scikit-learn 实战之监督学习 (/courses/866)

课程名称：scikit-learn 实战之监督学习

一、实验介绍

1.1 实验内容

监督学习（英语：Supervised learning）是机器学习中最为常见、应用最为广泛的分支之一。本次实验将带你了解监督学习中较为基础的广义线性模型，并学会使用 scikit-learn 来构建预测模型，用于解决实际问题。

1.2 实验知识点

- 广义线性回归模型介绍。
- 训练一个广义线性回归模型。
- 训练一个广义线性分类模型。

1.3 实验环境

- python3

1.4 适合人群

本课程难度为一般，属于初级级别课程，适合具有 Python 基础和线性代数基础，并对机器学习中分类问题感兴趣的用户。

scikit-learn 实战之监督学习 (/courses/866)

二、广义线性回归模型

scikit-learn 中包含的广义线性模型有最小二乘回归、感知机、逻辑回归、岭回归，贝叶斯回归等，由 `sklearn.linear_model` 模块导入。对于广义线性模型而言，即通过拟合线性函数（下图）去完成样本分类或回归预测。

$$y(w,x) = w_0 + w_1 \cdot x_1 + \cdots + w_p \cdot x_p$$

其中，最小二乘回归、岭回归、贝叶斯回归等是用于解决回归问题。而感知机、逻辑回归被用于解决分类问题。

2.1 最小二乘回归

这里，我们从比较基础的最小二乘法回归说起。最小二乘法是线性回归中最经典的方法之一，最小二乘的取名即来自于其选择了平方损失函数。在 scikit-learn 中，最小二乘法的实现方法如下：

名称	方法
最小二乘回归	sklearn.linear_model.Lin

使用 scikit-learn 去解决一个机器学习相关的问题时，我们的代码都大同小异，主要是由几个部分组成：

1. 调用一个机器学习方法构建相应的模型 `model` ，并设置模型参数。
2. 使用该机器学习模型提供的 `model.fit()` 训练模型。
3. 使用该机器学习模型提供的 `model.predict()` 方法用于预测。

下面，我们尝试通过最小二乘回归去拟合二维

scikit-learn 实战之监督学习 (/courses/866)

平面上的一些点。首先，执行上面的第一步，载入方法并构建模型。

☞ **示例代码：**

```
from sklearn import linear_model

model = linear_model.LinearRegressio
n() # 调用最小二乘回归方法
```

☞ **动手练习：**

接下来，使用模型带有的 `fit()` 方法去拟和 3 个点。三个点的特征向量分别为 [0, 0], [1, 1], [2, 2] , 对应的目标值为 [1, 2, 3]。

☞ **示例代码：**

```
model.fit ([[0, 0], [1, 1], [2, 2]],
[1, 2, 3]) # 模型拟合
```

☞ **动手练习：**

然后，我们就可以看到模型返回的参数。当我们调用方法且不填入任何参数时，即代表采用默认参数。

```
LinearRegression(copy_X=True, fit_in
tercept=True, n_jobs=1, normalize=Fa
lse)
```

训练时，选择的 [0, 0], [1, 1], [2, 2] 这三个点恰好在一条直线上，再结合目标值想象一下它们的空间位置关系。我们可以使用下面的方法，输出拟合直线 w 项和常数项值。

☞ **示例代码：**

scikit-learn 实战之监督学习 (/courses/866)


```
print(model.coef_)
print(model.intercept_)
```

☞ **动手练习：**

☞ **参考结果：**

```
[0.5 0.5]
1.0
```

如图所示，即实验拟合的函数应该为：

$$y(x) = 0.5 \cdot x_1 + 0.5 \cdot x_2 + 1$$

当我们输入新的数值，例如 [3,3] 时，根据上面的函数，因变量的值为 4。那么，我们使用模型来预测，看一看结果是否为 4?

☞ **示例代码：**

```
model.predict([[3, 3]])
```

☞ **动手练习：**

☞ **参考结果：**

```
array([4.])
```

结果的确和我们预想的一致，也标志着我们通过 scikit-learn 完成了一个基础的线性回归问题。

2.2 复杂实例

上面的例子比较简单，下面我们导入 scikit-learn 内置的 `diabetes` 糖尿病训练集来训练一个复杂一点的最小二乘回归模型。

第一步：导入数据，并将其划分为 70% 的训

scikit-learn 实战之监督学习 (/courses/866)

