[1]  # パッケージのインポート  import packages
     from sklearn.datasets import load_iris
     import pandas as pd
```

```
# Irisデータセットのロード load the iris dataset
iris = load_iris()

# データフレームとして保存 save as DataFrame
df = pd.DataFrame(iris.data, columns=iris.feature_names)
df['target'] = iris.target
df['target_name'] = df['target'].apply(lambda x: iris.target_names[x])

# データの表示 Confirm the Data
print(df.head())
```

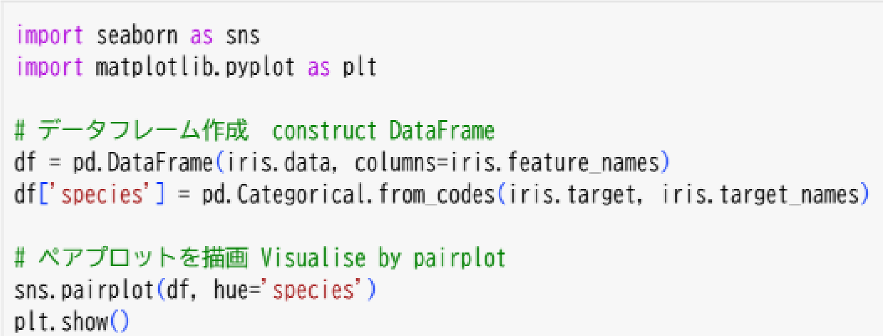
```
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  ¥
0                5.1               3.5                1.4               0.2
1                4.9               3.0                1.4               0.2
2                4.7               3.2                1.3               0.2
3                4.6               3.1                1.5               0.2
4                5.0               3.6                1.4               0.2

   target target_name
0       0      setosa
1       0      setosa
2       0      setosa
3       0      setosa
4       0      setosa
```

SVMーIRISデータへの適用②
SVM – Application to IRIS Data②

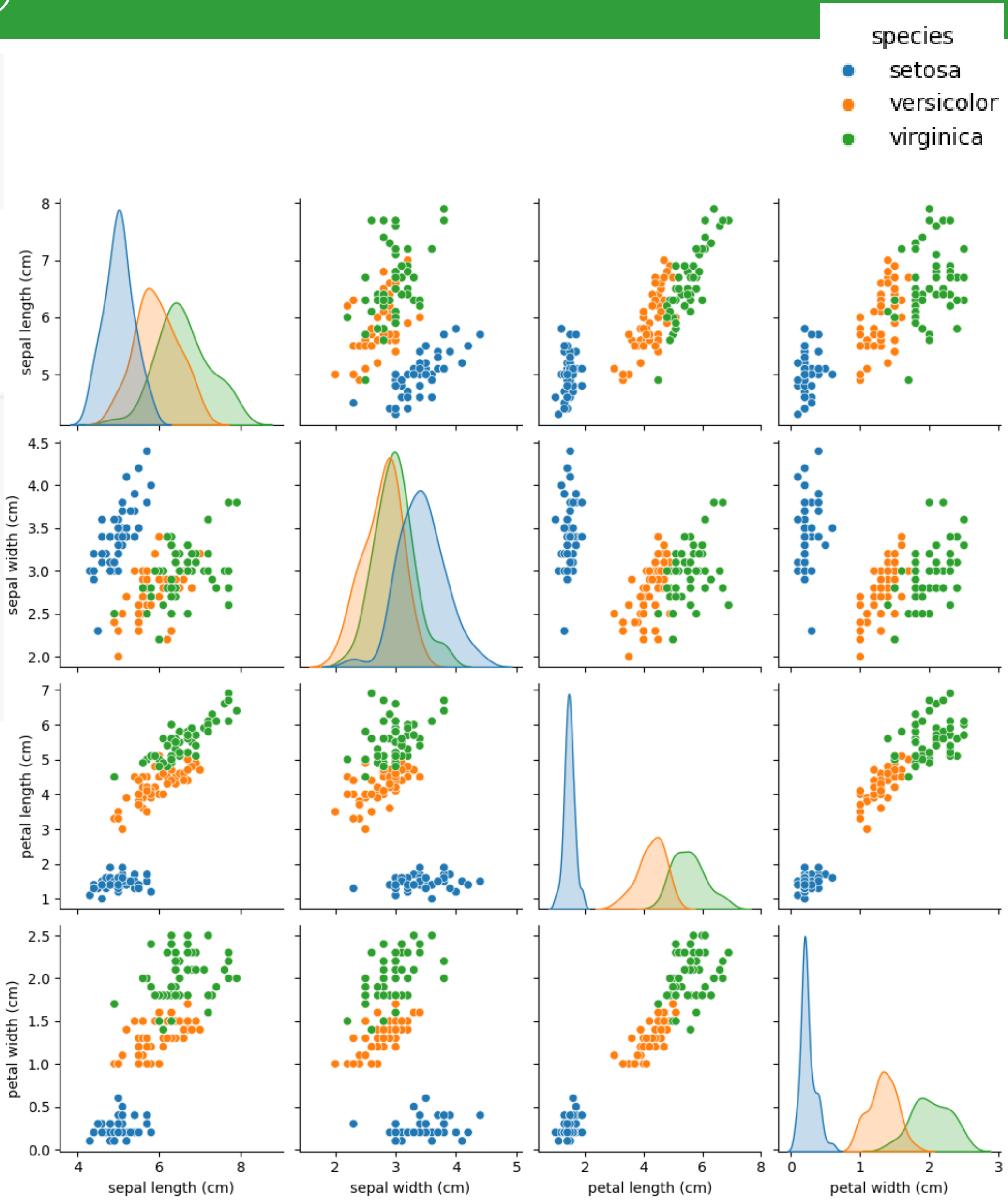


