東莞理工學院
DONGGUAN UNIVERSITY OF TECHNOLOGY

人工智能概论

实验二：分类模型

丁烨，计算机科学与技术学院
dingye@dgut.edu.cn

实验环境

https://websitesetup.org/wp-content/uploads/2020/04/Python-Cheat-Sheet.pdf
https://perso.limsi.fr/pointal/_media/python:cours:mementopython3-english.pdf

实验环境

- scikit-learn
- https://scikit-learn.org/

- 一个开源的科学计算及机器学习工具包
- 属于 SciPy 项目的一部分
- 包含了常见的、基础的机器学习算法
- 不支持深度学习
- 较难支持 GPU 加速

实验环境

- NumPy
- https://www.numpy.org/

- 针对数组运算提供大量的数学函数库
- 支持大规模的多维数组与矩阵运算
- NumPy 是 SciPy、Matplotlib 等扩展程序库的基础组件

实验环境

- 使用 pip 安装 scikit-learn：
- `pip3 install --user -U scikit-learn`

- 如果安装不成功，可尝试使用 apt 安装：
- `sudo apt install python3-sklearn`

数据集

- UCI 葡萄酒质量数据集
- https://archive.ics.uci.edu/dataset/186/wine+quality

Data Set Characteristics:	Multivariate	**Number of Instances:**	4898	**Area:**	Business
Attribute Characteristics:	Real	**Number of Attributes:**	12	**Date Donated**	2009-10-07
Associated Tasks:	Classification, Regression	**Missing Values?**	N/A	**Number of Web Hits:**	1992906

- 两套（红葡萄酒、白葡萄酒）的物理化学性质数据，已量化
- 这些葡萄酒对应的品质等级

数据集

```python
import numpy
from sklearn.linear_model import LinearRegression

data = numpy.genfromtxt('winequality-white.csv', delimiter=';')
data = numpy.delete(data, 0, axis=0)
print(data)
X = data[:-100, :-1]
y = data[:-100, -1]
m = LinearRegression().fit(X, y)
r = m.predict(data[100:, :-1])
print(r)
g = data[100:, -1]
print(g)
print(((g - r) ** 2).mean(axis=0))
```

```
[[ 7.    0.27  0.36 ...  0.45  8.8   6.  ]
 [ 6.3   0.3   0.34 ...  0.49  9.5   6.  ]
 [ 8.1   0.28  0.4  ...  0.44 10.1   6.  ]
 ...
 [ 6.5   0.24  0.19 ...  0.46  9.4   6.  ]
 [ 5.5   0.29  0.3  ...  0.38 12.8   7.  ]
 [ 6.    0.21  0.38 ...  0.32 11.8   6.  ]]
[5.63144651 6.10115806 5.85674592 ... 5.34351476 6.57567039
6.35192467]
[5. 5. 5. ... 6. 7. 6.]
0.5630446931574506
```

数据集

```python
import numpy
from sklearn.linear_model import LinearRegression

data = numpy.genfromtxt('winequality-white.csv', delimiter=';')
data = numpy.delete(data, 0, axis=0)
print(data)
X = data[:-100, :-1]
y = data[:-100, -1]
m = LinearRegression().fit(X, y)
r = m.predict(data[100:, :-1])
print(r)
g = data[100:, -1]
print(g)
print(((g - r) ** 2).mean(axis=0))
```

```
[[ 7.    0.27  0.36 ...  0.45  8.8   6.  ]
 [ 6.3   0.3   0.34 ...  0.49  9.5   6.  ]
 [ 8.1   0.28  0.4  ...  0.44 10.1   6.  ]
 ...
 [ 6.5   0.24  0.19 ...  0.46  9.4   6.  ]
 [ 5.5   0.29  0.3  ...  0.38 12.8   7.  ]
 [ 6.    0.21  0.38 ...  0.32 11.8   6.  ]]
[5.63144651 6.10115806 5.85674592 ... 5.34351476 6.57567039
6.35192467]
[5. 5. 5. ... 6. 7. 6.]
0.5630446931574506
```

