基于 MindSpore 的鸢尾花分类

华为技术有限公司

华为技术有限公司

地址： 深圳市龙岗区坂田华为总部办公楼 邮编：518129

网址： http://e.huawei.com

1 实验介绍

机器学习分为监督学习、无监督学习、半监督学习、强化学习。监督学习是指利用一组已知类别的样本调整分类器的参数，使其达到所要求性能的过程，也称为监督训练或有教师学习。分类和回归是监督学习中的两种典型任务。

本章实验涉及线性回归、逻辑回归、KNN 回归等算法，通过不同算法的效果对比来加深对算法的理解。

本章实验难度为：初级。

- 初级实验：鸢尾花二分类实验实验目的

通过本章实验的学习，您将能够：

- 掌握使用 MindSpore 进行逻辑回归
- 掌握模型评估的方法

1.1 实验清单

实验	简述	难度	软件环境	开发环境
鸢尾花二分类预测（分类）	基于两个种类的鸢尾花数据，进行逻辑回归实验，实现鸢尾花的二分类预测	初级	MindSpore	ModelArts、PC 64bit

1.2 实验开发环境

- Mindspore-1.3

若选择在华为云 ModelArts 上快速搭建开发环境，可参考文末附录：ModelArts

2 鸢尾花二分类实验

2.1 实验介绍

2.1.1 简介

逻辑回归（Logistic Regression）是机器学习最经典的算法之一，与线性回归有很多不同，这两种回归都属于广义线性回归（Generalized Linear Regression）的范畴。逻辑回归具有如下特点：

- 逻辑回归对自变量分布没有要求；
- 因变量是离散型变量，即分类变量；
- 逻辑回归分析的是因变量取某个值的概率与自变量的关系。

本实验主要介绍使用 MindSpore 在 2 分类数据集上进行逻辑回归实验，分析自变量和因变量（概率）之间的关系，即求得一个概率函数。

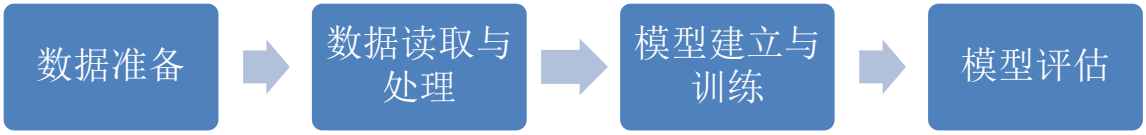
2.1.2 实验目的

- 了解逻辑回归的基本概念；
- 了解如何使用 MindSpore 进行逻辑回归实验。

2.2 实验环境要求

- MindSpore 1.3（MindSpore 版本会定期更新，本指导也会定期刷新，与版本配套）；
- 华为云 ModelArts：ModelArts 是华为云提供的面向开发者的一站式 AI 开发平台，集成了昇腾 AI 处理器资源池，用户可以在该平台下体验 MindSpore。环境搭建可参考文末附录。

2.3 实验总体设计



2.4 实验过程

2.4.1 数据准备

步骤 1　下载数据

Iris 数据集是模式识别最著名的数据集之一。数据集包含 3 类，每类 50 个实例，其中每个类都涉及一种鸢尾植物。 第一类与后两类可线性分离，后两类之间不能线性分离，所以本实验取前两类数据，做一个 2 分类数据集。

Iris 数据集的官网：[Iris Data Set](http://archive.ics.uci.edu/ml/datasets/Iris)。

- 方式一，从 Iris 数据集官网下载[iris.data 文件](http://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data)。
- 方式二，从华为云 OBS 中下载[iris.data 文件](https://share-course.obs.cn-north-4.myhuaweicloud.com/dataset/iris.data)。