```
# ターゲット（カテゴリ）のカウント Count the category of each iris
counts = df['target_name'].value_counts()

print(counts)
```

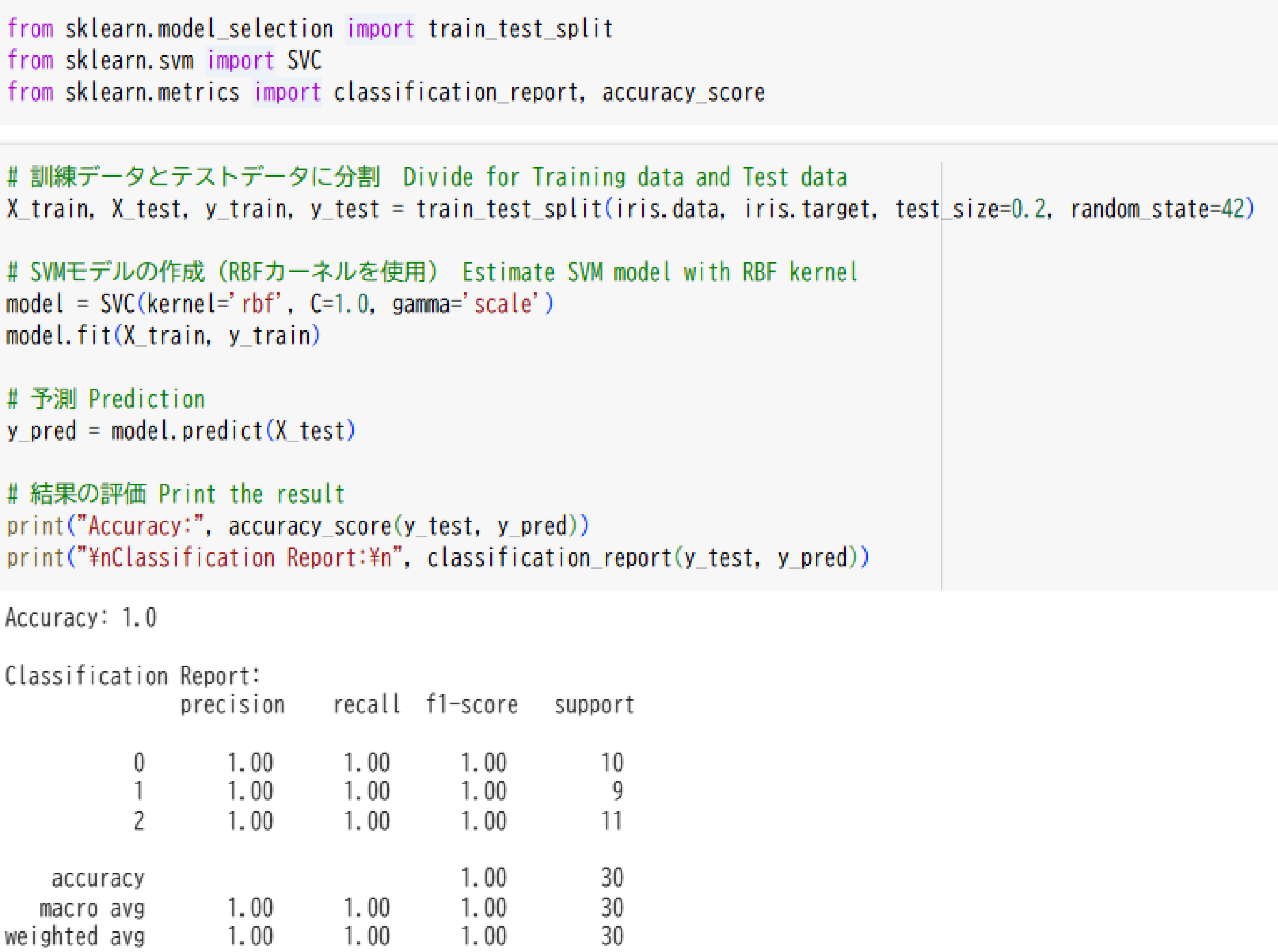
```
target_name
setosa        50
versicolor    50
virginica     50
Name: count, dtype: int64
```



```
import seaborn as sns
import matplotlib.pyplot as plt

# データフレーム作成  construct DataFrame
df = pd.DataFrame(iris.data, columns=iris.feature_names)
df['species'] = pd.Categorical.from_codes(iris.target, iris.target_names)

