<과제물 작성시 주의사항>

[공통]

과제물 제출시 완성된 **소스파일 및 보고서**를 반드시 **'HW_01_학번.zip'** 형식으로 압축하여 첨부합니다.

(iris.py, main_app.py, 이름 약어.py, HW_01_학번.pdf)

[소스파일]

1. 소스파일은 **.py파일만 작성**하며 반드시 문제에서 지시 또는 요구한 조건에 맞추어서 작성합니다.

(jupyter로 작성하였어도 코드를 제출 시 py파일로 작성하여 제출하여야 합니다.)

2. 각 코드마다 **반드시 주석을 달아 주셔야 합니다.** 주석을 달지 않을 경우, 부분적으로 감점이 있을 수 있습니다.
3. 결과가 올바르더라도 과정이 옳지 않을 경우, 부분적으로 감점이 있을 수 있습니다.
4. 제출한 파일이 실행되지 않을 경우, 제출한 과제물은 0점 처리됩니다.

[보고서]

1. **PDF**로 제출하며, 표지를 포함해야 합니다.
2. 보고서에는 **(과제 제목 및 목적), (소스 코드에 대한 설명), (실행 결과), (참고문헌)**이 포함되어야 합니다.
3. 자신의 코드에 대한 설명이 명확하지 않거나 copy한 글이라면 0점 처리됩니다.
4. **실행 결과는 실행 결과를 캡처하여 첨부하도록 합니다.**
5. 참고문헌은 반드시 적어도 한 개 이상을 명시하여야 합니다.

NAIVE BAYESIAN CLASSIFIER DESIGN

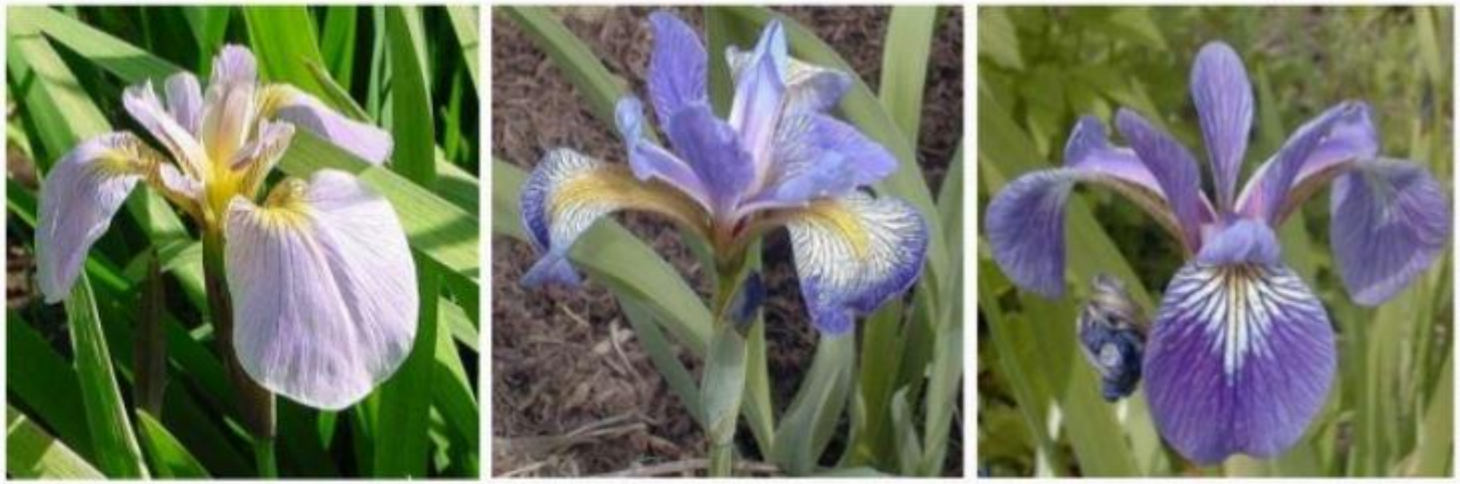
Machine learning homework-1

Assistant : Choongseop Lee, cndtjq97@gmail.com

INDEX

INTRODUCTION

- Build naïve Bayesian classifier which can classify some flowers



Iris setosa Iris versicolor Iris virginica

- Given 4 feature
  - Petal width
  - Petal length
  - Sepal width
  - Sepal length