步骤 2　上传数据到实验环境

在新建的 notebook 实验环境中，通过如图所示的“上传”按钮，然后选择自己本地已下载好的数据文件“iris.data”，将数据文件上传到实验环境中。

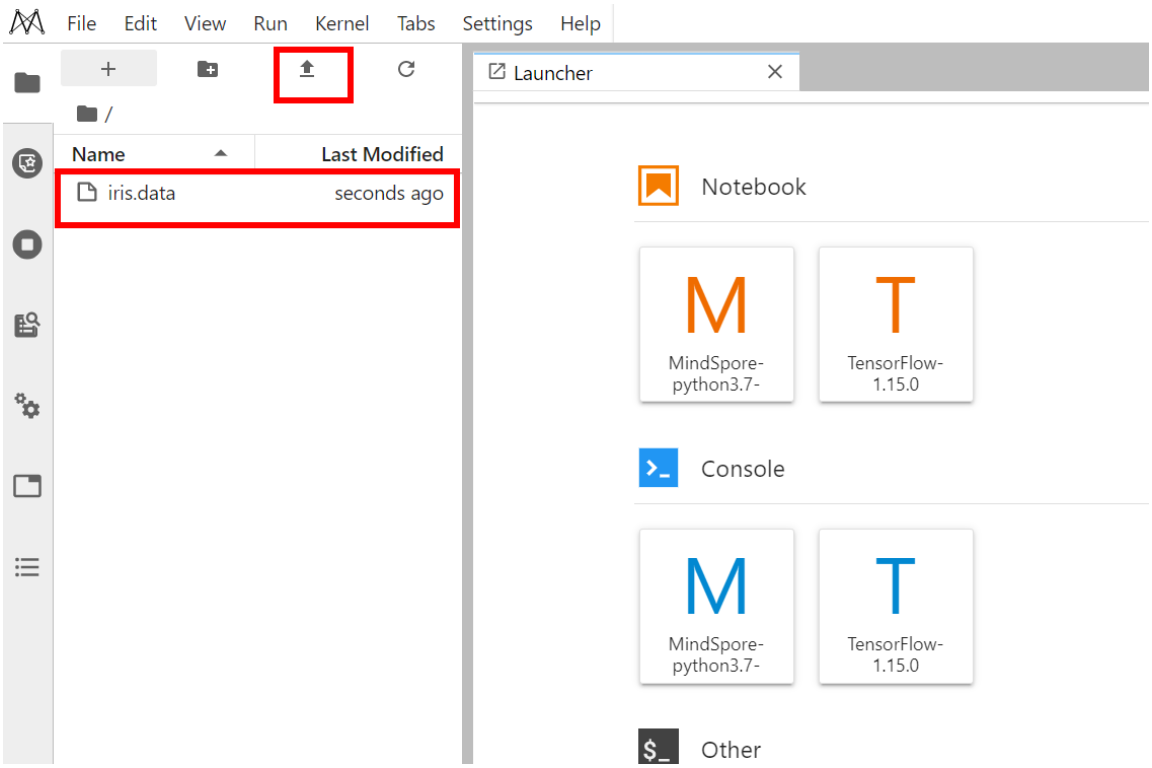


图2-1 上传数据集到实验环境

2.4.2 数据读取与处理

步骤 1 导入 MindSpore 模块和辅助模块

```
import os
# os.environ['DEVICE_ID'] = '6'
import csv
import numpy as np

import mindspore as ms
from mindspore import nn
from mindspore import context
from mindspore import dataset
from mindspore.train.callback import LossMonitor
from mindspore.common.api import ms_function
from mindspore.ops import operations as P

context.set_context(mode=context.GRAPH_MODE, device_target="Ascend")
```

步骤 2 读取 Iris 数据集，并查看部分数据

```
with open('iris.data') as csv_file:
    data = list(csv.reader(csv_file, delimiter=','))
print(data[40:60]) # 打印部分数据
```