# ペアプロットを描画 Visualise by pairplot
sns.pairplot(df, hue='species')
plt.show()
```



SVM ーIRISデータへの適用③
SVM – Application to IRIS Data③

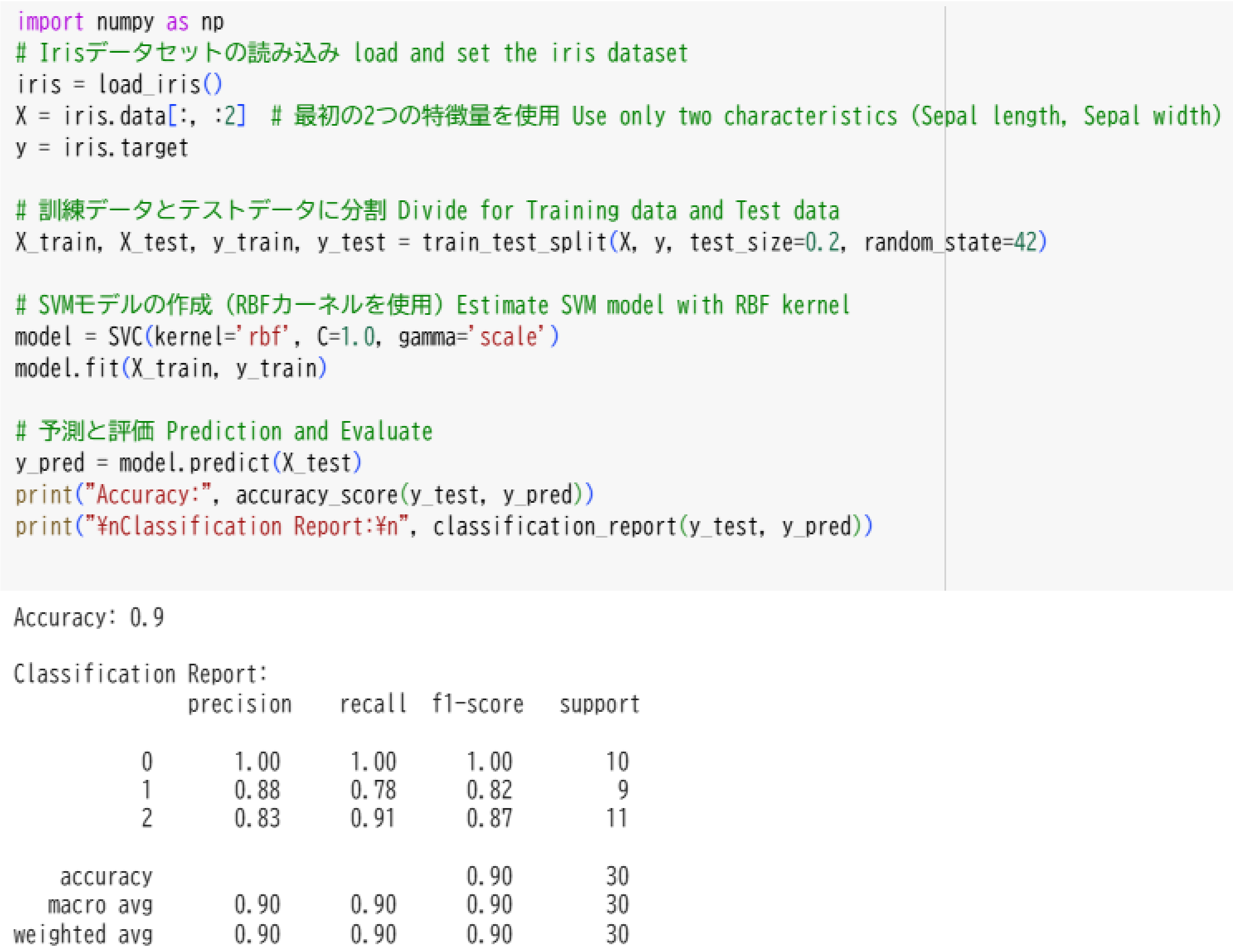


```
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import classification_report, accuracy_score
```

```
# 訓練データとテストデータに分割  Divide for Training data and Test data
X_train, X_test, y_train, y_test = train_test_split(iris.data, iris.target, test_size=0.2, random_state=42)

# SVMモデルの作成 (RBFカーネルを使用)  Estimate SVM model with RBF kernel
model = SVC(kernel='rbf', C=1.0, gamma='scale')
model.fit(X_train, y_train)

# 予測 Prediction
y_pred = model.predict(X_test)

# 結果の評価 Print the result
print("Accuracy:", accuracy_score(y_test, y_pred))
print("¥nClassification Report:¥n", classification_report(y_test, y_pred))
```

```
Accuracy: 1.0

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        10
           1       1.00      1.00      1.00         9
           2       1.00      1.00      1.00        11

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```

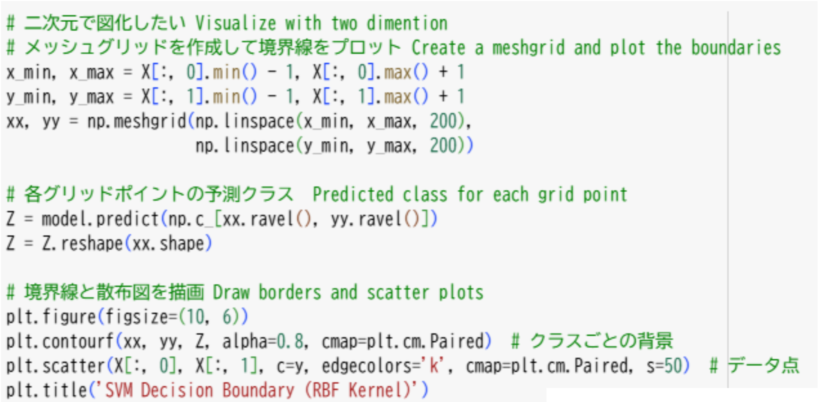
SVMーIRISデータへの適用④
SVM – Application to IRIS Data④



