实验一 Regression

一、 说明

- 实验采用 jupyter notebook, 请填写完代码后提交完整的 ipynb 文件
- 文件命名规则：班级_姓名_ML2018_HW1.ipynb，如计科 1701_张三_ML2018_HW1.ipynb
- 提交方式：采用在线提交至：

  http://pan.csu.edu.cn:80/link/AF79DFF26A99902D63879E10B2113044
- 实验提交截止日期：2018.10.7 24：00

二、 实验内容

本实验在一个具体的应用中逐步地指导用户训练出一个 Regression 模型。实验使用的训练方法有梯度下降法、AdaGrad 法和正规方程法。

梯度下降法是机器学习中常用的用于训练模型方法。梯度下降法通过使参数往负梯度方向移动的方法不断降低模型的损失函数值，训练得到拟合训练数据的模型。本实验会实现梯度下降法来解决具体问题，并直观展示梯度下降的过程。实验中会体现特征的归一化、学习率和梯度下降的迭代次数对训练的影响。

AdaGrad 法是梯度下降的优化算法，可以提高梯度下降算法的训练速度。本实验会实现 AdaGrad 法来解决具体问题。

正规方程法是求线性回归模型的另一种方法，通过具体公式可以直接求出线性回归模型的参数。本实验会实现正规方程法来解决具体问题。

三、 实验目标

- 掌握搭建 python 的开发环境，并能够使用 numpy 工具。
- 掌握特征的归一化处理。
- 掌握梯度下降法的具体过程，并能够实现梯度下降法，能够根据具体情况选择适当的超参数——学习率、迭代次数。
- 掌握 AdaGrad 的具体过程，并能够 AdaGrad。

