Q1 实验报告

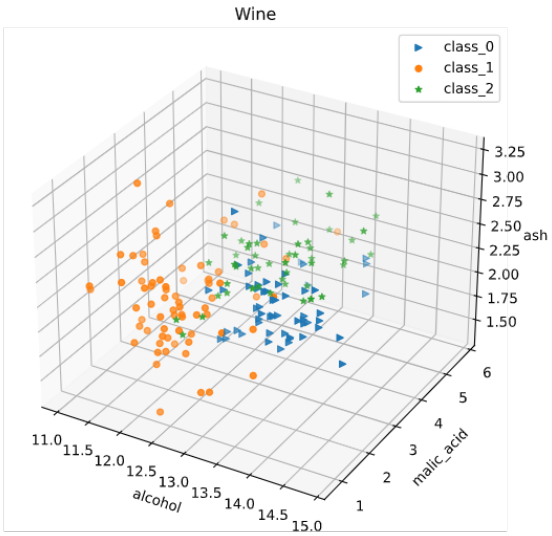
1854116 朱明志

目录

1 数据描述

这两张图片是小汽车驾驶位置的照片，照片拍摄于行驶时。用这两张 640*480 像素的图片数据为例解释维度灾难。

使用 sklearn 包的 Wine 果酒分类数据集，一共 3 个类别，178 个样本，每个样本具有 13 个特征，对这个数据集进行降维操作。选取三个特征进行降维，这三个特征分别为：alcohol、malic_acid、ash。降维前数据分布如下：



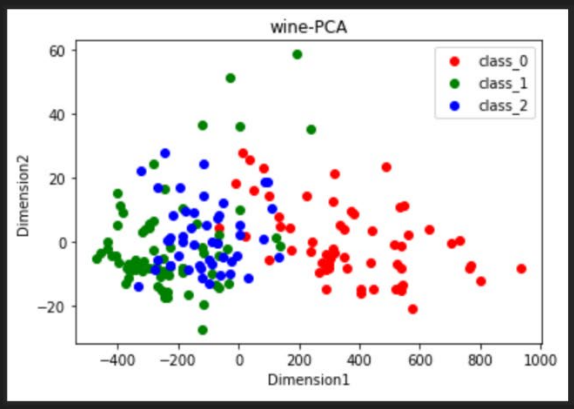
使用 sklearn 包的 Iris 鸢尾花数据集。这是一个经典分类数据集，一共 3 个类别，150 个样本，每个样本具有 4 个特征。

2 代码运行结果屏幕拷贝

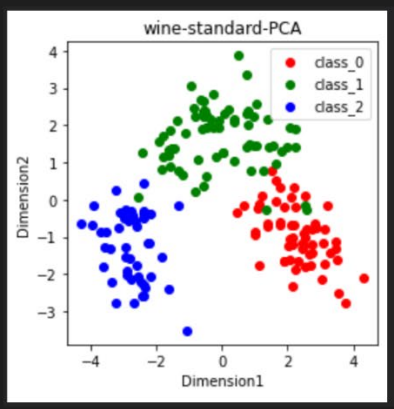
2.1 降维操作运行结果

```
[33]: # 导入果酒数据...
```

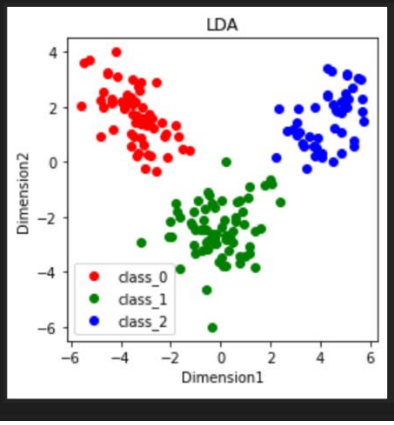
```
[34]: # 利用PCA降维，降到二维...
```




```
[35]: # 标准化后降维...
```