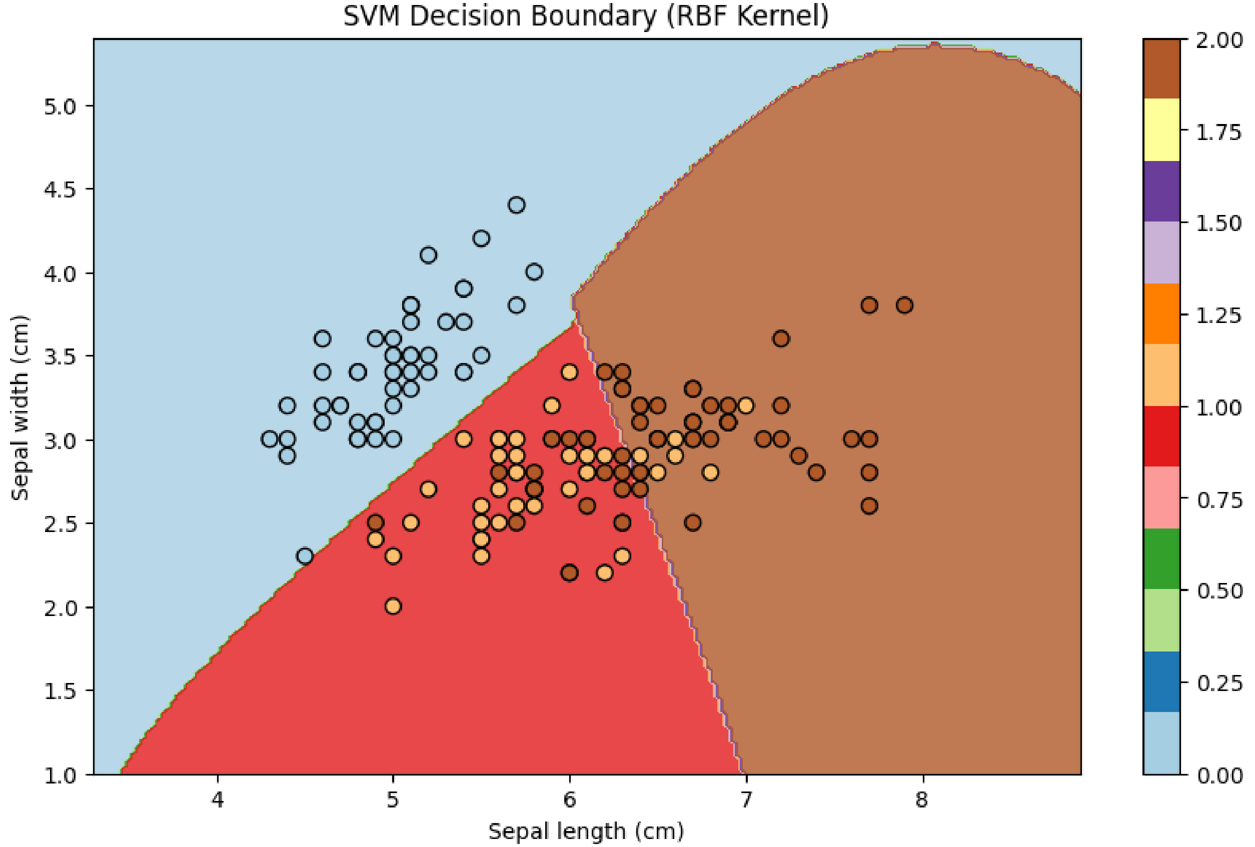
```
import numpy as np
# Irisデータセットの読み込み load and set the iris dataset
iris = load_iris()
X = iris.data[:, :2]  # 最初の2つの特徴量を使用 Use only two characteristics (Sepal length, Sepal width)
y = iris.target

# 訓練データとテストデータに分割 Divide for Training data and Test data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# SVMモデルの作成（RBFカーネルを使用）Estimate SVM model with RBF kernel
model = SVC(kernel='rbf', C=1.0, gamma='scale')
model.fit(X_train, y_train)

# 予測と評価 Prediction and Evaluate
y_pred = model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print("¥nClassification Report:¥n", classification_report(y_test, y_pred))
```

```
Accuracy: 0.9

Classification Report:
               precision    recall  f1-score   support

           0       1.00      1.00      1.00        10
           1       0.88      0.78      0.82         9
           2       0.83      0.91      0.87        11

    accuracy                           0.90        30
   macro avg       0.90      0.90      0.90        30
weighted avg       0.90      0.90      0.90        30
```