- 掌握正规方程的公式，并能够使用正规方程计算回归模型。

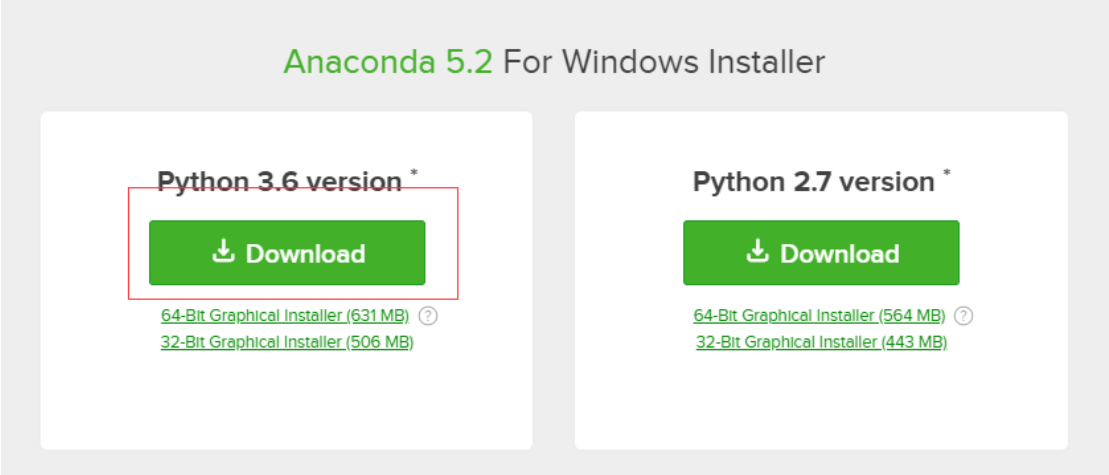
四、 实验操作步骤

1. 搭建 python 环境

本实验需要用到的 python 环境包括

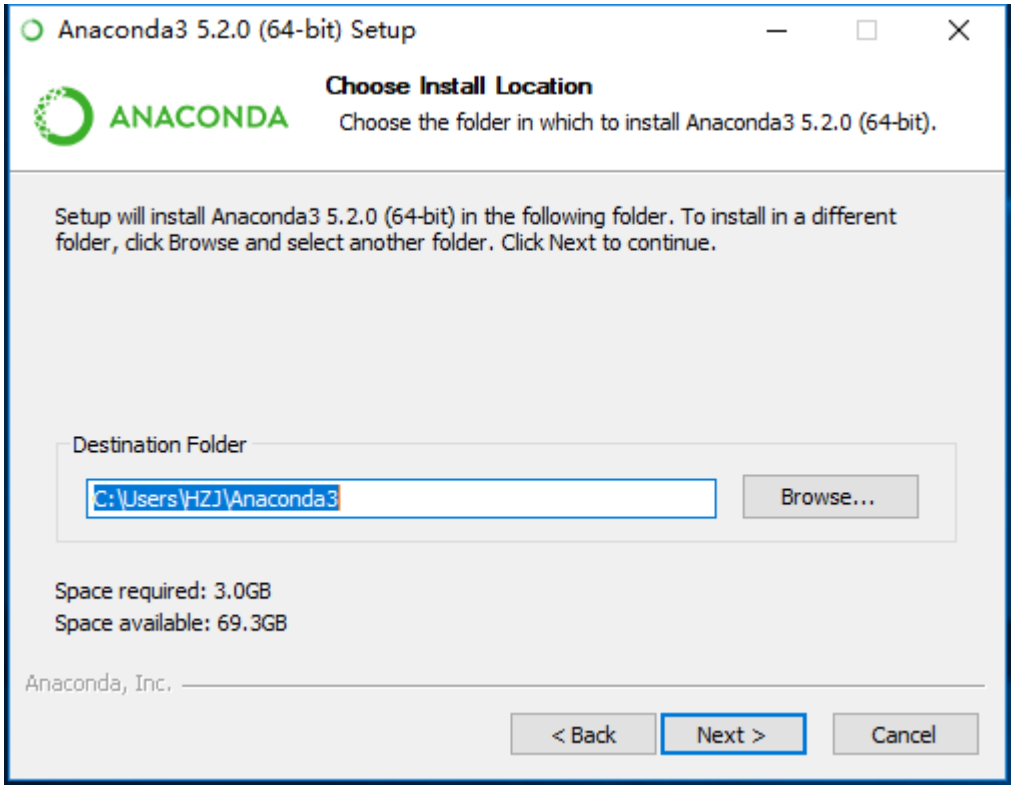
名称	版本
Python	3.6.5
Numpy	1.14.3
matplotlib	2.2.2
jupyter	1.0.0

推荐安装 anaconda，下载页面：https://www.anaconda.com/download/。
选择 Python 3.6 版本下载：（根据操作系统不同选择不同版本，这里演示 anaconda 在 windows 系统的安装过程）



双击运行下载好的文件

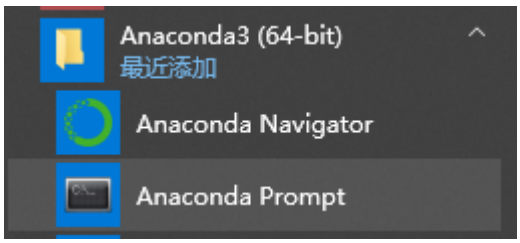
在安装界面一路选择 next，安装目录自己选择，比如“C:\Users\HZJ\Anaconda3”。



安装完毕后，Python、Numpy、matplotlib 和 jupyter 都安装成功。

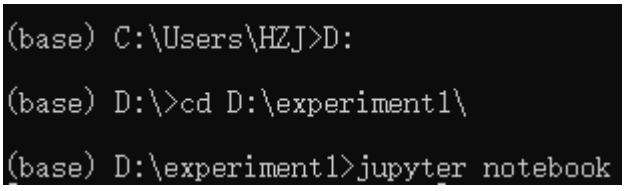
2. 启动 jupyter notebook

接着启动 jupyter notebook。**在 Windows 系统中**，启动目录选择 Anaconda> Anaconda Prompt



在弹出的命令行中 cd 到实验目录，打开 jupyter notebook，比如实验目录在”D:\experiment1\”，输入下列命令：

```
D:
cd D:\experiment1\
jupyter notebook
```