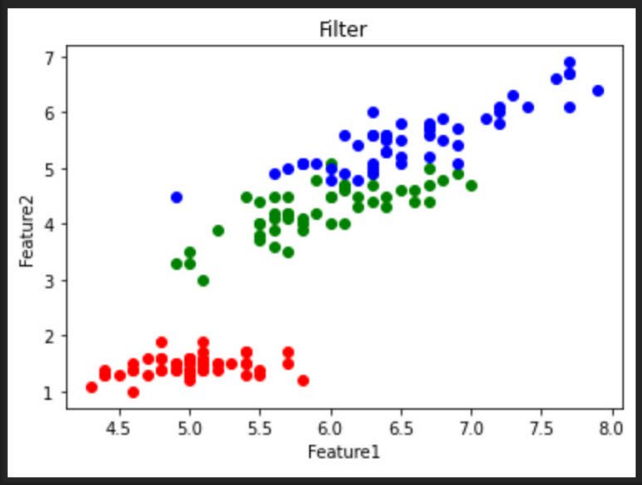

```
[36]: # 有监督的LDA降维...
```



2.2 特征选择操作

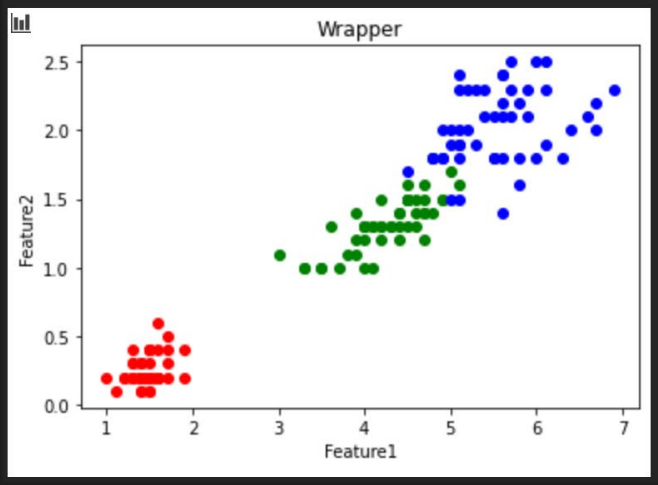
```
[30] #方差选择法，返回值为特征选择后的数据...
```

```
csv save!
```



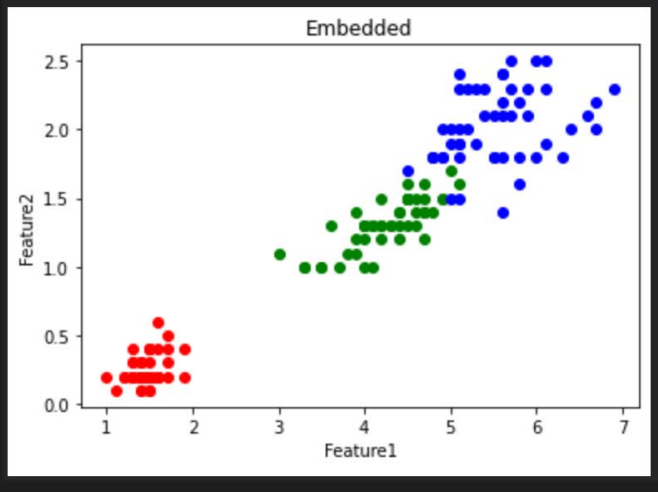

```
[31] #递归特征消除法，返回特征选择后的数据...
```

```
csv save!
```




```
[32] #带L2惩罚项的逻辑回归作为基模型的特征选择...
```

```
csv save!
```