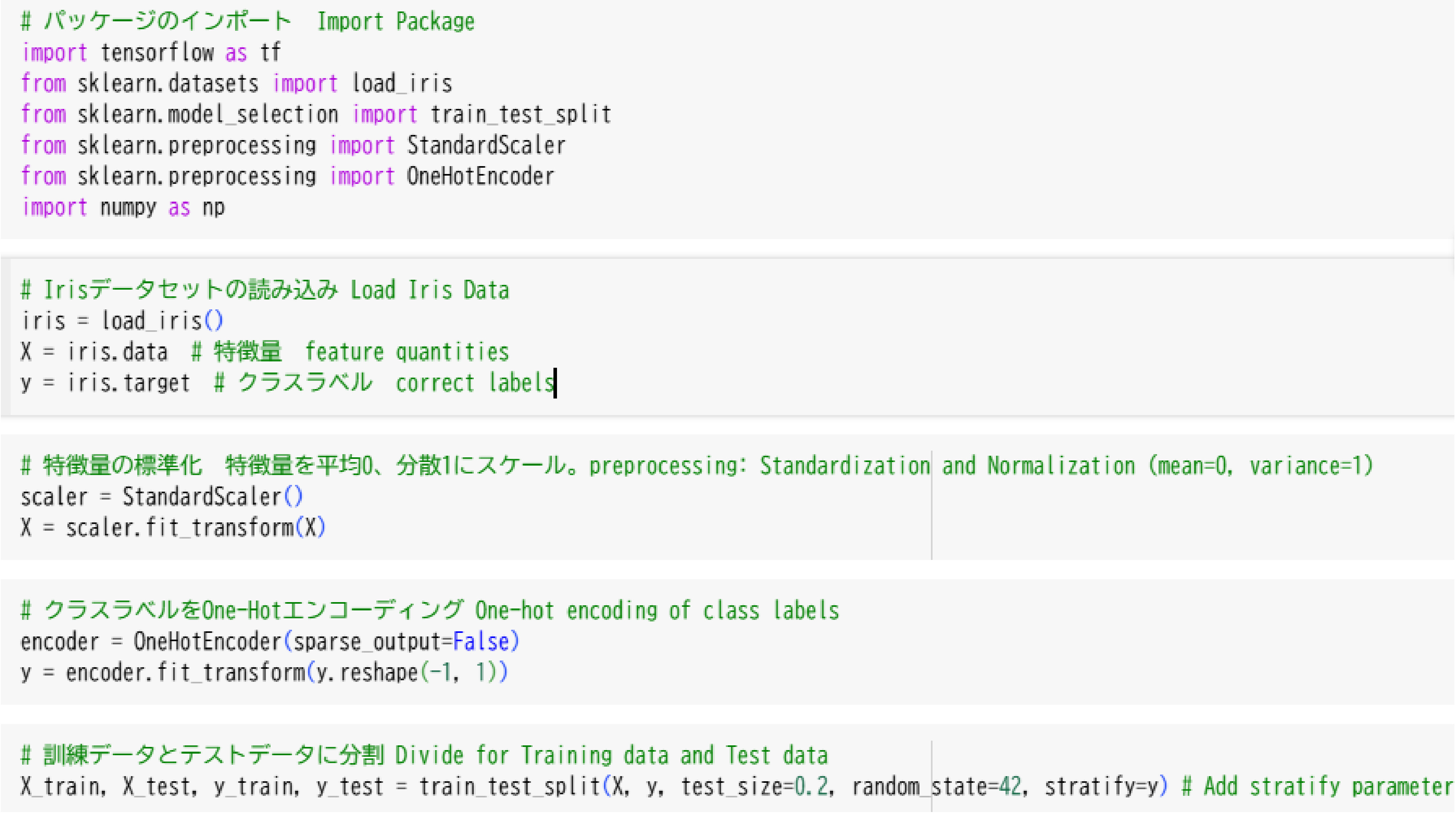
SVMーIRISデータへの適用④
SVM – Application to IRIS Data④



```
# 二次元で図化したい Visualize with two dimention
# メッシュグリッドを作成して境界線をプロット Create a meshgrid and plot the boundaries
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
xx, yy = np.meshgrid(np.linspace(x_min, x_max, 200),
                     np.linspace(y_min, y_max, 200))

# 各グリッドポイントの予測クラス  Predicted class for each grid point
Z = model.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)

# 境界線と散布図を描画 Draw borders and scatter plots
plt.figure(figsize=(10, 6))
plt.contourf(xx, yy, Z, alpha=0.8, cmap=plt.cm.Paired)  # クラスごとの背景
plt.scatter(X[:, 0], X[:, 1], c=y, edgecolors='k', cmap=plt.cm.Paired, s=50)  # データ点
plt.title('SVM Decision Boundary (RBF Kernel)')
plt.xlabel('Sepal length (cm)')
plt.ylabel('Sepal width (cm)')
plt.colorbar()
plt.show()
```



NN－IRISデータへの適用①
NN – Application to IRIS Data①



```
# パッケージのインポート  Import Package
import tensorflow as tf
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import OneHotEncoder
import numpy as np
```

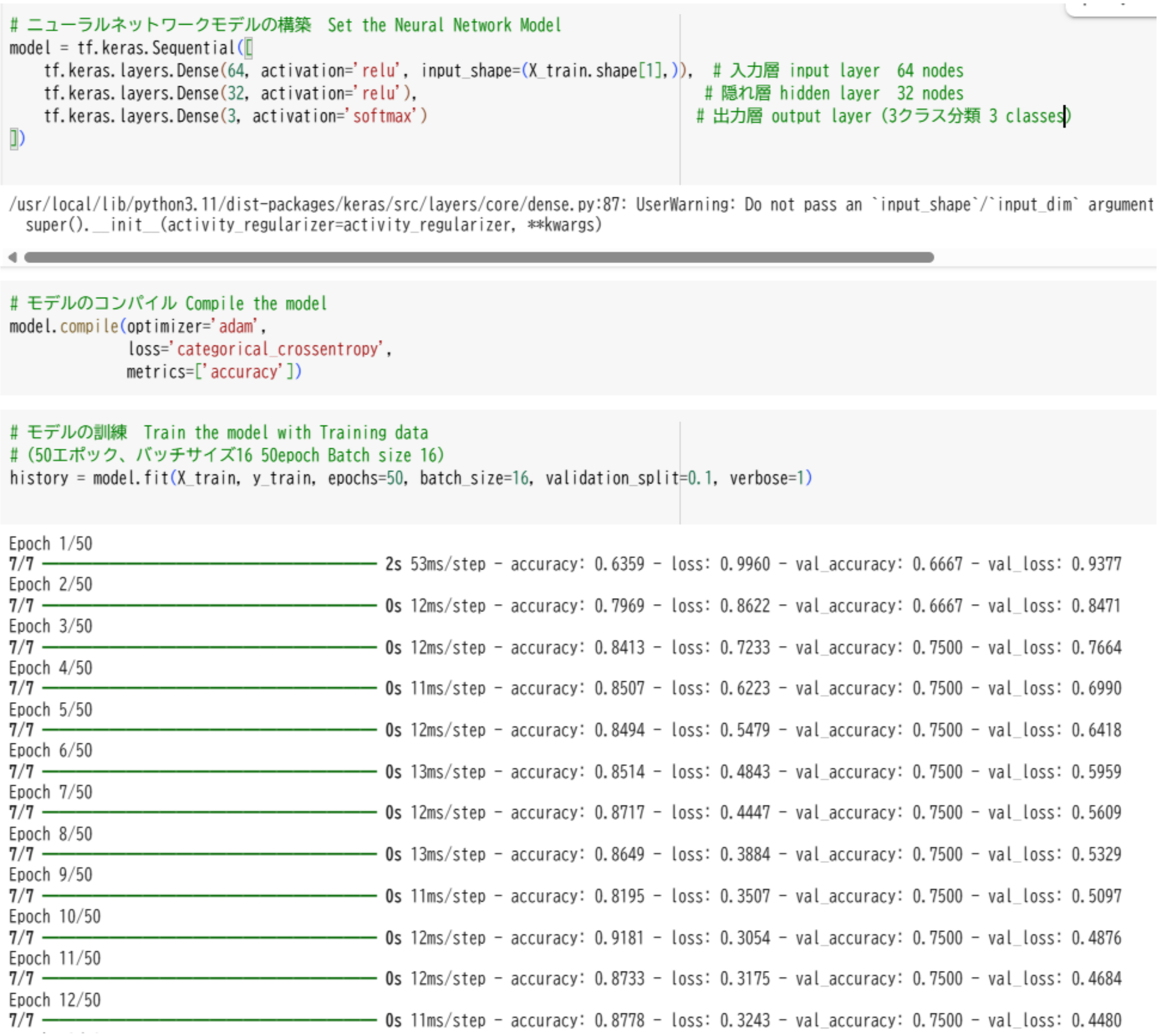
```
# Irisデータセットの読み込み Load Iris Data
iris = load_iris()
X = iris.data  # 特徴量  feature quantities
y = iris.target  # クラスラベル  correct labels
```

```
# 特徴量の標準化  特徴量を平均0、分散1にスケール。preprocessing: Standardization and Normalization (mean=0, variance=1)
scaler = StandardScaler()
X = scaler.fit_transform(X)
```

```
# クラスラベルをOne-Hotエンコーディング One-hot encoding of class labels
encoder = OneHotEncoder(sparse_output=False)
y = encoder.fit_transform(y.reshape(-1, 1))
```

```
# 訓練データとテストデータに分割 Divide for Training data and Test data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y) # Add stratify parameter
```