然后弹出 jupyter notebook 的页面。

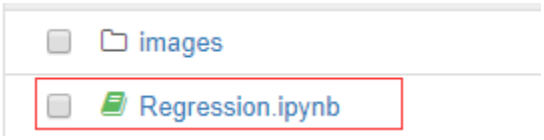
在 Linux 系统中，打开一个终端，在确保 python 环境已经安装好并且 jupyter 命令能够

在终端执行之后，把终端的活动目录切换为实验目录，比如实验目录在”~/experiment1/”，然后启动 jupyter notebook，具体如下：

```
cd ~/experiment1/
jupyter notebook
```

然后弹出 jupyter notebook 的页面。

在弹出的 jupyter notebook 页面，可以看到有文件 Regression.ipynb（如下图），单击打开它。



在新的页面中可以看到具体的实验内容。

在 Regression.ipynb 中有许多任务，每个任务需要实现相应代码，有” ### START CODE HERE ###”的标记说明这里是要填写代码，” ### END CODE HERE ###”说明到这里终止。

```
### START CODE HERE ###

X_train = None

### END CODE HERE ###
```

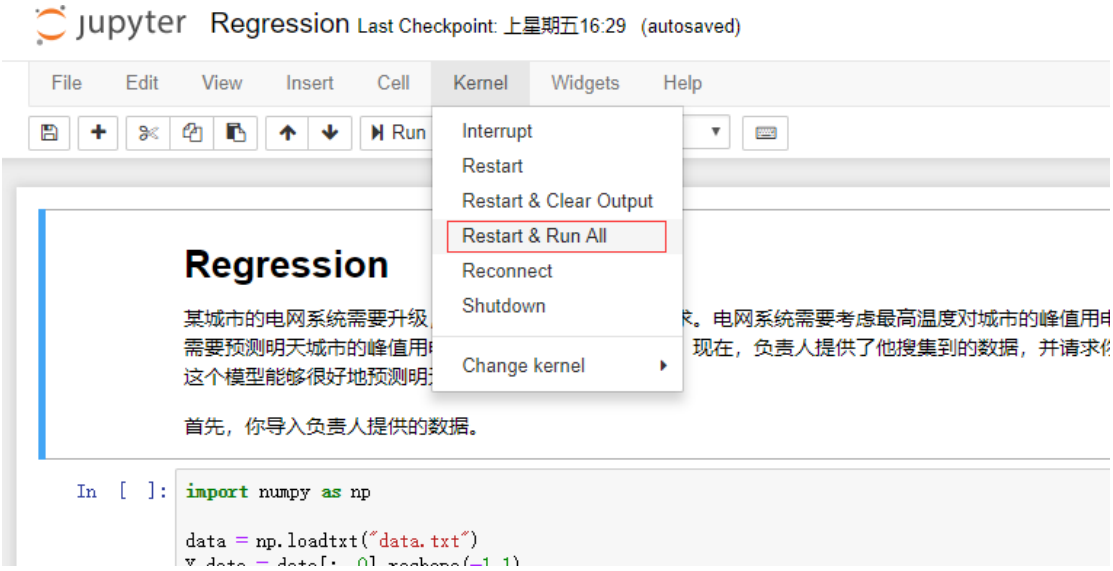
填写完代码并运行这个任务，会有一些结果输出，比如

```
]: import numpy as np
   import os

   data = np.loadtxt("data.txt")
   # data 数据第一列为人口信息
   X_data = data[:, 0].reshape(-1,1)
   # data 数据第三列为城市峰值用电量
   y_data = data[:, 2].reshape(-1,1)

   print("X shape: ", X_data.shape)
   print("y shape: ", y_data.shape)
```

```
X shape:  (100, 1)
y shape:  (100, 1)
```

