项目导学

【项目概述】

机器学习已应用于许多领域，远超出大多数人的想象，下面提到的每一个场景都会碰到机器学习：假设你想送朋友一份礼物，你打开浏览器搜索生日礼物，搜索引擎显示了 10 个最相关的链接，你选择了第二个链接，搜索引擎将记录这次点击，并从中学习以优化下次搜索结果。当你自驾去超市，路上导航会根据目前的交通状况为你推荐最佳路线，到了超市你给朋友的孩子挑选尿布，结账时，收款台软件基于以前的统计知识，会送你一张优惠券，可以用于购买灌装啤酒。

上面提到的所有场景，都有机器学习应用的存在。现在很多公司使用机器学习技术改善商业决策、提高生产率、检测疾病、预测天气等等。那么机器学习技术到底有哪些具体的技术，这些技术具体是如何应用的呢？

【项目目标】

(1)了解机器学习的定义、机器学习的工作流程和社会价值；

(2)了解机器学习的主要任务，以及每类任务的特点，适合处理什么样的问题；

(3)熟悉机器学习技术应用的常用开发框架和工具，了解其特点和适用范围；

(4)了解机器学习的核心技术和经典算法，能应用机器学习技术解决实际问题；

(5)能辨析机器学习在社会应用中面临的道德和法律问题。

【项目分析】

该教学模块由 8 个教学任务与 8 个主要内容组成，通过引入机器学习在行业中的应用，了解机器学习技术对我们生活带来的改变。增强机器学习的知识，体会机器学习的应用价值，了解机器学习算法的基本原理及技术处理流程，让人们在享受机器学习带来的便捷的同时，也要让人们对机器学习的滥用产生自我保护意识。

任务一　线性回归

建议学时：4～6 学时

任务描述

通过本任务的学习，理解线性回归的基本概念、基本特征；掌握线性回归的建模方法，线性回归参数的求解原理；能够对线性回归的参数给出合理的分析与解释。

任务目的

（一）认识线性回归的模型；

（二）了解线性回归模型的应用场景，以及线性回归参数的意义；

（三）掌握线性回归建模过程。

任务要求

（一）以小组的形式完成任务；

（二）完成一元线性回归的建模代码实现；

（三）完成一元线性回归实践报告 1 份。

基础知识

1. 线性回归的来源

“回归”一词的英文是 Regression，本义里有“衰退”的意思。实际上，“回归”一词来源于生物统计学家高尔顿研究父母身高和子女身高时发现，即使父母的身高都“极端”高，其子女不见得会比父母高，而是有“衰退”(regression)(也称作“回归”)至平均身高的倾向。高尔顿当时拟合了父母平均身高 x 和 子女平均身高 y 的经验方程：

$$y = 3.78 + 0.516\,x$$

可以看到，父代身高每增加一个单位，其成年子女的平均身高只增加 0.516 个单位，它反映了这种“衰退”效应(“回归”到正常人平均身高)。虽然之后的 x 与 y 变量之间并不总是具有“衰退”(回归)关系，但是为了纪念高尔顿这位伟大的统计学家，“线性回归”这一名称就保留了下来。

回归分析中，只包括一个自变量和一个因变量，且二者的关系可用一条直线近似表示，这种回归分析称为**一元线性回归分析**。如果回归分析中包括两个或两个以上的自变量，且因变量和自变量之间是线性关系，则称为多元线性回归分析。

2. 一元线性回归

一元线性回归的主要任务是从两个相关变量中的一个变量去估计另一个变量，被估计的变量，称因变量，可设为 y；另一个称自变量，设为 x。如上面身高的例子所示，在模型中有一个自变量 x，自变量 x 与因变量 y 之间存在着直线关系，其模型可以统一写为：

$$y = b + wx + \varepsilon$$

在式中，b 称为截距，w 称为斜率，ε 为随机误差。在线性回归中，通常假设随机误差是不相关的，且均值为 0，方差 σ^2 未知。

当 b 和 w 已知的时候，画在坐标图内是一条直线(这就是“线性”的含义)，如图 1-1 所示。线性回归就是要找一条直线，并且让这条直线尽可能地拟合图中的数据点。