IRISデータへのニューラルネットワークの適用②
Application of Neural Network model to IRIS Data②



```
# ニューラルネットワークモデルの構築  Set the Neural Network Model
model = tf.keras.Sequential([
    tf.keras.layers.Dense(64, activation='relu', input_shape=(X_train.shape[1],)),  # 入力層 input layer  64 nodes
    tf.keras.layers.Dense(32, activation='relu'),                                  # 隠れ層 hidden layer  32 nodes
    tf.keras.layers.Dense(3, activation='softmax')                                # 出力層 output layer (3クラス分類 3 classes)
])
```

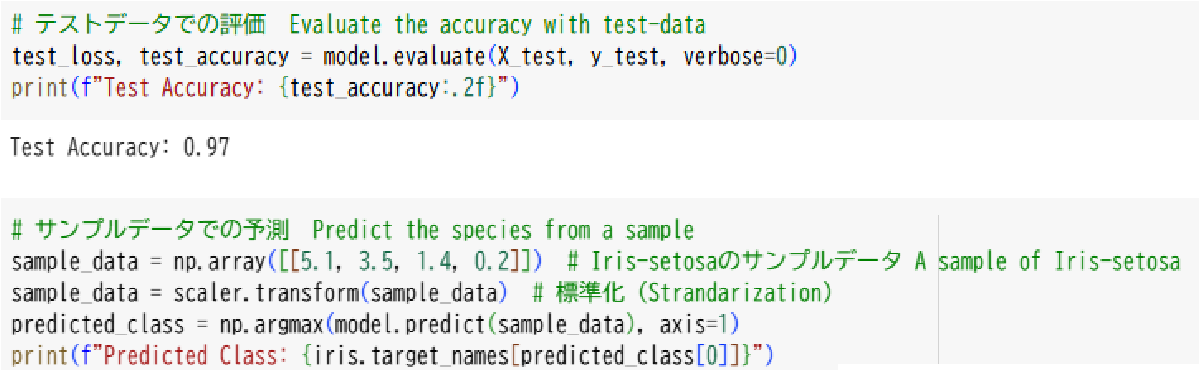
```
/usr/local/lib/python3.11/dist-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
```

```
# モデルのコンパイル Compile the model
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

```
# モデルの訓練  Train the model with Training data
# (50エポック、バッチサイズ16 50epoch Batch size 16)
history = model.fit(X_train, y_train, epochs=50, batch_size=16, validation_split=0.1, verbose=1)
```