3 实验内容讨论

3.1 维度灾难问题回答

对于绝大多数数据而言，在一维空间或者说低维空间都是很难完全分割的，与之相反在高维空间就往往很容易找到一个超平面，将这些数据完美分割。

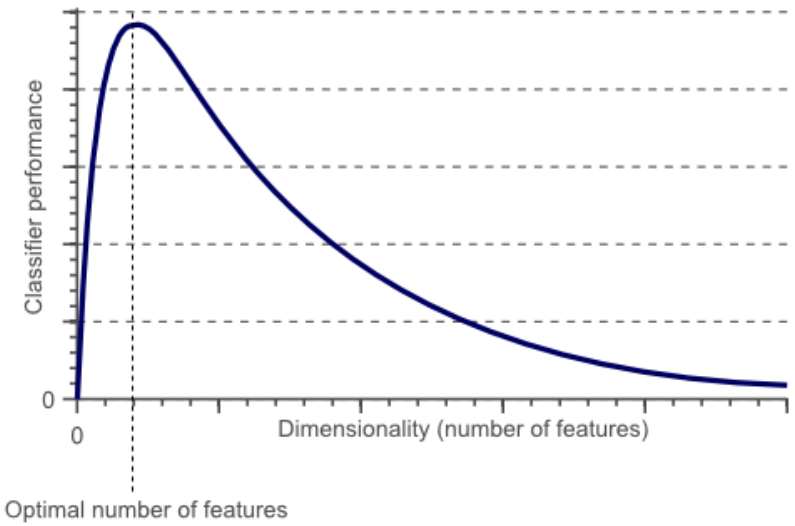
以上 2 张图为例，我们要判断图中司机是在玩手机还是在正常行驶。一张图片相当于 640*480 个像素点的矩阵，一个像素点有 3 个属性，代表该点的三个 RGB 值。一个简单的想法是：用一个线性的分类器区分它们，将红、绿、蓝三种颜色作为识别特征，将每个像素点的三种颜色的 RGB 值分别加权平均得到三个值（green，red，blue）。而这个分类器的阈值应该是由大量的数据训练得到，通过数据不断训练这个分类器，得到现在的这个分类器阈值。

```
def div(green,red,blue):
    if ((0.5*red> 79)&(0.3*green> 132)&(0.2*blue> 200)):
        return "mobile"
    else:
        return "driving"
```

但是这样的分类器显然是过于简单的，不能很好的区分两种照片。为此我们需要增加一些特征，例如像素矩阵的秩、行列数据的梯度等，这样就有 6 个特征了，对于每一个特征，分类器都可以学习出一个阈值用于分类。

我们为了得到更为精确的分类器，基于各种各样的统计差动、色彩等找出各种各样的特征。但是有限的增加特征可以提高分类器的准确率，但是特征的数量一旦超过一个极值，再增加特征，反而会让分类器的性能下降。这就是“维度灾难”。

一张经典的描述维度灾难的图见下：



可以看出，刚开始随着维度的增加，分类器的性能有着显著提升，但是分类器性能到达峰值后，再增加维度，分类器的性能反而下降了。若无限增加维度，这个分类器很快就变的没有使用价值了。

维度灾难带来最直接的问题就是过拟合。假设我们有 10 个训练集，在 1 特征空间为 1 维时，10 个训练实例覆盖在一维轴上，假设轴长为 5，那么每个样本平均每个单位有 2 个样本。特征空间为 2 维时，我们仍然用 10 个实例进行训练，此时二维空间的面积是 5x5=25，样本密度是 10/25=0.4，即每个单位面积有 0.4 个样本。在特征空间为 3 维时，10 个样本的密度是 10/(5x5x5)=0.08，即每个单位体积有 0.08 个样本。

如果我们不断的增加特征，特征维度就会不断变大，同时变得越来越稀疏。由于稀疏的原因，随着特征维度的不断变大，我们很容易的就找到一个能将样本按类别完美分开的超平面，因为训练样本落到该空间的最优超平面错误一边的概率会随着维度增加无限变小。但是，如果将高维分类映射回低维，我们能很容易发现一个严重的问题: 使用太多的特征导致了过拟合。分类器学习了很多异常特征（如噪声等），因此对于新的数据泛化性能不好。

假设一个 3D 分类结果投影到 2D 空间中的情景，我们可以发现不像高维空间中，在低维空间中，数据并没有显示出可分性。而实际上，通过增加第三维度来获得最优分类效果等价于在低维空间使用复杂的非线性分类器，而往往复杂的模型结构也是导致过拟合的原因之一。结果就是分类器学习到了很多数据集中的特例，因此对于现实数据往往会效果较差，因为现实数据是没有这些噪声以及异常特性的。