在 jupyter 中每个分隔的区域称为 cell，一个 cell 可以是 python 代码块，或者是文字说明块。运行 cell 的方式可以是单击页面头部的菜单栏的运行按钮，或者使用快捷键”shift+enter”。实验的任务可能包含多个代码 cell 和文字说明 cell，所以运行任务可能要按照顺序运行多个 cell，然后才能看到输出结果，具体输出位置请看输出代码（例如”print”）所在位置。（注意，新打开的.ipynb 文件要从第一个 cell 开始运行）

做完实验后，最好重新运行一遍（Restart & Run All）（如下图的操作）。



3. 完成实验任务

任务 1 生成特征向量。

```
X_train shape:  (100, 2)
X_train[:5,:] =  [[ 1.   39.44]
 [ 1.   37.22]
 [ 1.   33.33]
 [ 1.   26.11]
```

```
 [ 1.   21.67]]
```

任务 2 初始化模型的参数。

```
theta_init shape:  (2, 1)
theta_init =  [[0.5488135 ]
 [0.71518937]]
```

任务 3 实现代价函数。

```
loss_init =  148.87165656055174
```

任务 4 实现 gradient 函数。

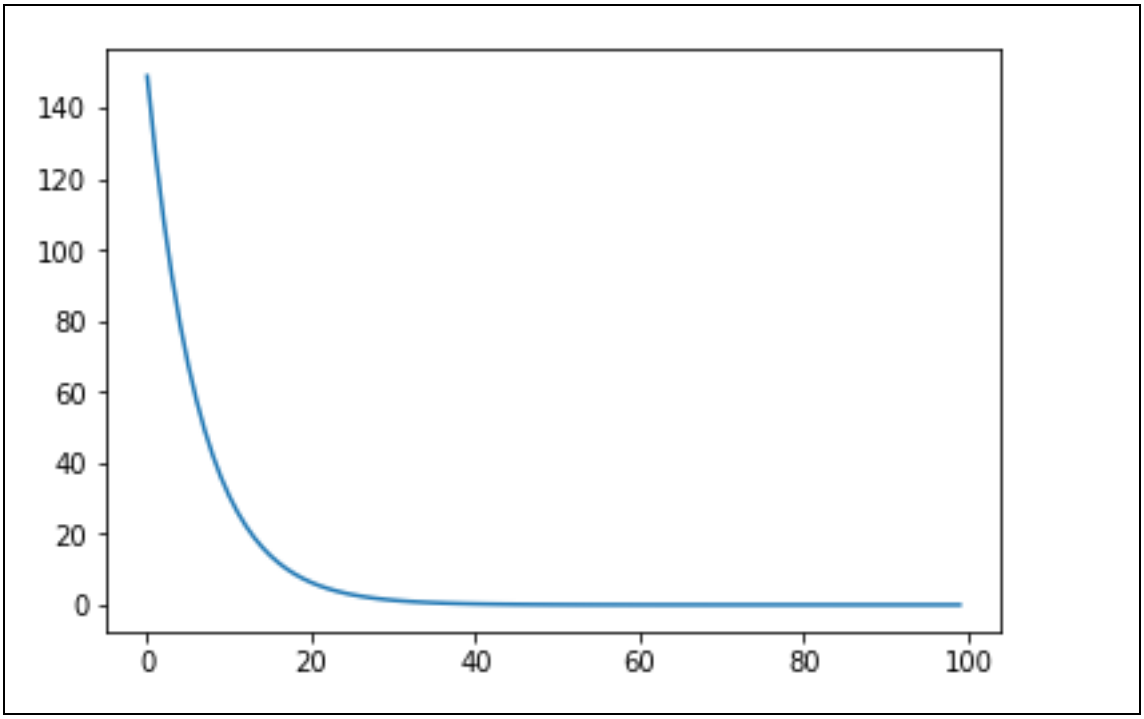
```
gradients_init shape :  (2, 1)
gradients_init =  [[ 16.32138507]
 [475.51600546]]
```