```
Epoch 1/50
7/7 ━━━━━━━━━━━━━━━━━━━━ 2s 53ms/step - accuracy: 0.6359 - loss: 0.9960 - val_accuracy: 0.6667 - val_loss: 0.9377
Epoch 2/50
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step - accuracy: 0.7969 - loss: 0.8622 - val_accuracy: 0.6667 - val_loss: 0.8471
Epoch 3/50
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step - accuracy: 0.8413 - loss: 0.7233 - val_accuracy: 0.7500 - val_loss: 0.7664
Epoch 4/50
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 11ms/step - accuracy: 0.8507 - loss: 0.6223 - val_accuracy: 0.7500 - val_loss: 0.6990
Epoch 5/50
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step - accuracy: 0.8494 - loss: 0.5479 - val_accuracy: 0.7500 - val_loss: 0.6418
Epoch 6/50
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step - accuracy: 0.8514 - loss: 0.4843 - val_accuracy: 0.7500 - val_loss: 0.5959
Epoch 7/50
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step - accuracy: 0.8717 - loss: 0.4447 - val_accuracy: 0.7500 - val_loss: 0.5609
Epoch 8/50
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step - accuracy: 0.8649 - loss: 0.3884 - val_accuracy: 0.7500 - val_loss: 0.5329
Epoch 9/50
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 11ms/step - accuracy: 0.8195 - loss: 0.3507 - val_accuracy: 0.7500 - val_loss: 0.5097
Epoch 10/50
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step - accuracy: 0.9181 - loss: 0.3054 - val_accuracy: 0.7500 - val_loss: 0.4876
Epoch 11/50
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step - accuracy: 0.8733 - loss: 0.3175 - val_accuracy: 0.7500 - val_loss: 0.4684
Epoch 12/50
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 11ms/step - accuracy: 0.8778 - loss: 0.3243 - val_accuracy: 0.7500 - val_loss: 0.4480
```

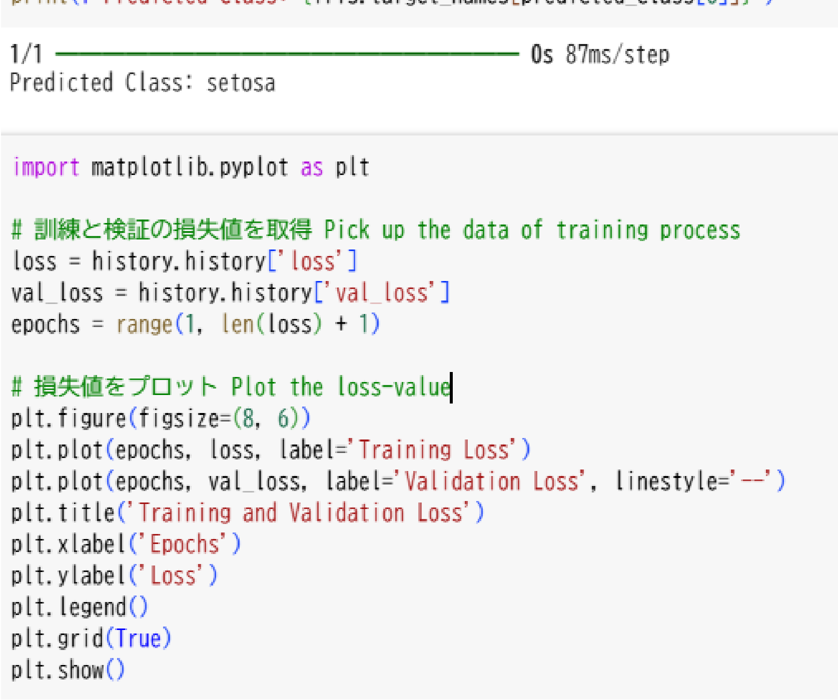
IRISデータへのニューラルネットワークの適用③
Application of Neural Network model to IRIS Data③



```
# テストデータでの評価  Evaluate the accuracy with test-data
test_loss, test_accuracy = model.evaluate(X_test, y_test, verbose=0)
print(f"Test Accuracy: {test_accuracy:.2f}")
```

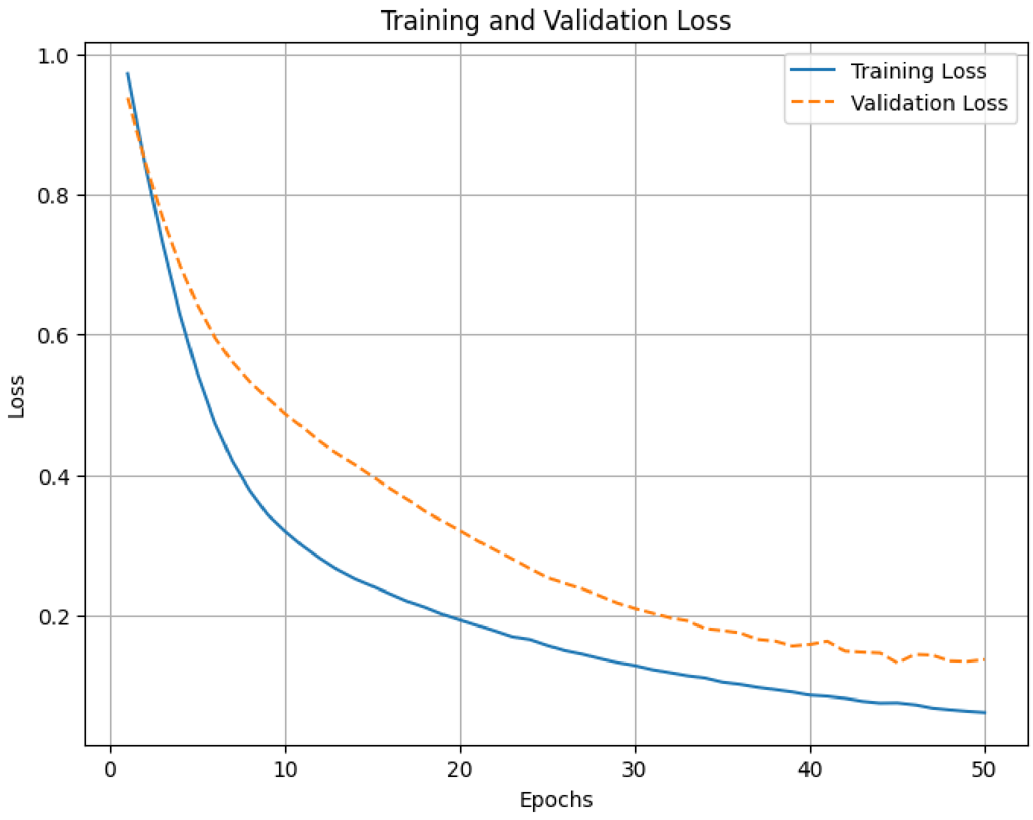
```
Test Accuracy: 0.97
```