数据集

```
import numpy
from sklearn.linear_model import LinearRegression

data = numpy.genfromtxt('winequality-white.csv', delimiter=';')
data = numpy.delete(data, 0, axis=0)
print(data)
X = data[:-100, :-1]
y = data[:-100, -1]
m = LinearRegression().fit(X, y)
r = m.predict(data[100:, :-1])
print(r)
g = data[100:, -1]
print(g)
print(((g - r) ** 2).mean(axis=0))
```

```
[[ 7.    0.27  0.36 ...  0.45  8.8   6.  ]
 [ 6.3   0.3   0.34 ...  0.49  9.5   6.  ]
 [ 8.1   0.28  0.4  ...  0.44 10.1   6.  ]
 ...
 [ 6.5   0.24  0.19 ...  0.46  9.4   6.  ]
 [ 5.5   0.29  0.3  ...  0.38 12.8   7.  ]
 [ 6.    0.21  0.38 ...  0.32 11.8   6.  ]]

[5.63144651 6.10115806 5.85674592 ... 5.34351476 6.57567039
6.35192467]

[5. 5. 5. ... 6. 7. 6.]

0.5630446931574506
```

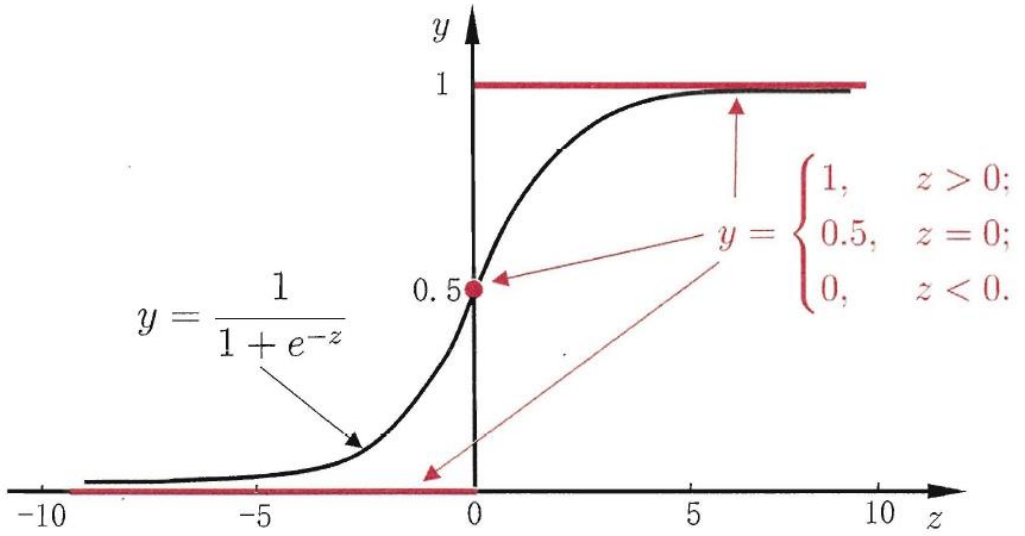
回归与分类

- 回归（Regression）
- 预测模型输出结果为连续的值

- 分类（Classification）
- 预测模型输出结果为离散的值

对数几率回归

- 对数几率回归（Logistic / Logit Regression）
- 使用对数几率函数（Logistic Function）
- 将分类任务的真相与线性回归模型的预测值联系起来
- 本质上来说是用线性回归模型的预测结果去逼近真相的对数几率