任务 5 实现 update_parameters 函数。

```
theta_one_iter =  [[ 0.38559965]
 [-4.03997069]]
```

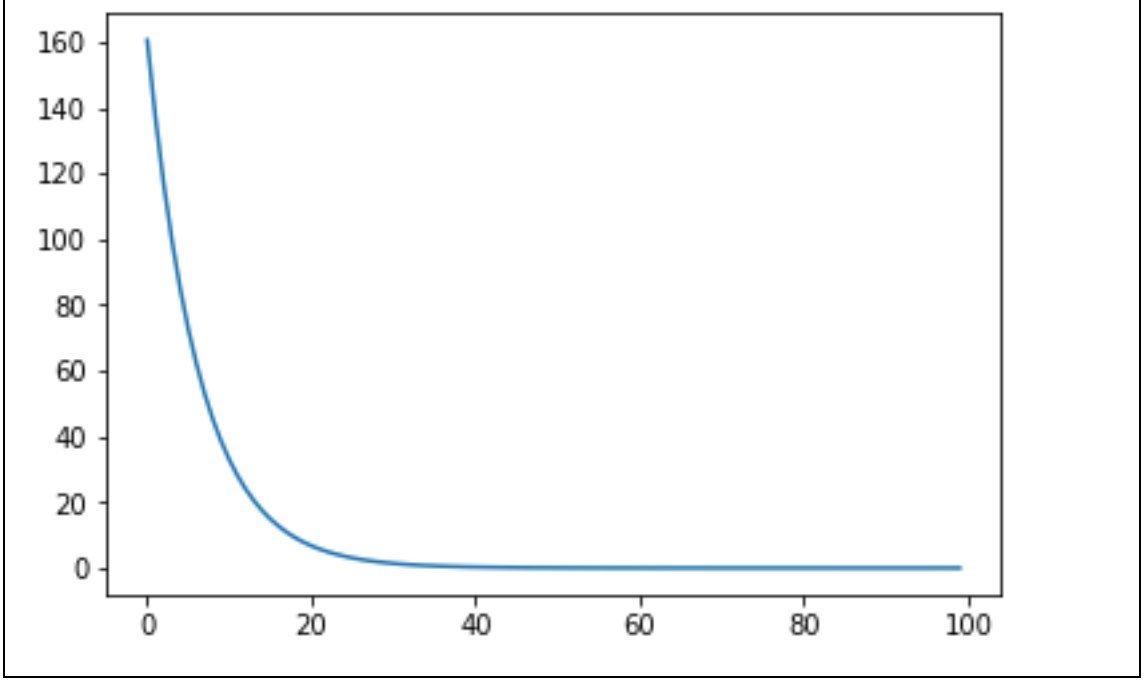
任务 6 实现梯度下降函数。

```
theta =  [[0.52759494]
 [0.09026695]]
loss =  0.06366695038230548
```