因此说，过拟合是维度灾难带来的最直接结果。

简单的分类器对于未知的数据具有更好的泛化性能，因为其不会学习到训练集中偶然出现的特例。换句话说，维度灾难可以通过使用更少的特征来避免，同时这样分类器就不会对训练数据过拟合。

从另一个角度来阐述，我们假设每个特征的范围是从 0-1，同时每个喝水和玩手机都具有不同的特征。如果我们想使用全数据的 20%的数据来训练模型，那么在一维的情况下，我们的特征范围只需要取 20%的范围，也就是 0.2 就够了。当上升到 2 维的情况下，我们需要 45%的每维特征（0.45*0.45=0.2）才可以覆盖特征空间中的 20%，当上升到 3 维空间时，我们则需要每维 58%的特征范围（0.58^3 = 0.2）。

换句话说，如果特征数目一定，那么随着维度增加过拟合就会出现。另一方面，如果持续增加维度，那么训练数据需要指数级的增加才能保持同样的距离分布来避免过拟合。例如在一维空间中，样本密度是 1000 个/单位，那么我们在任意测试样本 0.001 的单位距离都可以找到一个样本。但是如果到 2 维空间，那么我们需要 1000^2 个样本才能保证任意样本的 0.001 单位距离内都有一个样本。

3.2 降维实验

3.2.1 实验原理

- PCA（主成分分析）

  Principal Component Analysis(PCA)是最常用的线性降维方法，它的目标是通过某种线性投影，将高维的数据映射到低维的空间中表示，并期望在所投影的维度上数据的方差最大，以此使用较少的数据维度，同时保留住较多的原数据点的特性。

通俗的理解，如果把所有的点都映射到一起，那么几乎所有的信息（如点和点之间的距离关系）都丢失了，而如果映射后方差尽可能的大，那么数据点则会分散开来，以此来保留更多的信息。可以证明，PCA 是丢失原始数据信息最少的一种线性降维方式。

- LDA（线性判别分析）

判别分析(Discriminant Analysis) 所追求的目标与 PCA 不同，不是希望保持数据最多的信息，而是希望数据在降维后能够很容易地被区分开来。

与 PCA 不同，LDA 是一种有监督的（supervised）线性降维算法。与 PCA 保持数据信息不同，LDA 的核心思想：往线性判别超平面的法向量上投影，是的区分度最大（高内聚，低耦合）。LDA 是为了使得降维后的数据点尽可能地容易被区分!

3.2.2 实验过程

本次实验使用了果酒数据集数据的 alcohol、malic_acid、ash 属性。

本次实验分别实验了两种不同的降维方法对这些数据进行了降维：

1. 直接 PCA 降维；
2. 标准化数据后进行 PCA 降维；
3. LDA 有监督降维。

使用 Python 语言进行实现，具体实现见代码。

3.3 特征选择实验

3.3.1 实验原理

常来说，从两个方面考虑来选择特征：

1. 特征是否发散：如果一个特征不发散，例如方差接近于 0，也就是说样本在这个特征上基本上没有差异，这个特征对于样本的区分并没有什么用。
2. 特征与目标的相关性：这点比较显见，与目标相关性高的特征，应当优选选择。除方差法外，本文介绍的其他方法均从相关性考虑。

根据特征选择的形式又可以将特征选择方法分为 3 种：

1. Filter：过滤法，按照发散性或者相关性对各个特征进行评分，设定阈值或者待选择阈值的个数，选择特征。
2. Wrapper：包装法，根据目标函数（通常是预测效果评分），每次选择若干特征，或者排除若干特征。
3. Embedded：嵌入法，先使用某些机器学习的算法和模型进行训练，得到各个特征的权值系数，根据系数从大到小选择特征。类似于 Filter 方法，但是是通过训练来确定特征的优劣。