练集和 30% 的测试集。机器学习中，我们习惯采用这样的比例来划分训练集和测试集。

☞ **示例代码：**

```
from sklearn import datasets # 导入内置数据集模块
from sklearn.model_selection import train_test_split # 导入数据集切分模块
import numpy as np # 导入数值计算模块

diabetes = datasets.load_diabetes() # 载入糖尿病数据集

diabetes_feature = diabetes.data[:, np.newaxis, 2] # 该数据集的特征较多，这里只选取其中一个
diabetes_target = diabetes.target #设定目标值

train_feature, test_feature, train_target, test_target = train_test_split(diabetes_feature, diabetes_target, test_size=0.33,random_state=56) # 切分数据集为 77% 的训练集和 33% 的预测集
```

☞ **动手练习：**

第二步：载入最小二乘回归模型，并训练数据。

☞ **示例代码：**

```
from sklearn import linear_model # 导入线性模型模块

model = linear_model.LinearRegression() # 构建最小二乘线性回归模型
model.fit(train_feature, train_target) # 使用训练集数据训练模型
```

☞ **动手练习：**

第三步：使用测试集进行预测，并将结果绘图。

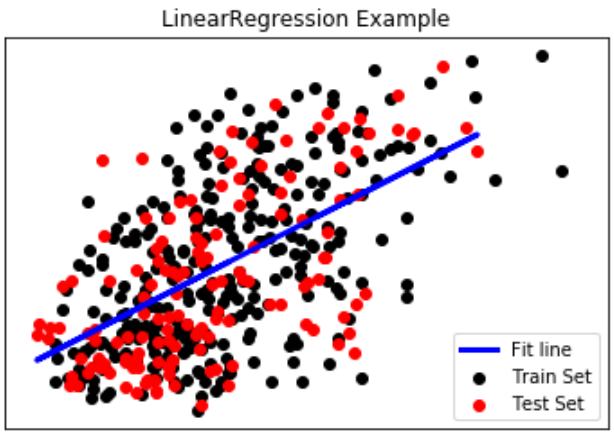
scikit-learn 实战之监督学习 (/courses/866)

☞ **示例代码：**

```
import matplotlib.pyplot as plt # 导入 matplotlib 绘图模块
%matplotlib inline

# 绘图
plt.scatter(train_feature, train_target,  color='black') # 绘制训练集散点图
plt.scatter(test_feature, test_target,  color='red') # 绘制测试集散点图
plt.plot(test_feature, model.predict(test_feature), color='blue',
         linewidth=3) # 绘制拟合直线

# 绘制图例
plt.legend(('Fit line', 'Train Set', 'Test Set'), loc='lower right')
plt.title('LinearRegression Example')

plt.xticks(())
plt.yticks(())

plt.show()
```


☞ **动手练习：**

最后，我们可以通过绘制的图像，更加直观地看出采用最小二乘回归模型进行线性拟合的结果。



2.3 其他线性回归模型

scikit-learn 实战之监督学习 (/courses/866)

对于其他常见的线性回归模型，它们和最小二乘线性回归模型非常相似，只是采用了不同的损失函数。

例如，岭回归采用了带罚项的残差平方和损失函数。

$$\min_w ||Xw - y||_2^2 + \alpha||w||_2^2$$

而另一种常见的 Lasso 回归，同样采用了带 L1 范数的罚项平方损失函数。

$$\min_w \frac{1}{2n_{samples}}||Xw - y||_2^2 + \alpha||w||_1$$

下面列举了一些常见的广义线性回归模型，及它们在 scikit-learn 中对应的方法。

名称	方法
贝叶斯岭回归	sklearn.linear_mode
Lasso 回归	sklearn.linear_mode
岭回归	sklearn.linear_mode
随机梯度下降回归	sklearn.linear_mode
鲁棒回归	sklearn.linear_mode

这些方法相对于普通最小二次线性回归模型而言，均增加了一些罚项。这样会提示模型的泛化能力，在实际应用中效果会好一些。

三、广义线性分类模型

除了线性回归，scikit-learn 还提供了一些解决线性分类的方法。其中，感知机就是非常有代表性的线性分类模型。

3.1 感知机

scikit-learn 实战之监督学习 (/courses/866)

感知机是一个经典的二分类方法，由 Rosenblatt 于 1957 年时提出。它是神经网络和支持向量机的基础。感知机模型非常简单，输入为一些特征向量，输出则由正类和负类组成。而输入和输出之间，则是由符号函数连接。

$$f(x) = sign(w \cdot x + b); sign(x) = \begin{cases} +1, x \geq 0 \\ -1, x \leq 0 \end{cases}$$

感知机的损失函数是错误分类点到分离超平面之间的距离总和，其学习策略同样也是损失函数最小化。

$$\min \sum y(w \cdot x + b)$$

在 scikit-learn 中，实现感知机通过调用 `sklearn.linear_model.Perceptron()` 方法完成。

下面，我们通过一个例子来展示感知机的分类过程。首先，使用 scikit-learn 提供的 `make_classification` 方法生成一组可被二分类的二维数组作为数据集。