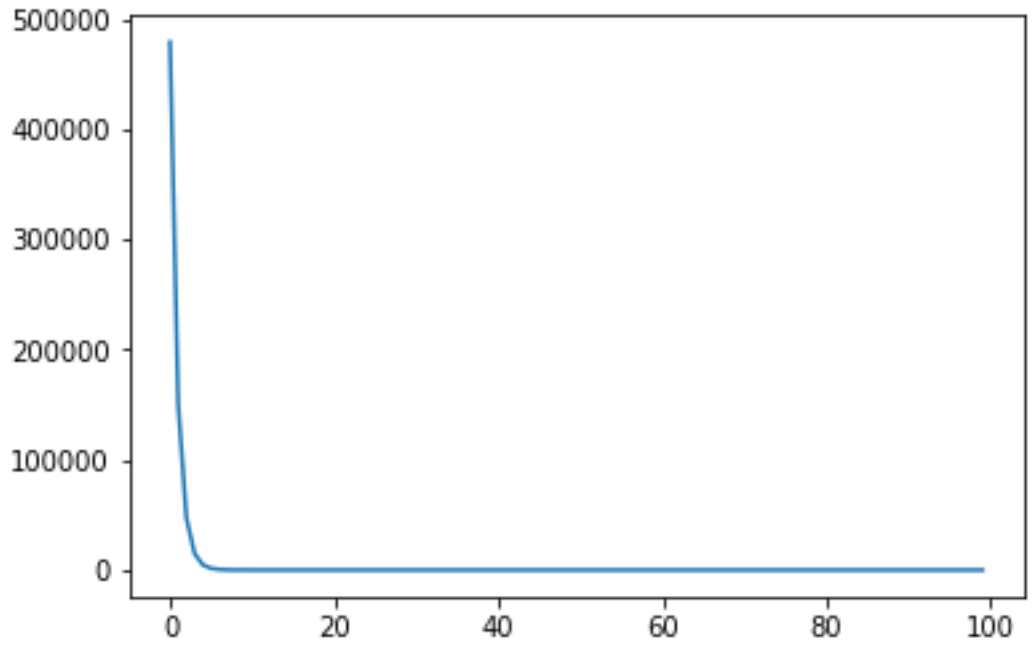
任务 7 训练一个多变量回归模型。

```
theta =  [[0.52617755]
 [0.06744549]
 [0.57591696]]
loss =  0.0869443412049637
```



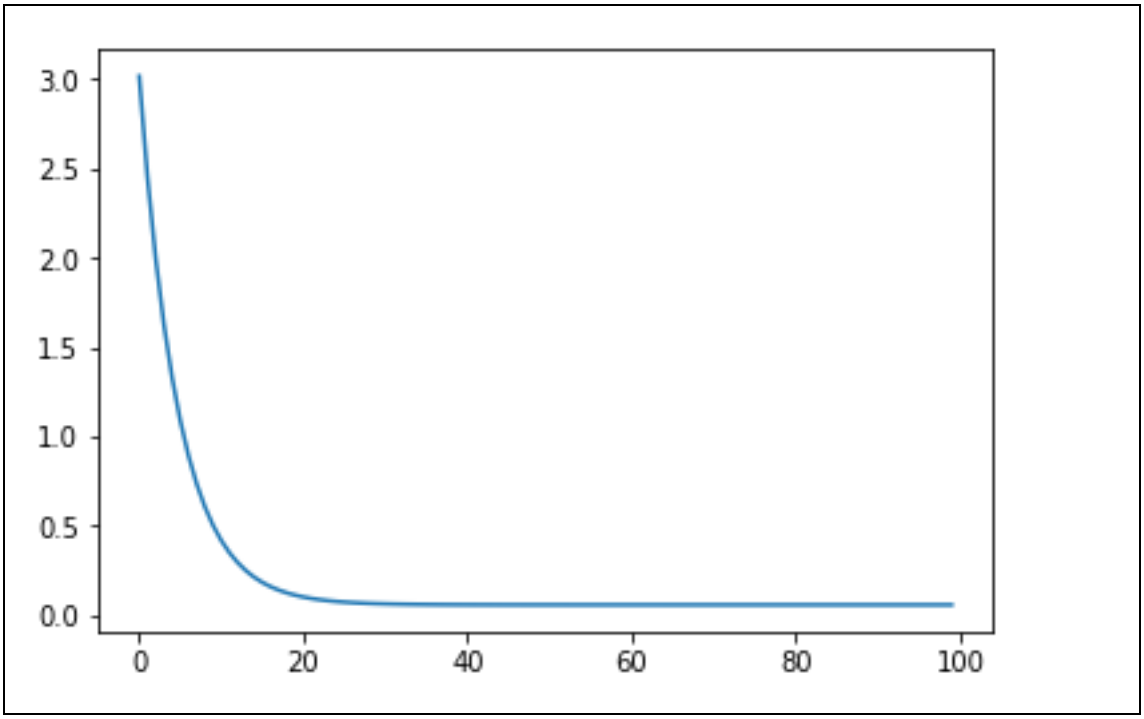
任务 8 训练一个多项式模型。

```
theta =  [[ 0.54791482]
 [ 0.69399861]
 [ 0.60155744]
 [ 0.51529176]
 [-0.12983411]
 [ 0.0924052 ]]
loss =  39.916368596778184
```