输出：

```
[['5.0', '3.5', '1.3', '0.3', 'Iris-setosa'], ['4.5', '2.3', '1.3', '0.3', 'Iris-setosa'], ['4.4', '3.2', '1.3', '0.2', 'Iris-setosa'], ['5.0', '3.5', '1.6', '0.6', 'Iris-setosa'], ['5.1', '3.8', '1.9', '0.4', 'Iris-setosa'], ['4.8', '3.0', '1.4', '0.3', 'Iris-setosa'], ['5.1', '3.8', '1.6', '0.2', 'Iris-setosa'], ['4.6', '3.2', '1.4', '0.2', 'Iris-setosa'], ['5.3', '3.7', '1.5', '0.2', 'Iris-setosa'], ['5.0', '3.3', '1.4', '0.2', 'Iris-setosa'], ['7.0', '3.2', '4.7', '1.4', 'Iris-versicolor'], ['6.4', '3.2', '4.5', '1.5', 'Iris-versicolor'], ['6.9', '3.1', '4.9', '1.5', 'Iris-versicolor'], ['5.5', '2.3', '4.0', '1.3', 'Iris-versicolor'], ['6.5', '2.8', '4.6', '1.5', 'Iris-versicolor'], ['5.7', '2.8', '4.5', '1.3', 'Iris-versicolor'], ['6.3', '3.3', '4.7', '1.6', 'Iris-versicolor'], ['4.9', '2.4', '3.3', '1.0', 'Iris-versicolor'], ['6.6', '2.9', '4.6', '1.3', 'Iris-versicolor'], ['5.2', '2.7', '3.9', '1.4', 'Iris-versicolor']]
```

步骤 3　抽取样本

取前两类样本（共 100 条），将数据集的 4 个属性作为自变量 X。将数据集的 2 个类别映射为{0, 1}，作为因变量 Y。

```
label_map = {
    'Iris-setosa': 0,
    'Iris-versicolor': 1,
}

X = np.array([[float(x) for x in s[:-1]] for s in data[:100]], np.float32)
Y = np.array([[label_map[s[-1]]] for s in data[:100]], np.float32)
```

步骤 4　样本可视化

取样本的前两个属性进行 2 维可视化，可以看到在前两个属性上两类样本是线性可分的。

```
from matplotlib import pyplot as plt
%matplotlib inline
plt.scatter(X[:50, 0], X[:50, 1], label='Iris-setosa')
plt.scatter(X[50:, 0], X[50:, 1], label='Iris-versicolor')
plt.xlabel('sepal length')
plt.ylabel('sepal width')
plt.legend()
```

输出：

```
<matplotlib.legend.Legend at 0xffff5d6599d0>
```

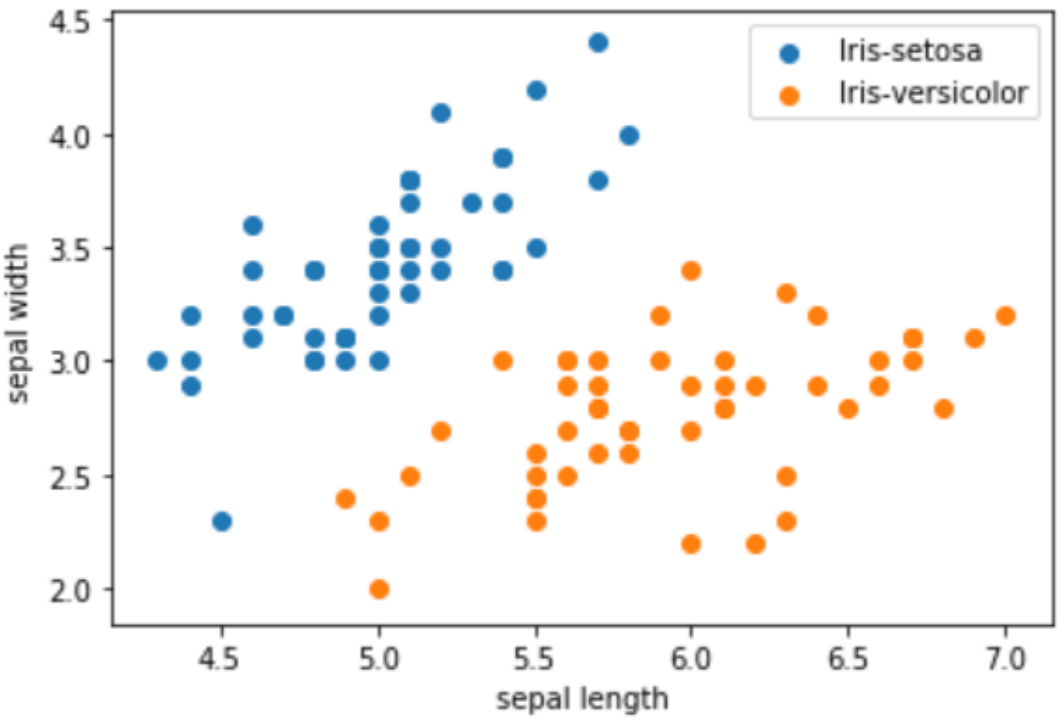


图2-2 样本数据可视化

步骤 5 分割数据集

将数据集按 8:2 划分为训练集和验证集：

```
train_idx = np.random.choice(100, 80, replace=False)
test_idx = np.array(list(set(range(100)) - set(train_idx)))
X_train, Y_train = X[train_idx], Y[train_idx]
X_test, Y_test = X[test_idx], Y[test_idx]
```