3. 一元线性回归分析过程

以“食品利润”数据集为例，使用 sklearn 估计器构建一元线性回归模型，其过程如下：

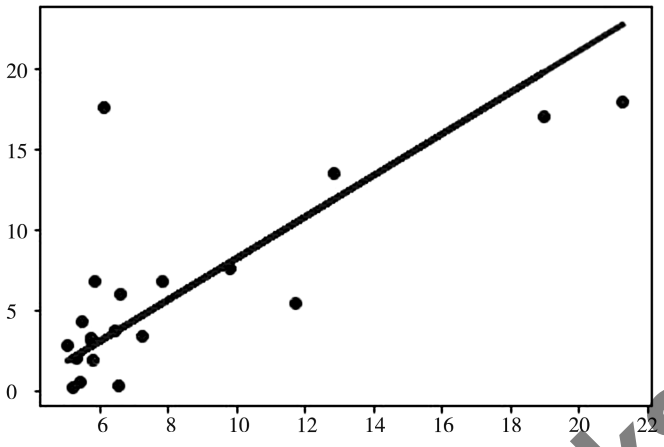

图 1-1 一元线性回归示意图

(1)确定自变量为城市人口(Population),数据集中的第一列数据,因变量为利润(Profit),数据集中的第二列数据,负数表明亏损。数据集有 97 个观测值。数据样例如下所示。

```
6.1101,17.592
5.5277,9.1302
8.5186,13.662
7.0032,11.854
5.8598,6.8233
8.3829,11.886
7.4764,4.3483
8.5781,12.
6.4862,6.5987
5.0546,3.8166
```

(2)将数据集划分为训练集和测试集,其中训练集用于构建回归模型,得到回归系数,测试集用于对构建的回归模型进行评估。一般情况下训练集和测试集的比例为 8:2 或 7:3。

sklearn 的 model_selection 模块提供了 train_test_split 函数,能够实现对数据集的拆分。

(3)在训练集上进行线性回归训练,得到线性回归的系数。

sklearn 的 linear_model 模块提供了 LinearRegression 函数,能够实现线性回归建模,可以使用 fit、predict、score 来训练、评价训练模型。

```
Model =sklearn.linear_model.LinearRegression()      # 建立模型
Model.fit(x,y)  # 训练模型
Model.score(x,y) # 评估模型
```

(4)在测试集上评估回归模型。

常用的回归模型评价指标有平均绝对误差、均方误差、中值绝对误差、可解释方差值和R^2值。如下所示为R^2值。

```
y_test_pre = Model.predict(x)   # 模型预测
r2_score(y_test, y_test_pred_lr) # 模型评价
```

4. 多元线性回归

在回归分析中,如果有两个或两个以上的自变量,就称为多元回归。事实上,一种现象常常是与多个因素相联系的,由多个自变量的最优组合共同来预测或估计因变量,比只用一个自变量进行预测或估计更有效,更符合实际。因此多元线性回归比一元线性回归的实用意义更大。

一般的情形,对于给定n个特征的示例$x=(x_1,x_2,\cdots,x_n)^T$,其中x_i是x在第i个属性上的取值,其多元线性模型可表示为:

$$f(x)=b+w_1x_1+\cdots+w_nx_n$$

用向量形式表示,可写成:

$$f(x)=b+w^{\mathrm{T}}x$$

在式中,b称为截距,$w_j(j=1,2,\cdots,n)$称为回归系数。参数w_j表示当其他回归变量$x_i(i\neq j)$保持不变的时,x_j每变化一个单位,响应变量y均值的变化期望值,也就是y平均变化多少。

下面我们来看看线性回归分析的具体实施。

任务实施

(一)一元线性回归分析实施

首先,创建 Python 文件,从 Windows 开始菜单打开 Anaconda 的 Jupyter Notebook (anaconda)。如图 1-2。

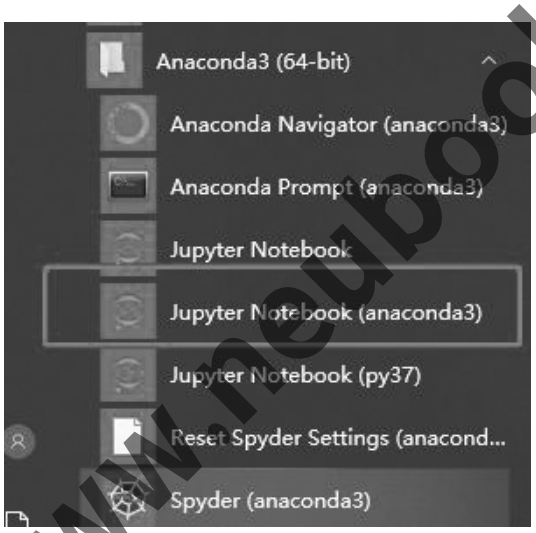

图 1-2 Jupyter Notebook(anaconda)

