**ML2016\_HW2\_report** 姓名: 徐有慶 學號: r05922162

1. **Logistic regression function**

**Step1.** 將spam\_train.csv中的資料使用矩陣來儲存，形成矩陣X，其中，spam\_train.csv所提供的57個features全部用上，並加上一行(column)全部為1的行向量給bias使用，所以每一列共有58維。並將每筆資料的label存到矩陣Y\_head。

**Step2.** 初始一個矩陣W，其中，每個元素的範圍為0~0.01，當作初始的權重。

**Step3.** W\*X^T = Z，並將Z丟到sigmoid function求得Y

**Step4.** (Y-Y\_head)\*X = gradients，其中的元素即為每個權重的gradient。

**Step5.** Gradient descent

**Step6.** Y中元素大於0.5則設為1，反之設為0，則，其中yi，y\_headi。

**Step7.** 重複Step3~6直到error <= 280

**Step8.** 使用spam\_test.csv預測資料

**Primary code:**

def training(self):

weights = self.weights

x = self.train\_set

z = weights.dot(x.getT())

y\_head = self.y\_head

y = self.\_sigmoid(z)

gradients = (y - y\_head).dot(x)

self.past\_gradients.append(gradients)

self.\_gradientDescent(gradients)

return self.\_error(y, y\_head)

def \_gradientDescent(self, gradients):

learn\_rate = 0.1

sigma\_past = np.matrix(np.zeros(self.features\_dim))

for past in self.past\_gradients:

sigma\_past = sigma\_past + np.power(past,2)

for i in range(self.weights.size):

self.weights[0,i] = self.weights[0,i] - learn\_rate \* gradients[0,i] / math.sqrt(sigma\_past[0,i])

1. **Describe your another method, and which one is best**

另一方法使用Probabilistic generative model，機率的分布使用Gaussian Distribution。

**Step1.** 將spam\_train.csv中的資料使用矩陣來儲存，並依照每筆資料的label分別存入class1及class0當中，若label為1則存入class1，反之存入class0，其中，spam\_train.csv所提供的57個features全部用上，所以class1及class0中每一列共有57維。

**Step2.** 直接計算出、、Σ

**Step3.** 使用spam\_test.csv預測資料

**Primary code:**

1. **Discussion**

Logistic regression的部分利用對training set的正確率來停止，並在gradient descent的部份使用adagrad來加速收斂的速度，初始learning rate設為0.1。Probabilistic generative model的部分，若Σ為singular matrix則算其pseudo inverse來處理例外狀況。Logistic regression在leaderboard public set上的分數大約在0.92~0.933之間，使用Probabilistic generative model的話則在0.87左右，明顯地，Logistic regression在此資料上表現較佳。下表為不同training set的大小對Probabilistic generative model在test set上結果的影響。

|  |  |
| --- | --- |
| Training set size | Private set score |
| 500筆 | 0.85333 |
| 1000筆 | 0.86333 |
| 2000筆 | 0.85667 |
| 3000筆 | 0.86000 |
| 4000筆 | 0.86000 |

結果顯示在此資料上使用Probabilistic generative model做訓練，training set的大小對結果影響不大。