INTRODUCTION

- You should build 6 python functions at util.py
  - feature_normalization (10 points) - **1**
  - get_normal_parameter (20 points) - **2**
  - get_prior_probability (10 points) - **3**
  - Gaussian_PDF (10 points) - **4**
  - Gaussian_Log_PDF (10 points) - **5**
  - Gaussian_NB (40 points) - **6**

- You can classify the flowers using main_app.py
- Submission : change below files to zip file and submit it by KLAS
  - iris.csv
  - main_app.py
  - Name.py (please change util.py to your name.py)
    - E.g. if your name is 홍길동 --> (util.py --> GDH.py)
  - HW_01_student ID.pdf (E.g. HW_01_202110605.pdf) <= report

STRUCTURE OF CLASSIFIER

Feature normalization → Spilt test data and train data → Get parameter and prior probability → Classify the test data using posterior probability

FEATURE NORMALIZATION

- Normalization data $\bar{x} = \frac{x - mu}{\sigma}$ - **1**
  - You can use below function or any built-in function
  - Numpy.sum
  - Numpy.mean
  - Numpy.std

PARAMETER ESTIMATION

- Calculate the mean and standard deviation of train data each for labels and features - **2**
- Calculate the prior probability - **3**
- the example of mean matrix of train data

	Feature1	Feature2	Feature3	Feature4
Label1	Mean	Mean	Mean	Mean
Label2	Mean	Mean	Mean	Mean
Label3	Mean	Mean	Mean	mean

- You can use below function or any built-in function
  - Numpy.mean
  - Numpy.std
  - Numpy.where
  - List comprehension (python syntax)

ESTIMATION OF PROBABILITY DISTRIBUTION

- Use Naive Bayesian theorem
  - $p(C_k|x) = \frac{p(C_k)p(x|C_k)}{p(x)}$ = **Posterior**
  - $p(x|C_k)$ = k_{th} - feature (observation) = **Likelihood**
  - $p(x)$ = normalization factor ($\sum_{k=1}^{class\_num} p(C_k)p(x|C_k)$) = Evidence (Never mind)
  - $p(C_k)$ = initial probability of class = **prior**

- Calculate **Likelihood** using Gaussian PDF or Gaussian Log PDF function based on parameters – **4, 5**

ESTIMATION OF PROBABILITY DISTRIBUTION

- Estimate the probability (**posterior**) of feature vector each for classes - **6**
- Use Naive Bayesian theorem
  - $p(C_k|x) = \frac{p(C_k)p(x|C_k)}{p(x)}$ = **Posterior**

- Hints
- Use chain rule
  - $p(C_k)p(x_{1\ and}\ x_{2\ and}\ x_3|C_k) = p(C_k)p(x_1|C_k)p(x_2|C_k)p(x_3|C_k)$
- Use log scale
  - $\ln(p(C_k)p(x_1|C_k)p(x_2|C_k)p(x_3|C_k)) = \ln(p(C_k)) + \ln(p(x_1|C_k)) + \ln(p(x_2|C_k)) + \ln(p(x_3|C_k))$

ESTIMATION OF PROBABILITY DISTRIBUTION

- E.g)

