



ML Lab #2:

Breast Cancer Classification with Cross-validation

Sunglok Choi, Assistant Professor, Ph.D.
Dept. of Computer Science and Engineering, SEOULTECH
sunglok@seoultech.ac.kr | <https://mint-lab.github.io/>

Overview

- **Prerequisite**
 - Anaconda (Individual Edition)
- **Practice) Breast Cancer Classification with Cross-validation**
 - The given data
 - Expected results
 - Practice with the skeleton code
 - Step #1) Find your best classifier
- **Assignment**
 - Mission: Find your best classifier

Practice) Breast Cancer Classification

- The given data: [Breast Cancer Wisconsin \(Diagnostic\) Data Set](#)
 - Classes (#: **2**): *Malignant* (M; 악성종양 in Korean), *Benign* (B; 양성종양)
 - Attributes: **30** real numbers (except ID and target class)
 - Radius
 - Texture
 - Perimeter
 - Area
 - ...
 - The number of data: **569** (M: 212, B: 357)
 - Note) Load the dataset using scikit-learn [\[API\]](#)
`from sklearn import datasets`
`wdbc = datasets.load_breast_cancer()`

UCI Machine Learning Repository

https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29

UCI Machine Learning Repository
Center for Machine Learning and Intelligent Systems

About Citation Policy Donate a Data Set Contact

Repository Web

View ALL Data Sets

Breast Cancer Wisconsin (Diagnostic) Data Set

Download: [Data Folder](#) [Data Set Description](#)

Abstract: Diagnostic Wisconsin Breast Cancer Database



Data Set Characteristics:	Multivariate	Number of Instances:	569	Area:	Life
Attribute Characteristics:	Real	Number of Attributes:	32	Date Donated	1995-11-01
Associated Tasks:	Classification	Missing Values?	No	Number of Web Hits:	1604079

Source:

Creators:

1. Dr. William H. Wolberg, General Surgery Dept.
University of Wisconsin, Clinical Sciences Center
Madison, WI 53792
wolberg_h@eagle.surgery.wisc.edu
2. W. Nick Street, Computer Sciences Dept.
University of Wisconsin, 1210 West Dayton St., Madison, WI 53706
street_w@cs.wisc.edu 608-262-6619
3. Olvi L. Mangasarian, Computer Sciences Dept.
University of Wisconsin, 1210 West Dayton St., Madison, WI 53706
olvi_l@cs.wisc.edu

Donor:

Nick Street

Data Set Information:

Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image. A few of the images can be found at [\[Web Link\]](#)

Separating plane described above was obtained using Multisurface Method-Tree (MSM-T) [K. P. Bennett, "Decision Tree Construction Via Linear Programming," Proceedings of the 4th Midwest Artificial Intelligence and Cognitive Science Society, pp. 97-101, 1992], a classification method which uses linear programming to construct a decision tree. Relevant features were selected using an exhaustive search in the space of 1-4 features and 1-3 separating planes.

The actual linear program used to obtain the separating plane in the 3-dimensional space is that described in: [K. P. Bennett and O. L. Mangasarian: "Robust Linear Programming Discrimination of Two Linearly Inseparable Sets", Optimization Methods and Software 1, 1992, 23-34].

This database is also available through the UW CS ftp server:
ftp ftp.cs.wisc.edu
cd math-prog/cpo-dataset/machine-learn/WDBC/

Practice) Breast Cancer Classification

- Expected results
 - The default classifier: Decision tree (`tree.DecisionTreeClassifier`)
 - Accuracy @ training data: 1.000
 - Accuracy @ test data: 0.919
 - Your score: 12
- Evaluation (Total score: 20)
 - Your score = $10 + 100 \times (\text{your accuracy @ test data} - 0.9)$

Practice) Breast Cancer Classification

- The given skeleton code (wdbc_classification_cv.py)
 - Step #1) Find your best classifier

```
import numpy as np
from sklearn import (datasets, tree, model_selection)

if __name__ == '__main__':
    # Load a dataset
    wdbc = datasets.load_breast_cancer()

    # Train a model
    model = tree.DecisionTreeClassifier() # TODO
    cv_results = model_selection.cross_validate(model, wdbc.data, wdbc.target, cv=5, return_train_score=True)

    # Evaluate the model
    acc_train = np.mean(cv_results['train_score'])
    acc_test = np.mean(cv_results['test_score'])
    print(f'* Accuracy @ training data: {acc_train:.3f}')
    print(f'* Accuracy @ test data: {acc_test:.3f}')
    print(f'* Your score: {max(10 + 100 * (acc_test - 0.9), 0):.0f}')
```

Assignment

- Mission
 - Find your best classifier using the skeleton code (`wdbc_classification_cv.py`)
 - Note) Please consider a situation where training your model takes 1 hour, 1 day, 1 week, or even 1 month.
 - Submit your code (`wdbc_classification_cv.py`) and its accuracy (`wdbc_classification_cv.png`)
- Condition
 - Please follow the above filename convention.
 - You **can** start from scratch (without using the given skeleton code).
 - However, you **should** use the given data.
 - You **can** freely change the given skeleton code if necessary.
- Submission
 - Deadline: **November 12, 2025 23:59** (**firm deadline**; no extension)
 - Where: e-Class > Assignments
 - Score: Max 20 points