Python作业第四周

项目发起人:`[江柳]`
项目发起时间:`[2017.10.22]`

1.课程阅读

Python Numpy Tutorial

本次作业目标:
本次作业同样是实现第三周作业的任务。不过为了强化Numpy和Matplotlib库的使用，本章所有计算过程要求
全部使用矩阵计算完成，在要求你实施的代码中`不允许`出现`For`循环。

涉及内容:

- Linear Regression with multiple variables
- 损失函数的矩阵计算
- 函数梯度的矩阵计算

- 预测值的矩阵计算
- 损失曲线的显示

注:

- 本次作业全部代码在`ex4.py`文件中完成

2.作业布置

2.1 数据载入

文本`dataset_4.txt`中有两列数据，第一列表示加数x_1，第二列表示被加数x_2。

尝试补全`load(filename)`函数代码，使该函数能读取`dataset_4.txt`，并返回numpy.array类型`X, y`。

完成后直接运行`ex4.py`文件，输出结果如下:

```
Loading Data ...
The shape of the variable X is:  (1024, 2)
Expected shape of the variable X is (1024, 2)
The shape of the variable y is:  (1024, 1)
Expected shape of the variable y is (1024, 1)
X[0] = [ 941.  136.]
Expected X[0] = [ 941.  136.]
y[0] = [ 1077.]
Expected Y[0] = [ 1077.]
```

2.2 特征缩放

使用Mean normalization方法进行特征缩放:

$$x_i = \frac{x_i - u_i}{S_i}$$

其中:
u_i为所有x_i的均值或期望。
S_i为标准差(standard deviation)

尝试补全`featureScaling(vals)`函数代码，使该函数能够对`vals`根据上式子完成归一化。

完成后直接运行`ex4.py`文件，输出结果如下:

```
Feature Scaling ...
(1024, 2)
(1024, 1)
X_norm[0] = [ 1.45964021 -1.3017674 ]
Expected X_norm[0] = [ 0.42792509 -0.37489605]
y_norm[0] = [ 0.12507497]
Expected y_norm[0] = [ 0.02610716]
```

备注:

> numpy.mean(X, axis=..., keepdims=True)可以用来计算X的期望
> numpy.std(X, axis=..., keepdims=True)可以用来计算X的标准差

更多资料:
numpy.mean
numpy.std

2.3 损失计算

我们假设数据满足模型:

$$h_\theta(X) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

最接近现实情况的模型，我们需要衡量什么叫最接近现实情况，在这里我

们定义一个损失函数用来衡量模型与现实情况的近似程度。

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)^2$$

其中m是样本的个数，x^i和y^i是第i个样本，$J(\theta)$表示损失大小。

尝试补全`computeCost(X, y, theta)`函数代码，使该函数能够根据参数返回损失数值。

注:

> 在调用该函数时，我们传递进去的X_norm已经添加了一列全1的x_1。P132行。

完成后直接运行`ex4.py`文件，输出结果如下:

```
Testing costFunction ...
With theta =
[[ 0.]
 [ 0.]
 [ 0.]]
Cost computed = 0.500000
Expected cost computed = 0.500000
With theta =
[[ 1.]
 [ 2.]
 [ 3.]]
Cost computed = 3.914376
Expected cost computed = 3.914376
```

2.4 梯度下降算法

现在我们已经有了损失函数了，求最接近现实情况的模型，就是求使$J(\theta)$最小的θ_0和θ_1。这里已经变成一个参数优化问题了，解参数优化问题的方

法很多，可以自行了解，这里选用随机梯度下降算法`(Gradient Descent)`。

- 随机梯度下降算法
  - 计算J的梯度
  - 沿梯度反方向更新参数，重复此过程，直到损失函数收敛。

即:

$$\theta_0 = \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i)$$

$$\theta_1 = \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i)x^i$$

$$\theta_2 = \theta_1 - \alpha \frac{2}{m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i)x^i$$

注:

- θ_0、θ_1和θ_2要同时变化。

尝试补全`gradientDescent(X, y, theta, alpha, num_iters)`函数代码，使该函数返回模型对应的`theta`，以及每次迭代过程中记录下俩的损失`J_log`.