对数几率回归

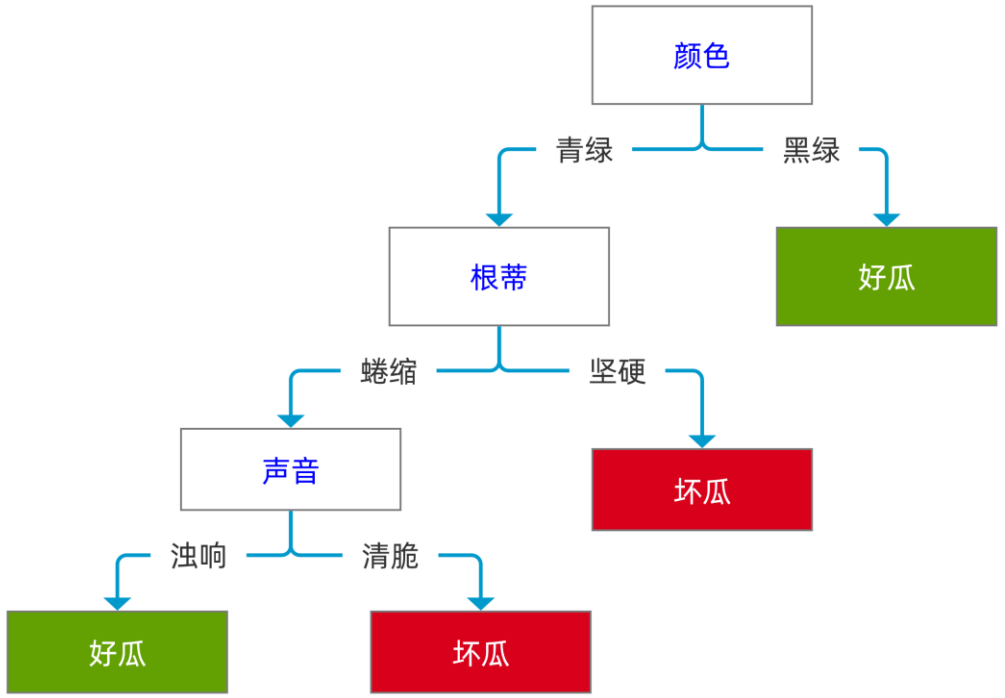
```python
import numpy
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import precision_score

data = numpy.genfromtxt('winequality-white.csv', delimiter=';')
data = numpy.delete(data, 0, axis=0)
print(data)
X = data[:-100, :-1]
y = data[:-100, -1]
m = LogisticRegression().fit(X, y)
r = m.predict(data[100:, :-1])
print(r)
g = data[100:, -1]
print(g)
print(precision_score(g, r, average='micro'))
```

```
[[ 7.    0.27  0.36 ...  0.45  8.8   6.  ]
 [ 6.3   0.3   0.34 ...  0.49  9.5   6.  ]
 [ 8.1   0.28  0.4  ...  0.44 10.1   6.  ]
 ...
 [ 6.5   0.24  0.19 ...  0.46  9.4   6.  ]
 [ 5.5   0.29  0.3  ...  0.38 12.8   7.  ]
 [ 6.    0.21  0.38 ...  0.32 11.8   6.  ]]
[5. 6. 6. ... 6. 6. 6.]
[5. 5. 5. ... 6. 7. 6.]
0.47061275531471447
```

决策树

- 决策树（Decision Tree）
- 又称判定树
- 是一个基于树结构进行决策的常见机器学习方法
- 其基本流程遵循简单且直观的策略：
- “分而治之”



决策树

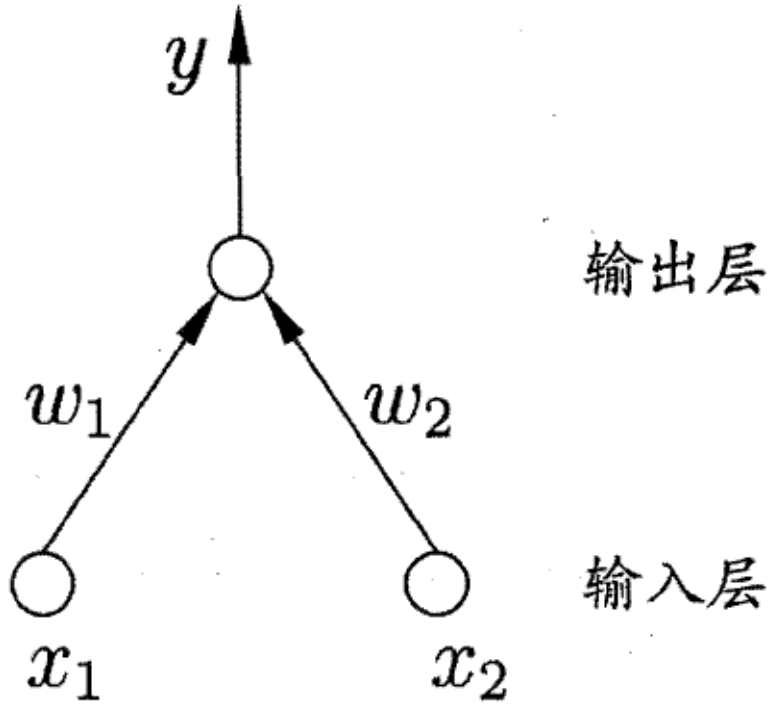
```
import numpy
from sklearn.metrics import precision_score
from sklearn.tree import DecisionTreeClassifier

data = numpy.genfromtxt('winequality-white.csv', delimiter=';')
data = numpy.delete(data, 0, axis=0)
print(data)
X = data[:-100, :-1]
y = data[:-100, -1]
m = DecisionTreeClassifier().fit(X, y)
r = m.predict(data[100:, :-1])
print(r)
g = data[100:, -1]
print(g)
print(precision_score(g, r, average='micro'))
```

```
[[ 7.    0.27  0.36 ...  0.45  8.8   6.  ]
 [ 6.3   0.3   0.34 ...  0.49  9.5   6.  ]
 [ 8.1   0.28  0.4  ...  0.44 10.1   6.  ]
 ...
 [ 6.5   0.24  0.19 ...  0.46  9.4   6.  ]
 [ 5.5   0.29  0.3  ...  0.38 12.8   7.  ]
 [ 6.    0.21  0.38 ...  0.32 11.8   6.  ]]
[5. 5. 5. ... 5. 7. 6.]
[5. 5. 5. ... 6. 7. 6.]
0.9893705710712797
```