任务 9 对数据进行零均值单位方差归一化处理。

```
mu =  [26.1164  1.1463]
sigma =  [8.81399938 0.36134376]
theta =  [[2.9055374 ]
 [0.71437892]
 [0.00597022]]
loss =  0.060170919181121205
```

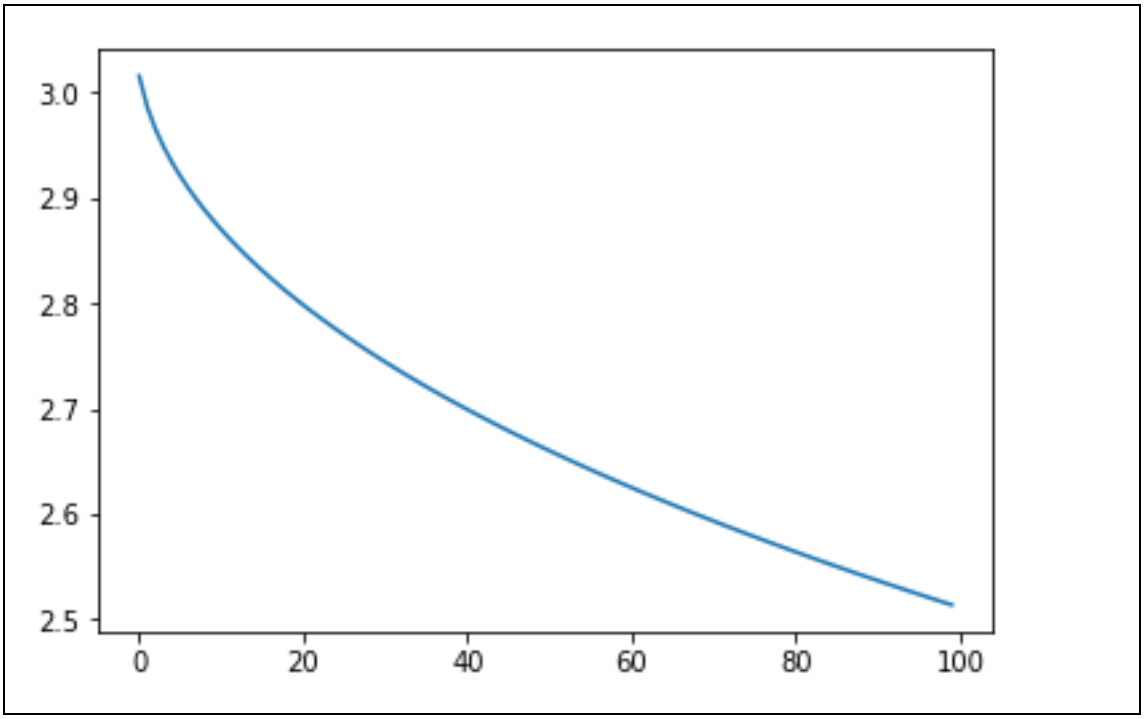


任务 10 实现 AdaGrad 的参数更新函数。

```
theta_one_iter_adagrad =  [[0.5588135 ]
 [0.70518939]
 [0.59276338]]
```

任务 11 实现 `gradient_descent_ada` 函数。

```
theta =  [[0.73194447]
 [0.68347052]
 [0.42872122]]
loss =  2.5139427812263593
```



任务 12 实现正规方程。

```
theta =  [[2.9056    ]
 [0.71439293]
 [0.0059491 ]]
```

任务 13 使用正规方程求得的参数对新数据进行预测。

```
predict =  [[3.22290432]]
```