- 方差选择法（Filter 类）

使用方差选择法，先要计算各个特征的方差，然后根据阈值，选择方差大于阈值的特征。

- 递归特征消除法（Wrapper 类）

递归消除特征法使用一个基模型来进行多轮训练，每轮训练后，消除若干权值系数的特征，再基于新的特征集进行下一轮训练。

- 基于惩罚项的特征选择法（Embedded 类）

使用带惩罚项的基模型，除了筛选出特征外，同时也进行了降维。

3.3.2 实验过程

本次实验使用了鸢尾花数据集数据。

本次实验分别实验了三种不同的特征选择方法对这些数据进行了特征选择：

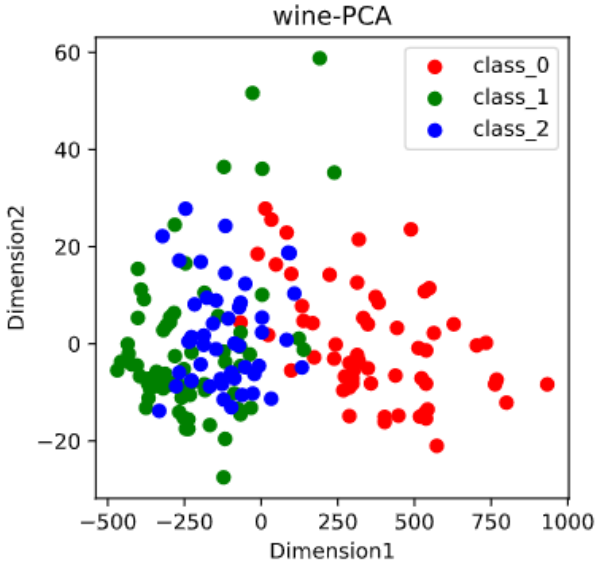
1. 使用 feature_selection 库的 VarianceThreshold 类来选择特征（方差阈值设置为 0.6）；
2. 使用 feature_selection 库的 RFE 类来选择特征；
3. 使用 feature_selection 库的 SelectFromModel 类结合带 L2 惩罚项的逻辑回归模型来选择特征。