感知机

- 感知机（Perceptron）
- 由两层神经元组成
- 输入层接收外界输入信号后传递给输出层
- 输出层是 M-P 神经元



感知机

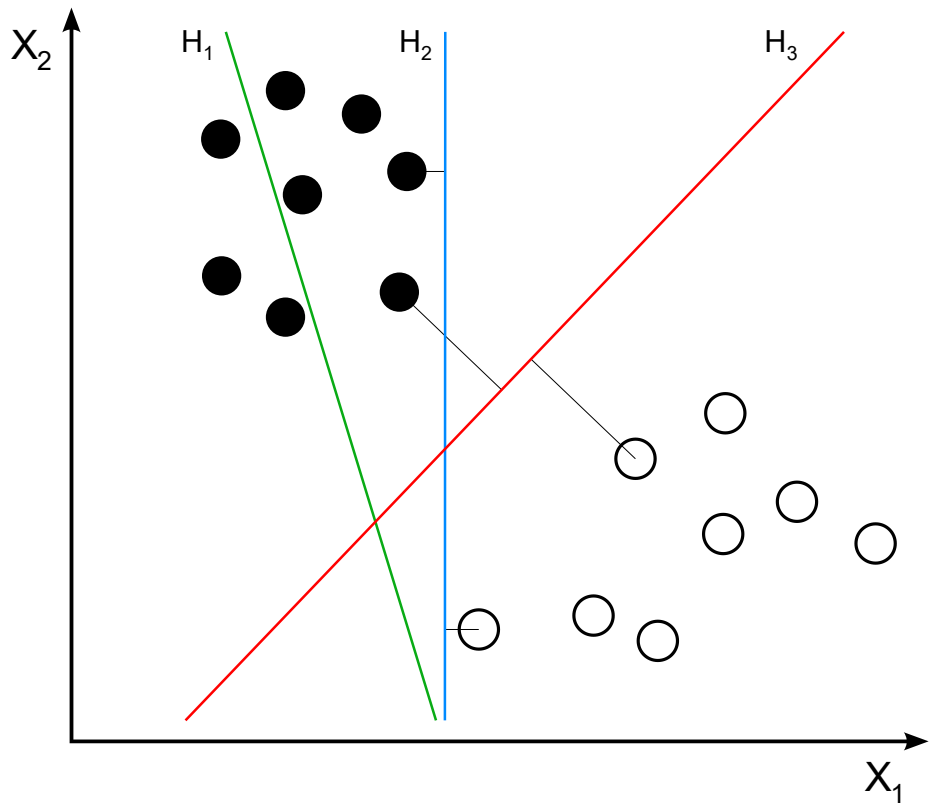
```python
import numpy
from sklearn.linear_model import Perceptron
from sklearn.metrics import precision_score

data = numpy.genfromtxt('winequality-white.csv', delimiter=';')
data = numpy.delete(data, 0, axis=0)
print(data)
X = data[:-100, :-1]
y = data[:-100, -1]
m = Perceptron().fit(X, y)
r = m.predict(data[100:, :-1])
print(r)
g = data[100:, -1]
print(g)
print(precision_score(g, r, average='micro'))
```

```
[[ 7.    0.27  0.36 ...  0.45  8.8   6.  ]
 [ 6.3   0.3   0.34 ...  0.49  9.5   6.  ]
 [ 8.1   0.28  0.4  ...  0.44 10.1   6.  ]
 ...
 [ 6.5   0.24  0.19 ...  0.46  9.4   6.  ]
 [ 5.5   0.29  0.3  ...  0.38 12.8   7.  ]
 [ 6.    0.21  0.38 ...  0.32 11.8   6.  ]]
[5. 5. 5. ... 5. 5. 5.]
[5. 5. 5. ... 6. 7. 6.]
0.3249270529387245
```

支持向量机

- 支持向量（Support Vector）
- 给定一个训练样本集
- 在样本空间中找到一个超平面将不同类别的样本分开
- 距离超平面最近的这几个训练样本构成了支持向量（Support Vector）