步骤 6 数据类型转换

使用 MindSpore 的 GeneratorDataset 接口将 numpy.ndarray 类型的数据转换为 Dataset：

```
XY_train = list(zip(X_train, Y_train))
ds_train = dataset.GeneratorDataset(XY_train, ['x', 'y'])
# ds_train.set_dataset_size(80)
ds_train = ds_train.shuffle(buffer_size=80).batch(32, drop_remainder=True)
```

2.4.3 模型建立与训练

步骤 1 可视化逻辑回归函数

逻辑回归常用的联系函数是 Sigmoid（S 形函数），Sigmoid 函数如下图所示，可以将连续值映射到{0, 1}，同时也是单调可微的。

```
coor_x = np.arange(-10, 11, dtype=np.float32)
```

```
coor_y = nn.Sigmoid()(ms.Tensor(coor_x)).asnumpy()
plt.plot(coor_x, coor_y)
plt.xlabel('x')
plt.ylabel('p')
```

输出：

```
Text(0, 0.5, 'p')
```

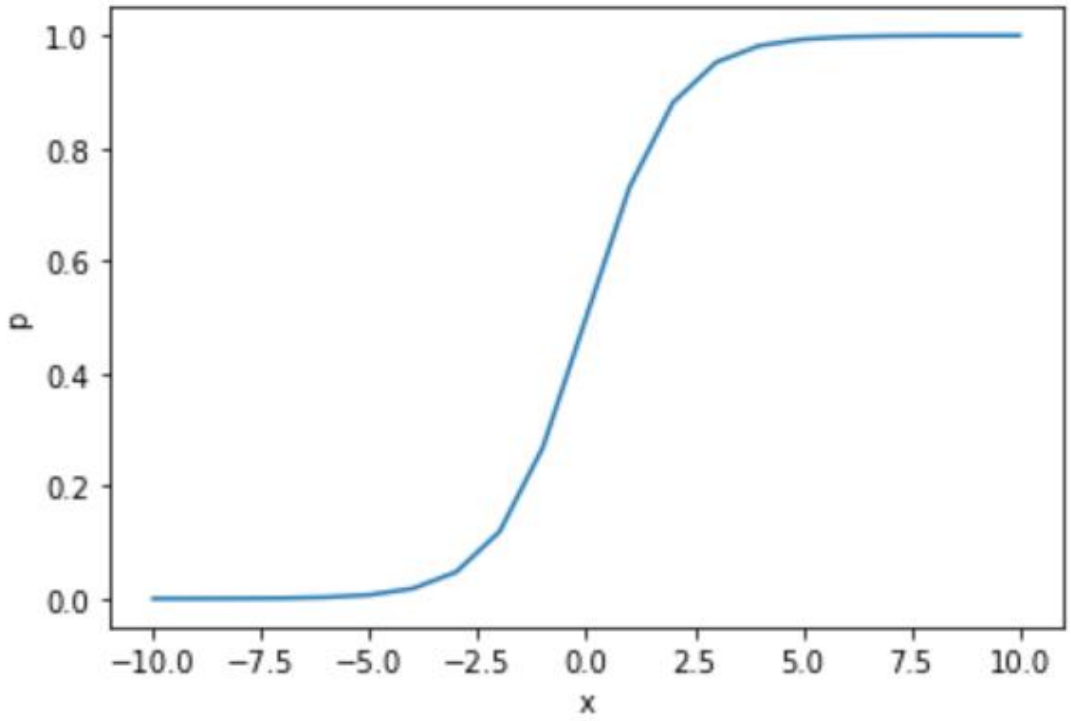


图2-3 Sigmoid 函数图示

步骤 2 建模

使用 MindSpore 提供的 nn.Dense(4, 1)算子（https://www.mindspore.cn/api/zh-CN/0.2.0-alpha/api/python/mindspore/mindspore.nn.html#mindspore.nn.Dense)作为线性部分，其中(4, 1)表示每个样本的输入是含 4 个元素的向量，输出是含 1 个元素的向量，即 W 是 1x4 的矩阵。算子会随机初始化权重 W 和偏置 b。使用 SigmoidCrossEntropyWithLogits 算子(https://www.mindspore.cn/api/zh-CN/0.3.0-alpha/api/python/mindspore/mindspore.ops.operations.html?#mindspore.ops.operations.SigmoidCrossEntropyWithLogits)作为非线性部分：

对于每个样本 N_i，模型的计算方式如下：

$$z_i = wx_i + b$$

$$p_i = sigmoid(z_i) = \frac{1}{1 + e^{-z_i}}$$

$$loss = \frac{1}{n}\sum_{i=1}^{n}(y_i * \ln(p_i) + (1 - y_i) * \ln(1 - p_i))$$

其中，x_i是 1D Tensor（含 4 个元素），z_i是 1D Tensor（含 1 个元素），y_i是真实类别（2 个类别{0, 1}中的一个），p_i是 1D Tensor（含 1 个元素，表示属于类别 1 的概率，值域为[0, 1]），loss 是标量。

```
# 自定义 Loss
class Loss(nn.Cell):
    def __init__(self):