使用 Python 语言进行实现，具体实现见代码。

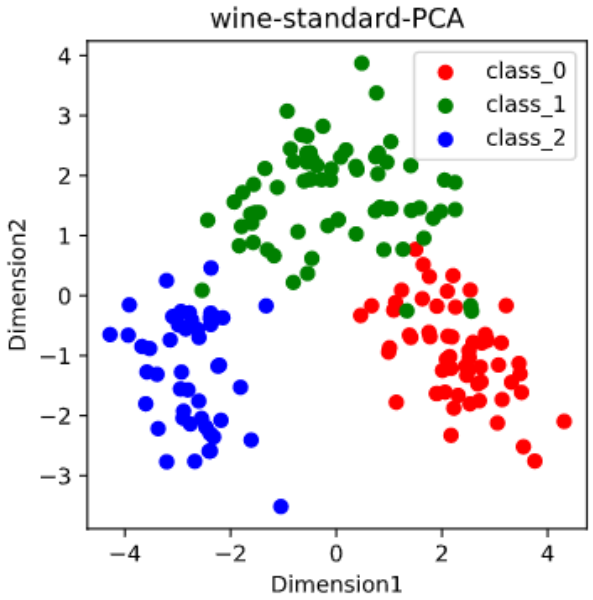
4 实验结果图表与分析

4.1 降维操作

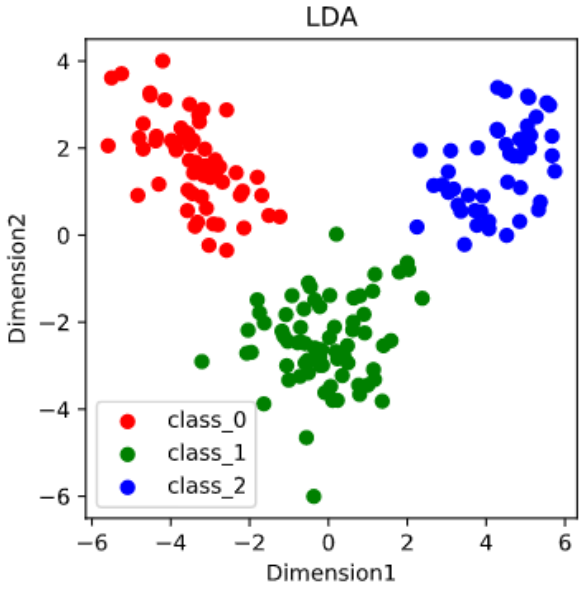
- PCA 将三维数据降至二维



- 标准化数据后 PCA 降维



- 有监督的 LDA 降维



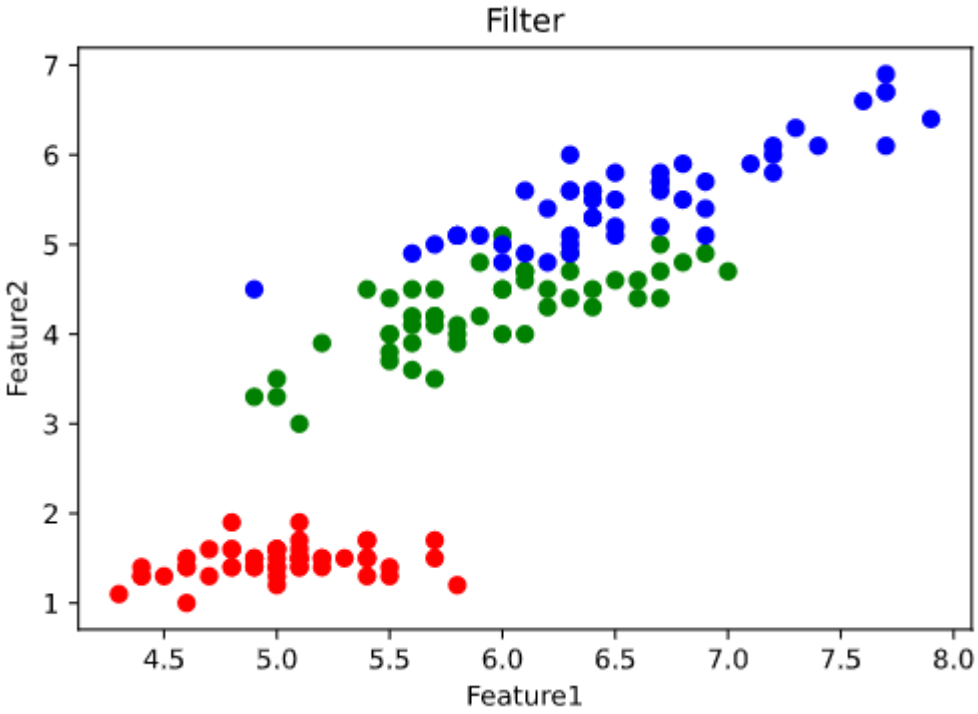
由上三张图可见，与 PCA 相比，LDA 能够将三类样本完全分开，且同类样本之间更为密集。一般而言，有监督的 LDA 更加准确。而标准化数据对提高 PCA 降维的性能有帮助。