完成后直接运行`ex4.py`文件，输出结果如下:

```
Runing Gradient Descent ...
iter{100} cost = 0.000000
....
iter{1500} cost = 0.000000
Theta =
[[ -3.47135278e-18]
 [  7.16327079e-01]
 [  7.07119287e-01]]
Expected Theta approximate=
[[-3.47135278e-18]
[  7.16327079e-01]
[  7.07119287e-01]]
type(J_log) =  <class 'list'>
Expected type(J_log) = <class 'list'>
```

2.5 预测

现在我们要预测[12.0, 21.0], [123.0, 321.0]两个样本对应的y。

尝试补全`predictVals(X, y, X_test, theta)`函数，使其返回正确的预测值.

注:

> 不要忘记了特征缩放，和还原

完成后直接运行`ex4.py`文件，输出结果如下:

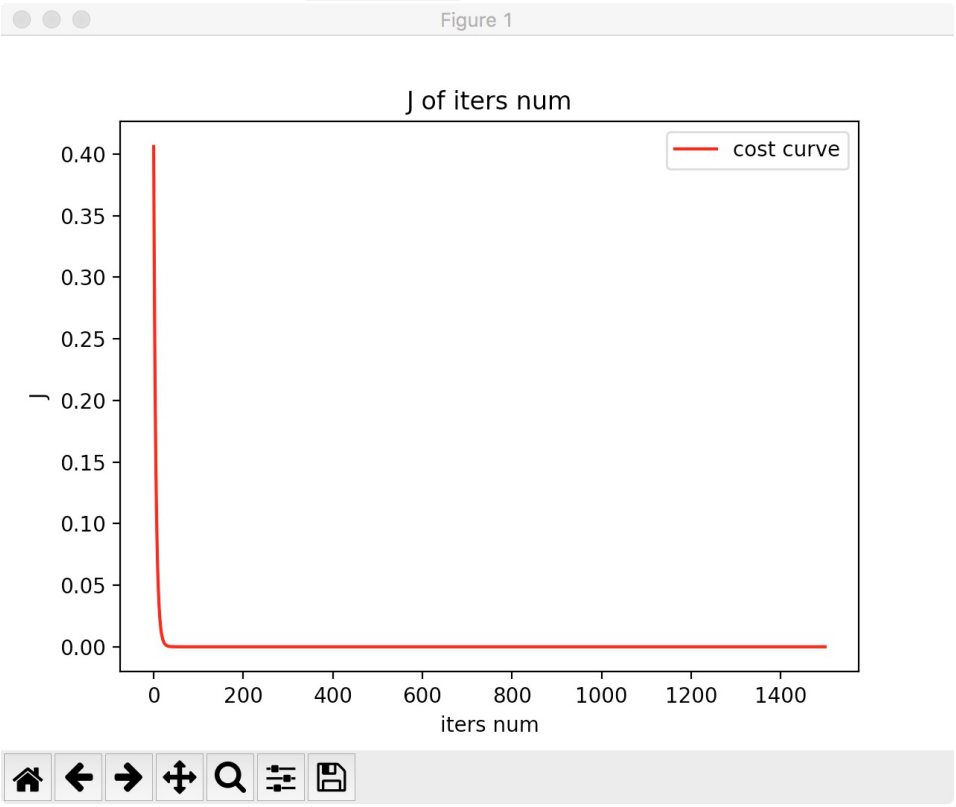
```
Predict values
predict =  [[  33.]
 [ 444.]]
Expected predict approximate= [[  33.]            [ 444.]]
```

2.6 显示损失曲线

根据J_log列表中的内容显示损失曲线。

尝试补全plotCost(J_log)函数。使其能够正确显示图线，坐标轴名字，标题，和铭牌。

完成后直接运行`ex4.py`文件，输出结果如下:



3.Deadline

- 2017.11.27(下周一)上午12:00截止
- 有不懂得可以随时Google或者找我问

4.提交方法

- 邮件发送到:`[root@oopy.org]`
- 邮件标题:`(姓名全拼)lecture4`
- 作业以邮件附件形式发送