        super(Loss, self).__init__()
        self.sigmoid_cross_entropy_with_logits = P.SigmoidCrossEntropyWithLogits()
        self.reduce_mean = P.ReduceMean(keep_dims=False)
    def construct(self, x, y):
        loss = self.sigmoid_cross_entropy_with_logits(x, y)
        return self.reduce_mean(loss, -1)

net = nn.Dense(4, 1)
loss = Loss()
opt = nn.optim.SGD(net.trainable_params(), learning_rate=0.003)
```

步骤 3　模型训练

使用 2 分类的 Iris 数据集对模型进行几代（Epoch）训练：

代码：

```
model = ms.train.Model(net, loss, opt)
model.train(10, ds_train, callbacks=[LossMonitor(per_print_times=ds_train.get_dataset_size())],
dataset_sink_mode=False)
```

输出：

```
    epoch: 1 step: 2, loss is 0.6513498
    epoch: 2 step: 2, loss is 0.5780734
    epoch: 3 step: 2, loss is 0.5097989
    epoch: 4 step: 2, loss is 0.4643281
    epoch: 5 step: 2, loss is 0.41714883
    epoch: 6 step: 2, loss is 0.39717123
    epoch: 7 step: 2, loss is 0.3307726
    epoch: 8 step: 2, loss is 0.29960862
    epoch: 9 step: 2, loss is 0.29403976
    epoch: 10 step: 2, loss is 0.27250308
```

2.4.4 模型评估

然后计算模型在测试集上精度，测试集上的准确率达到了 1.0 左右，即逻辑回归模型学会了区分 2 类鸢尾花。

代码：

```
x = model.predict(ms.Tensor(X_test)).asnumpy()
pred = np.round(1 / (1 + np.exp(-x)))
correct = np.equal(pred, Y_test)
acc = np.mean(correct)
```

```
print('Test accuracy is', acc)
```

输出：

```
Test accuracy is 1.0
```

2.5 实验小结

本实验使用 MindSpore 实现了逻辑回归，用来解决 2 分类问题。在 Iris 数据集上进行训练后，所得的模型可以很好的表示每个样本类别 y 和属性 x 的关系。