☞ **示例代码：**

```
from sklearn.datasets import make_classification # 导入分类数据生成模块

X1, Y1 = make_classification(n_features=2, n_redundant=0, n_informative=1, n_clusters_per_class=1) # 随机生成一组可以被二分类的数据
```

☞ **动手练习：**

这里，我们可以使用 matplotlib 将该数据集绘制出来。

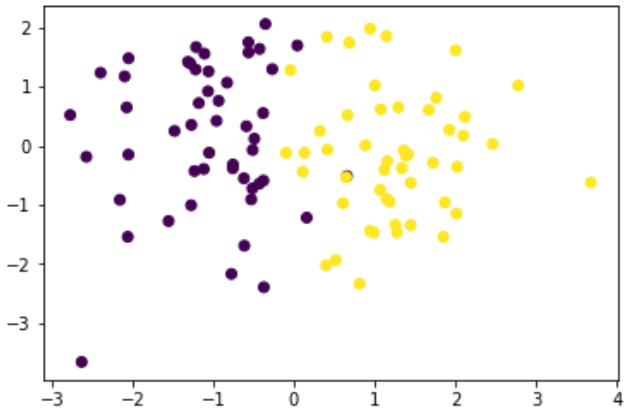
☞ **示例代码：**

scikit-learn 实战之监督学习 (/courses/866)


```
plt.scatter(X1[:, 0], X1[:, 1], marker='o', c=Y1) # 绘制数据集散点图
plt.show() # 显示图
```

☞ **动手练习：**

☞ **参考结果：**



我们可以看到，数据集分为红点和蓝点，并呈现出明显的线性界线。接下来，我们使用感知机对该数据集进行分类训练。

☞ **示例代码：**

```
from sklearn.linear_model import Perceptron # 感知机模块

train_feature, test_feature, train_target, test_target = train_test_split(X1, Y1, test_size=0.33, random_state=56) # 将数据集划分为 70% 训练集和 30% 测试集

model = Perceptron() # 建立感知机模型，使用默认参数
model.fit(train_feature, train_target) # 使用训练集训练模型
```

☞ **动手练习：**

训练结束后，我们用测试数据进行预测。这里使用 matplotlib 将测试数据绘制在原图上，

并将预测结果作为标签显示

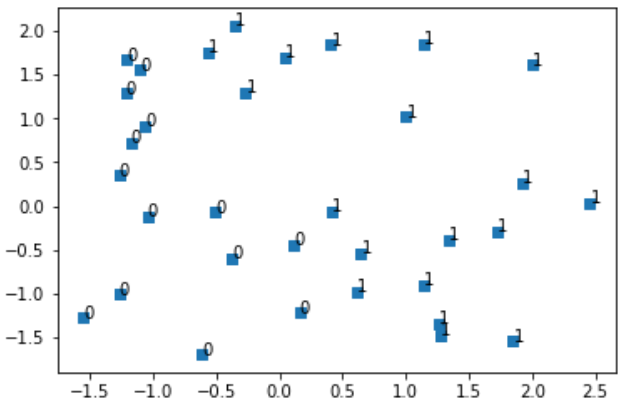
scikit-learn 实战之监督学习 (/courses/866)

☞ **示例代码：**

```
results = model.predict(test_feature
) # 使用测试集预测

plt.scatter(test_feature[:, 0], test
_feature[:, 1], marker=',') # 以方块样
式绘制测试数据

# 将预测结果用标签样式标注在测试数据左上方
for i, txt in enumerate(results):
    plt.annotate(txt, (test_feature[
:, 0][i],test_feature[:, 1][i]))

plt.show() # 显示图
```


☞ **动手练习：**

由于本次实验所使用的数据集是通过 `make_classification` 随机生成的，所以每一次运行的结果都不一样。



现在，我们就完成了一个简单的感知机分类实验。可以看出来，对于二分类问题，感知机的预测效果还是很不错的。

四、实验总结

广义线性模型是机器学习中十分简单基础的模型。但是由于其本身的特点，只能用于二分类

scikit-learn 实战之监督学习 (/courses/866)

问题。对于实际生活中经常遇到的多分类及非线性分类问题，无法适用。但对于刚刚入门机器学习的朋友来说，线性分类模型是不错范例。

五、课后习题

1. 尝试通过 `make_classification()` 方法随机生成 300 条可用于二分类的数据，并通过 `sklearn.linear_model.LogisticRegression()` 调用广义线性模型中的逻辑回归方法完成分类。最后通过 matplotlib 将训练集和测试集绘制出来。

上一节：监督学习简介 (/courses/866/labs/3155/document)

下一节：支持向量机用于分类和回归预测- 实战进阶 (/courses/866/labs/3158/document)