4.2 特征选择操作

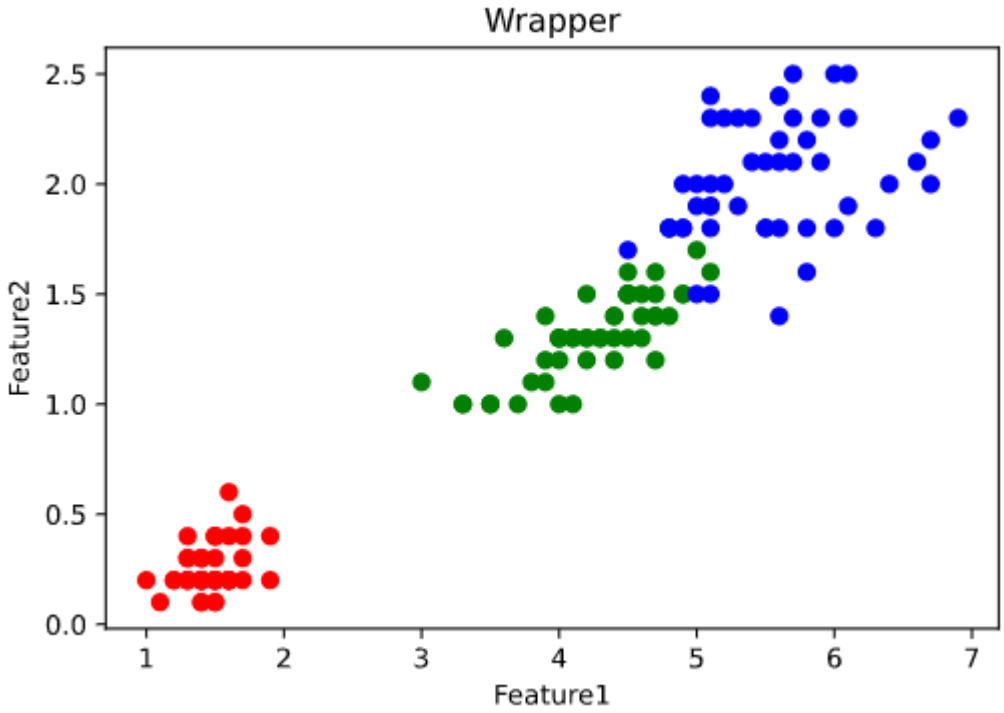
表格文件名	表格内容
out1.csv	方差选择法选取特征结果
out2.csv	递归特征消除法选取特征结果
out3.csv	基于惩罚项的特征选择法选取特征结果

数据表格的原始文件都在 csv 子文件夹下，每个数据表的具体内容见上表。

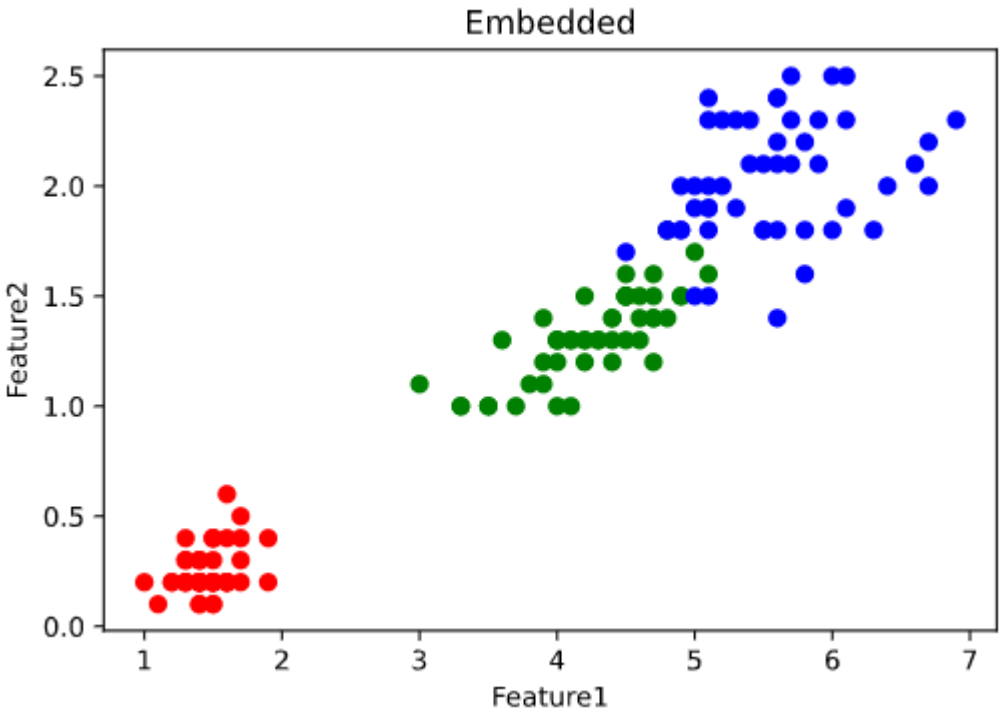
- 方差选择法选取特征



- 递归特征消除法



- 基于惩罚项的特征选择法


Embedded
Feature2
Feature1
2.5
2.0
1.5
1.0
0.5
0.0
1
2
3
4
5
6
7