2.6 创新设计

请使用 Softmax 函数作为概率映射函数，对完整的 Iris 数据集实现多分类任务。

3 附录：ModelArts 开发环境搭建

- ModelArts 平台：Mindspore-1.2

步骤 1 进入 ModelArts

在华为云主页搜索 Modelarts，点击“AI 开发平台 ModelArts”中的“进入控制台”。

图3-1

步骤 2 选择训练作业

选择“北京四”地区，在左侧下拉框中点击“开发环境”中的“Notebook”：

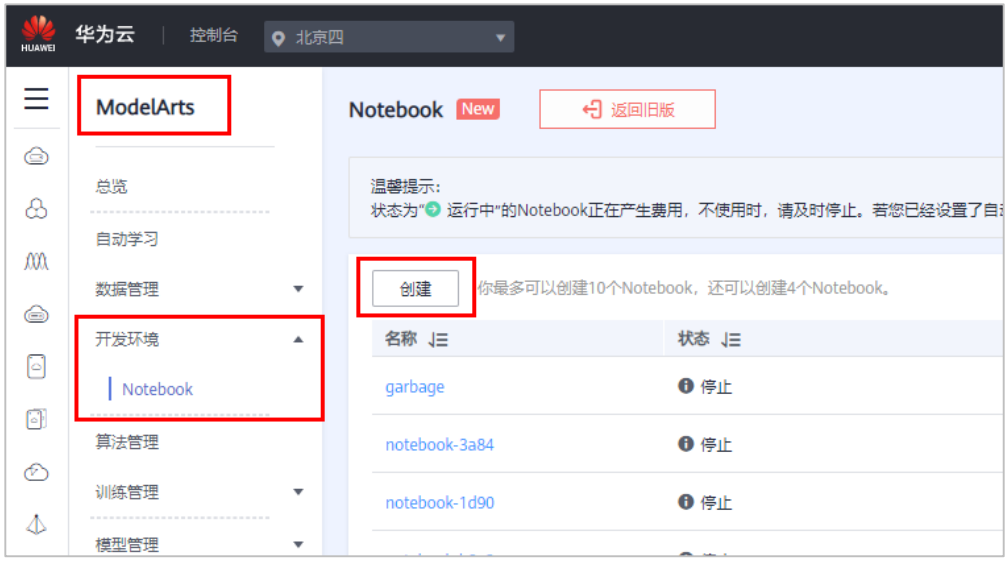


图3-2

步骤 3 创建 Notebook

点击创建按钮来创建一个新的 Notebook，选择如下配置：

- 名称：自定义。