支持向量机

```
import numpy
from sklearn.metrics import precision_score
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC

data = numpy.genfromtxt('winequality-white.csv', delimiter=';')
data = numpy.delete(data, 0, axis=0)
print(data)
X = data[:-100, :-1]
y = data[:-100, -1]
m = make_pipeline(StandardScaler(), SVC(gamma='auto')).fit(X, y)
r = m.predict(data[100:, :-1])
print(r)
g = data[100:, -1]
print(g)
print(precision_score(g, r, average='micro'))
```

```
[[ 7.    0.27  0.36 ...  0.45  8.8   6.  ]
 [ 6.3   0.3   0.34 ...  0.49  9.5   6.  ]
 [ 8.1   0.28  0.4  ...  0.44 10.1   6.  ]
 ...
 [ 6.5   0.24  0.19 ...  0.46  9.4   6.  ]
 [ 5.5   0.29  0.3  ...  0.38 12.8   7.  ]
 [ 6.    0.21  0.38 ...  0.32 11.8   6.  ]]
[6. 6. 5. ... 5. 7. 6.]
[5. 5. 5. ... 6. 7. 6.]
0.6158816173405586
```

高斯朴素贝叶斯分类器

- 高斯朴素贝叶斯分类器（Gaussian Naïve Bayes Classifier）
- 特征条件独立性假设：对已知类别假设所有特征相互独立
- 换言之，假设每个特征独立地对分类结果发生影响
- 朴素贝叶斯分类器的训练过程就是：

1. 基于训练集 D 来估计类先验概率 $P(c)$
2. 为每个特征估计条件概率 $P(x_i|c)$

- 采用高斯分布（正态分布）作为基础分布概率

高斯朴素贝叶斯分类器

```python
import numpy
from sklearn.metrics import precision_score
from sklearn.naive_bayes import GaussianNB

data = numpy.genfromtxt('winequality-white.csv', delimiter=';')
data = numpy.delete(data, 0, axis=0)
print(data)
X = data[:-100, :-
y = data[:-100, -1
m = GaussianNB()
m.fit(X, y)
r = m.predict(data[100:, :-1])
print(r)
g = data[100:, -1]
print(g)
print(precision_score(g, r, average='micro'))
```

```
[[ 7.    0.27  0.36 ...  0.45  8.8   6.  ]
 [ 6.3   0.3   0.34 ...  0.49  9.5   6.  ]
 [ 8.1   0.28  0.4  ...  0.44 10.1   6.  ]
 ...
 [ 6.5   0.24  0.19 ...  0.46  9.4   6.  ]
 [ 5.5   0.29  0.3  ...  0.38 12.8   7.  ]
 [ 6.    0.21  0.38 ...  0.32 11.8   6.  ]]
[5. 5. 5. ... 6. 7. 7.]
[5. 5. 5. ... 6. 7. 6.]
0.4468528553563985
```

作业

- UCI 葡萄干分类数据集
- https://archive.ics.uci.edu/dataset/850/raisin

Data Set Characteristics:	Multivariate	**Number of Instances:**	900	**Area:**	Life
Attribute Characteristics:	Integer, Real	**Number of Attributes:**	8	**Date Donated**	2021-04-01
Associated Tasks:	Classification	**Missing Values?**	N/A	**Number of Web Hits:**	1541601

- 两种土耳其葡萄干的视觉观测数据，已量化
- 葡萄干的类别（Kecimen 或 Besni）

作业

- 获取葡萄干分类数据：https://archive.ics.uci.edu/ml/machine-learning-databases/00617/Raisin_Dataset.zip
- 取后 100 条为测试数据，其他为训练数据
- 用训练数据训练至少三个分类模型
- 在测试数据上测试训练好的分类模型
- 使用准确度（Precision）评估模型效果

- 提供完整的代码
- 提供完整的实验结果截图

作业


Run: ai
1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1.]
0.8125
Process finished with exit code 0
Version Control
Run
Python Packages
TODO
Python Console
Problems
Terminal
Endpoints
Services
Bookmarks
Structure

作业

- 在作业系统中下载并完成本实验课对应实验报告
- https://hw.dgut.edu.cn/
- 注意：所有标识为 * 的地方都需要填写
- 截止日期：2024-05-06 23:59:59

课程名称：人工智能概论

实验名称	回归模型		
姓名	***	学号	***
实验地点	***	实验日期	***

四、 实验作业及分析

4.1 实验过程

*** 请将详细实验过程的截图和相关说明填写在此处 ***

4.2 实验结果

*** 请将实验结果的截图和相关说明填写在此处 ***


GOOD
LUCK!