山东大学 计算机科学与技术 学院

机器学习（双语） 课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号：201705130113 | 姓名：黄瑞哲 | | 班级：计科17.3 |
| 实验题目：线性回归 | | | |
| 实验学时：2 | | 实验日期： 2019.9.13 | |
| 实验目的： | | | |
| 硬件环境：  Intel Core i5-8300H @ 2.3GHz | | | |
| 软件环境：  Windows10 Pro 1903  Anaconda 4.7  Python 3.7 | | | |
| 实验步骤与内容： | | | |
| 结论分析与体会： | | | |

附录：程序源代码

*import* numpy *as* np  
*import* matplotlib.pyplot *as* plt  
*from* mpl\_toolkits.mplot3d *import* Axes3D  
  
  
# 标准化dt  
*def* scale(*dt*):  
 mu = np.mean(*dt*, axis=0)  
 sigma = np.std(*dt*, axis=0)  
 *return* (*dt* - mu) / sigma  
  
  
# 计算损失函数  
*def* j\_val(*X*, *Y*, *theta*):  
 hx = np.dot(*X*, *theta*)  
 *return* np.dot((hx - *Y*).transpose(), hx - *Y*) / (2 \* *X*.shape[0])  
  
  
# 梯度下降  
*def* gradient\_descent(*X*, *Y*, *theta*, *learning\_rate*=0.07, *max\_iteration*=1500):  
 # g = ((XO - Y)' \* (X))'  
 cost = np.zeros((*max\_iteration*, 1))  
 *for* i *in* range(*max\_iteration*):  
 cost[i] = j\_val(*X*, *Y*, *theta*)  
 hx = np.dot(*X*, *theta*)  
 theta = *theta* - (*learning\_rate* / *X*.shape[0]) \* (np.dot((hx - *Y*).transpose(), *X*)).transpose()  
 *return theta*, cost  
  
  
*def* task5():  
 x = np.loadtxt('data/ex1\_2x.dat')  
 y = np.loadtxt('data/ex1\_2y.dat').reshape(-1, 1)  
 m = x.shape[0]  
 # 将x标准化  
 x = scale(x)  
 x = np.hstack((np.ones((m, 1)), x))  
 m, n = x.shape  
 Theta, cost = gradient\_descent(x, y, np.zeros((n, 1)), learning\_rate=0.15, max\_iteration=50)  
 print(Theta, cost)  
 # plot3  
 plt.figure('Iteration')  
 plt.plot(np.linspace(0, 50, 50), cost)  
 plt.xlabel('number of iterations')  
 plt.ylabel('Cost J')  
 plt.show()  
  
  
*if* \_\_name\_\_ == "\_\_main\_\_":  
 x = np.loadtxt('data/ex1\_1x.dat').reshape(-1, 1)  
 y = np.loadtxt('data/ex1\_1y.dat').reshape(-1, 1)  
 m = x.shape[0]  
 x = np.hstack((np.ones((m, 1)), x))  
 m, n = x.shape  
 Theta, cost = gradient\_descent(x, y, np.zeros((n, 1)))  
 y1 = np.dot(np.array([1, 3.5]), Theta)  
 y2 = np.dot(np.array([1, 7]), Theta)  
 print("X=3.5\tY=%f" % y1)  
 print("X=7.0\tY=%f" % y2)  
 learn\_y = np.dot(x, Theta)  
 # plot1  
 plt.figure('Linear Regression')  
 plt.plot(x[:, 1], y, '.')  
 plt.plot(x[:, 1], learn\_y)  
 plt.xlabel('Age in years')  
 plt.ylabel('Height in meters')  
 plt.show()  
  
 # plot2  
 fig2 = plt.figure('J value')  
 theta\_0 = np.linspace(-3, 3, 100)  
 theta\_1 = np.linspace(-1, 1, 100)  
 j\_vals = np.zeros((100, 100))  
 *for* i *in* range(100):  
 *for* j *in* range(100):  
 j\_vals[i][j] = j\_val(x, y, np.array([theta\_0[i], theta\_1[j]]).reshape(-1, 1))  
 theta\_0, theta\_1 = np.meshgrid(theta\_0, theta\_1)  
 # surf  
 # ax = Axes3D(fig2)  
 # ax.plot\_surface(theta\_0, theta\_1, j\_vals)  
 plt.contour(theta\_0, theta\_1, j\_vals)  
 plt.xlabel('theta\_0')  
 plt.ylabel('theta\_1')  
 plt.show()  
  
 task5()