- 工作环境：Ascend+ARM 算法开发和训练基础镜像。
- 存储配置：默认存储。

点击“下一步”，确认规格如下后选择提交：

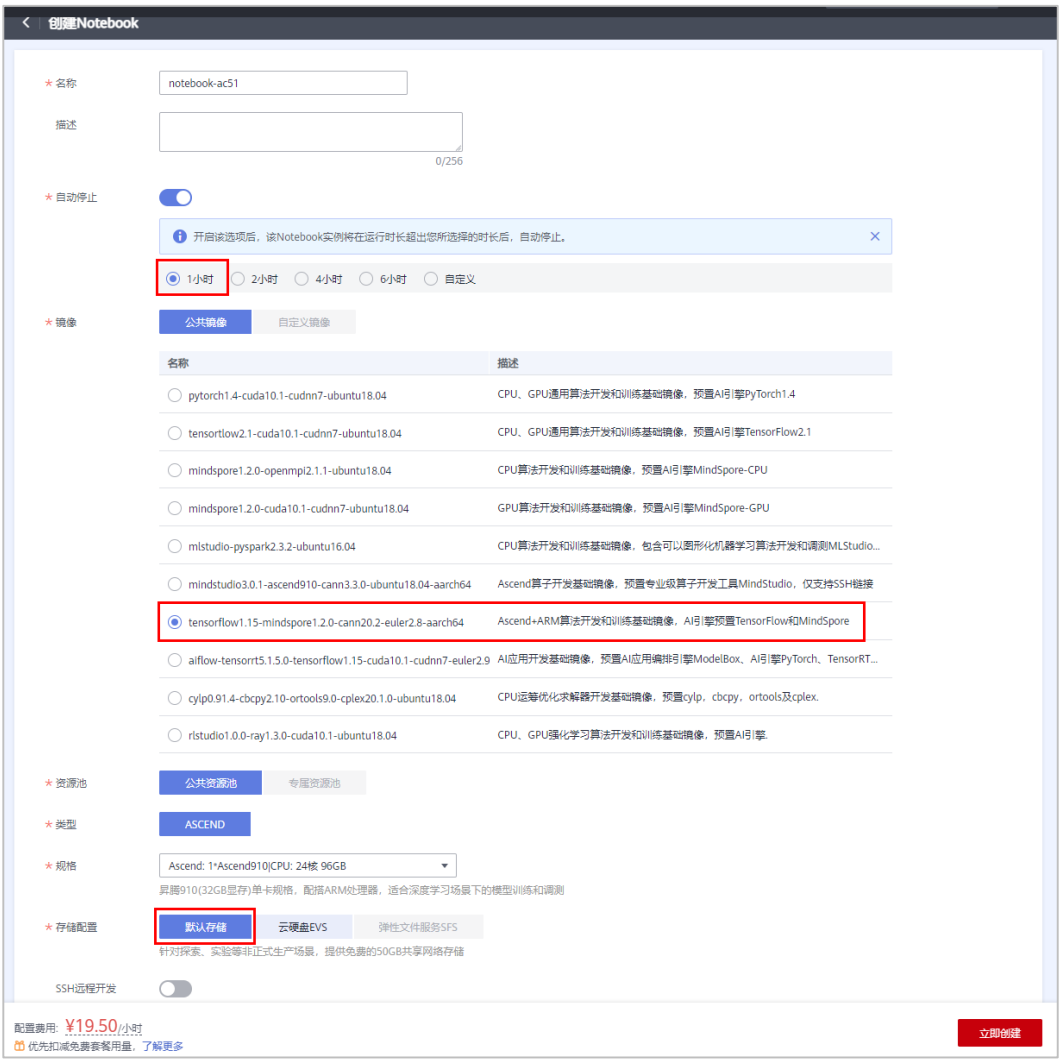


图3-3

步骤 4　启动 Notebook 进入开发环境

当 Notebook 状态变为“运行中”时，点击右侧“打开”按钮打开 Notebook。打开后选择右侧“MindSpore-python3.7-aarch64”按钮，进入 Notebook 环境：

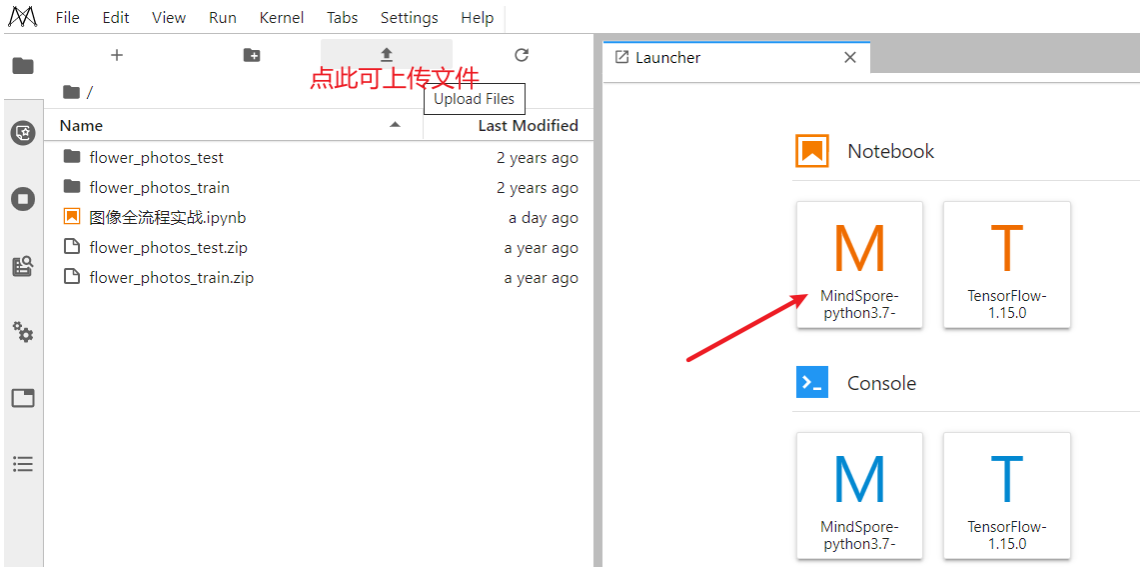


图3-4

步骤 5　停止实验环境

试验完成之后请及时停止实验环境，避免资源浪费，如下图：

图3-5 停止实验环境