点选右上角“New”,选择“Python3”,创建 Python 文件,如图 1-3 所示,然后,可通过点选“Untitled”重命名为“1.一元线性回归.ipynb”,如图 1-4。

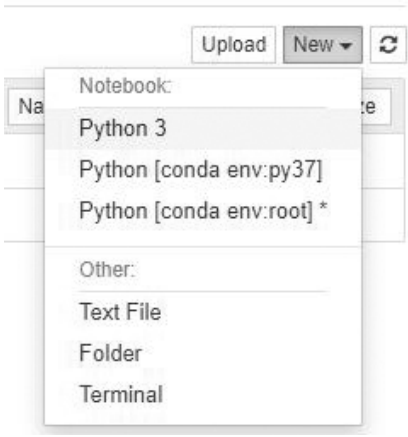

图 1-3 创建 ipynb 文件

图 1-4　重命名

一元线性回归分析实施具体步骤如下。

(1)打开文件,查看数据

利用 pandas 的 read_csv 方法读取指定位置的数据集“ex1data1.txt”文件,并将该文件中的数据保存到 data,使用 type()方法可查看 data 为 DataFrame 类型,使用 head()方法可查看数据集的前 5 行,如图 1-5。

```
1  import pandas as pd
2
3  path =  'data\ex1data1.txt'
4  # names添加列名, header用指定的行来作为标题,若原无标题且指定标题则设为None
5  data = pd.read_csv(path, header=None, names=['Population', 'Profit'])
6  data.head()
```

	Population	Profit
0	6.1101	17.5920
1	5.5277	9.1302
2	8.5186	13.6620
3	7.0032	11.8540
4	5.8598	6.8233

图 1-5　使用 pandas 读取 txt 文件

利用 Python 的 matplotlib 进行数据可视化,这里采用散点图观察数据的分布情况,得到的结果如图 1-6 所示,从中可以观察出人口(Population)与利润(Profit)呈现一定的线性关系。

matplotlib 是 Python 的主要绘图工具,plot 方法中 kind='scatter',表示做散点图,x 轴表示人口,y 轴表示利润。

从数据集中提取自变量 X 和因变量 y,如图 1-7 所示。

```
import matplotlib.pyplot as plt
data.plot(kind='scatter', x='Population', y='Profit', figsize=(8,5))
plt.show()
```

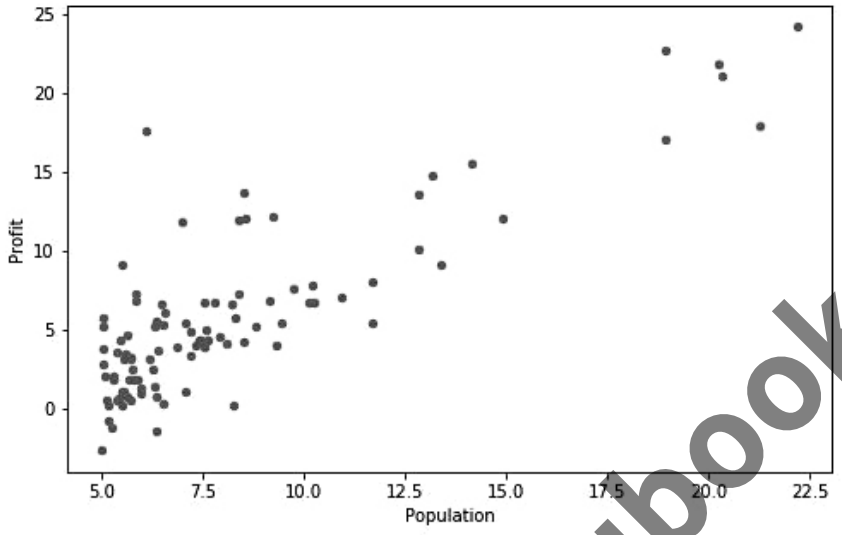

图 1-6 散点图

```
X = data.iloc[:,0:1]    # 第0列，即自变量
y = data.iloc[:,1:] # 取最后一列，即因向量
```

图 1-7 自变量与因变量提取

(2)数据集拆分

通过导入 train_test_split,实现数据拆分,如图 1-8。train_test_split 位于 sklearn 的 model_selection 中,形式为:

```
X_train, X_test, y_train, y_test = train_test_split(data, target, train_size=0.8, random_state=0)
```

参数解释:

- data:待划分样本数据;
- target:待划分样本数据的结果(标签);
- train_size:训练数据占样本数据的比例,若整数则样本数量;
- random_state:设置随机数种子,保证每次都是同一个随机数。若为 0 或不填,则每次得到数据都不一样。