	Probability of class1	Probability of class2	Probability of class3
Feature vector1	$\ln(p(C_1)p(x1\|C_1)p(x2\|C_1)p(x3\|C_1)p(x4\|C_1))$	$\ln(p(C_2)p(x1\|C_2)p(x2\|C_2)p(x3\|C_2)p(x4\|C_2))$	$\ln(p(C_3)p(x1\|C_3)p(x2\|C_3)p(x3\|C_3)p(x4\|C_3))$
Feature vector2	$\ln(p(C_1)p(x1\|C_1)p(x2\|C_1)p(x3\|C_1)p(x4\|C_1))$	$\ln(p(C_2)p(x1\|C_2)p(x2\|C_2)p(x3\|C_2)p(x4\|C_2))$	$\ln(p(C_3)p(x1\|C_3)p(x2\|C_3)p(x3\|C_3)p(x4\|C_3))$
Feature vector3	$\ln(p(C_1)p(x1\|C_1)p(x2\|C_1)p(x3\|C_1)p(x4\|C_1))$	$\ln(p(C_2)p(x1\|C_2)p(x2\|C_2)p(x3\|C_2)p(x4\|C_2))$	$\ln(p(C_3)p(x1\|C_3)p(x2\|C_3)p(x3\|C_3)p(x4\|C_3))$
Feature vector4	$\ln(p(C_1)p(x1\|C_1)p(x2\|C_1)p(x3\|C_1)p(x4\|C_1))$	$\ln(p(C_2)p(x1\|C_2)p(x2\|C_2)p(x3\|C_2)p(x4\|C_2))$	$\ln(p(C_3)p(x1\|C_3)p(x2\|C_3)p(x3\|C_3)p(x4\|C_3))$

.
.
.

- You can use below function or any built-in function
  - Numpy library : where, exp, log
  - Python library : len

OTHER FUNCTIONS

- def split_data, classifier, and accuracy are just utility functions
- However, you should report how that functions work

	Probability of class1	Probability of class2	Probability of class3
Feature vector1	$\ln(p(C_1)p(x_1\|C_1)p(x_2\|C_1)p(x_3\|C_1)p(x_4\|C_1))$	$\ln(p(C_2)p(x_1\|C_2)p(x_2\|C_2)p(x_3\|C_2)p(x_4\|C_2))$	$\ln(p(C_3)p(x_1\|C_3)p(x_2\|C_3)p(x_3\|C_3)p(x4\|C_3))$
Feature vector2	$\ln(p(C_1)p(x_1\|C_1)p(x_2\|C_1)p(x_3\|C_1)p(x_4\|C_1))$	$\ln(p(C_2)p(x_1\|C_2)p(x_2\|C_2)p(x_3\|C_2)p(x_4\|C_2))$	$\ln(p(C_3)p(x_1\|C_3)p(x_2\|C_3)p(x_3\|C_3)p(x4\|C_3))$
Feature vector3	$\ln(p(C_1)p(x_1\|C_1)p(x_2\|C_1)p(x_3\|C_1)p(x_4\|C_1))$	$\ln(p(C_2)p(x_1\|C_2)p(x_2\|C_2)p(x_3\|C_2)p(x_4\|C_2))$	$\ln(p(C_3)p(x_1\|C_3)p(x_2\|C_3)p(x_3\|C_3)p(x4\|C_3))$
Feature vector4	$\ln(p(C_1)p(x_1\|C_1)p(x_2\|C_1)p(x_3\|C_1)p(x_4\|C_1))$	$\ln(p(C_2)p(x_1\|C_2)p(x_2\|C_2)p(x_3\|C_2)p(x_4\|C_2))$	$\ln(p(C_3)p(x_1\|C_3)p(x_2\|C_3)p(x_3\|C_3)p(x4\|C_3))$

.
.
.

	Estimation class
Feature vector1	1
Feature vector2	2
Feature vector3	3
Feature vector4	1

.
.
.

RESULT

- After build all of function, you can see below result from python console when you compile “main_app.py” (or yielded 90% accuracy due to shuffling data)

```
In [21]: runfile('D:/3학기/머신러닝_조교/머신러닝과제1/답안지/main_app.py', wdir='D:/3학기/머신러닝_조교/머신러닝과제1/답안지')
Reloaded modules: utills
accuracy is 97.95918367346938% ! !
the number of correct data is 48 of 49 ! !

In [22]:
```

- Print out the results at least 10 times and write them in the report

REFERENCE

- https://en.wikipedia.org/wiki/Naive_Bayes_classifier
- https://en.wikipedia.org/wiki/Standard_score