```
# サンプルデータでの予測  Predict the species from a sample
sample_data = np.array([[5.1, 3.5, 1.4, 0.2]])  # Iris-setosaのサンプルデータ A sample of Iris-setosa
sample_data = scaler.transform(sample_data)  # 標準化 (Strandarization)
predicted_class = np.argmax(model.predict(sample_data), axis=1)
print(f"Predicted Class: {iris.target_names[predicted_class[0]]}")
```

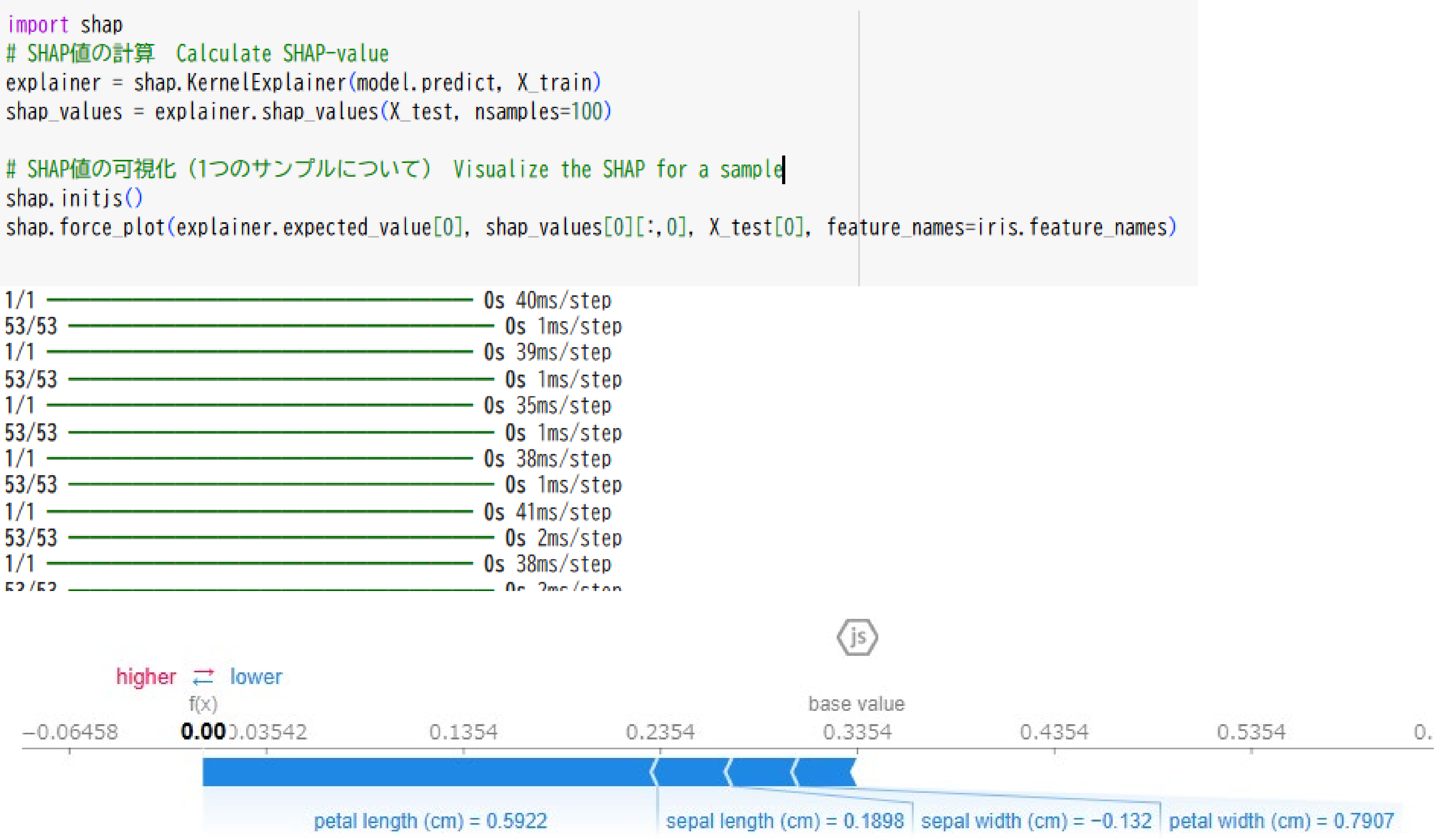


```
1/1 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 87ms/step
Predicted Class: setosa
```

```
import matplotlib.pyplot as plt

# 訓練と検証の損失値を取得 Pick up the data of training process
loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(1, len(loss) + 1)

# 損失値をプロット Plot the loss-value
plt.figure(figsize=(8, 6))
plt.plot(epochs, loss, label='Training Loss')
plt.plot(epochs, val_loss, label='Validation Loss', linestyle='--')
plt.title('Training and Validation Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.grid(True)
plt.show()
```



IRISデータへのニューラルネットワークの適用④
Application of Neural Network model to IRIS Data④



```python
import shap
# SHAP値の計算  Calculate SHAP-value
explainer = shap.KernelExplainer(model.predict, X_train)
shap_values = explainer.shap_values(X_test, nsamples=100)

# SHAP値の可視化 (1つのサンプルについて)  Visualize the SHAP for a sample
shap.initjs()
shap.force_plot(explainer.expected_value[0], shap_values[0][:,0], X_test[0], feature_names=iris.feature_names)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 40ms/step
53/53 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 39ms/step
53/53 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 35ms/step
53/53 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 38ms/step
53/53 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 1ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 41ms/step
53/53 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 38ms/step
53/53 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```

js
higher ⇄ lower
f(x)
base value
−0.06458
0.00
0.03542
0.1354
0.2354
0.3354
0.4354
0.5354
0.
petal length (cm) = 0.5922
sepal length (cm) = 0.1898
sepal width (cm) = −0.132
petal width (cm) = 0.7907