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
                                                    random_state=42)
```

图 1-8　数据拆分

得到的 X_train，y_train 为训练集的自变量和因变量数据，X_test，y_test 为测试数据集，这里给出的拆分比例为 8:2。

(3)线性回归建模

建立模型，得到回归系数，如图 1-9，导入 linear_model 模块后，对 LinearRegression 实例化，得到线性回归实例化模型 LR，通过其 fit 方法实现训练集上的建模训练。

```
from sklearn import linear_model
LR = linear_model.LinearRegression()
# 基于训练数据，对线性回归模型进行训练
LR.fit(X_train, y_train)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

图 1-9　LinearRegression 线性回归建模

线性回归方法 LinearRegression 调用后返回一个线性回归模型，损失函数为误差均方函数。sklearn 中的 LinearRegression 的函数原型为：

```
class sklearn.linear_model.LinearRegression(fit_intercept=True,
                                            normalize=False,
                                            copy_X=True,
                                            n_jobs=1)
```

参数解释：

• fit_intercept：模型是否存在截距。

• normalize：模型是否对数据进行标准化(在回归之前，对 X 减去平均值再除以二范数)，如果 fit_intercept 被设置为 False 时，该参数将忽略。

• copy_X：默认 True，否则 X 会被改写。

• n_jobs：默认为 1，表示使用 CPU 的个数。当－1 时，代表使用全部 CPU。

LinearRegression 有 fit()方法，用于在训练集上训练模型，有属性 intercept_可供查看截距，有属性 coef_可供查看模型训练后得到的估计系数，如果获取的估计系数太大，说明模型有可能过拟合。

如图 1-10 所示，该回归系数 1.288 表明在给定人口数据下，如果城市人口每增加 1 个单位，对应的利润将增加 1.288。

```
print('intercept_:%.3f' % LR.intercept_)  # 截距项
print('coef_:%.3f' % LR.coef_)            # 回归系数(斜率)
```

```
intercept_:-4.732
coef_:1.288
```

图 1-10　回归系数

(4)评估回归模型

利用第 2 步拆分得到的测试数据集，在测试数据集上得到该模型的测试结果 y_test_pre，并将该测试结果与真实结果 y_test 进行比较，通过 LinearRegression 的 score 方法，得到在测试集上的精确度(取值范围 0～1)，1 表示无误差，0 表示无法完成拟合。如图 1-11 所示为测试数据的部分展示。

```
X_test.head()
```

	Population
62	21.2790
40	5.4069
93	5.3054
18	6.4296
81	5.1884

```
y_test.head()
```

	Profit
62	17.92900
40	0.55657
93	1.98690
18	3.65180
81	0.20421

图 1-11　测试数据集的前 5 个样本

LinearRegression 的 predict 方法以测试数据集的 X_test 作为输入，得到预测结果，记为 y_test_pre，其中图 1-12 的框中所示数据为图 1-11 前 5 个测试数据的预测输出。

```
y_test_pre = LR.predict(X_test)
print(y_test_pre)
```

```
[[22.66492686]
 [ 2.22914165]
 [ 2.09845748]
 [ 3.54589731]
 [ 1.94781662]
 [ 7.84508459]
 [19.67786014]
 [10.34198911]
 [ 2.62029289]
 [ 3.13453187]
 [ 3.75164441]
 [ 3.69821196]
 [ 2.73707175]
 [11.78402132]
 [ 5.34212868]
 [ 2.81226342]
 [ 2.33626404]
 [ 2.65029231]
 [ 1.78172541]
 [ 4.57115646]]
```

图 1-12　测试数据集上的预测结果

使用 sklearn 的 metrics 模块提供的方法，可以得到在测试集上的评估结果，如图 1-13 所示。

```
from sklearn.metrics import mean_squared_error, r2_score
print('Mean squared error: %.3f' % mean_squared_error(y_test, LR.predict(X_test)))
print('score: %.3f' % r2_score(y_test, LR.predict(X_test)))
```

```
Mean squared error: 15.709
score: 0.500
```

图 1-13　模型在测试集上的评估

mean_squared_error 方法是计算均方误差(MSE)，MSE 值越小越好；r2_score 方法是计算 R2 决定系数(拟合优度)，取值范围 0～1，越趋向于 1 表示模型越好，本任务中得到的值 0.5 偏小，预测效果一般。

(5)对比预测结果与实际结果

如图 1-14，从下面的对比来看，得到的该模型对数据的预测存在一定的偏差，解决这个问题，可以增加训练数据量。