Machine Learning 2016 Homework 1

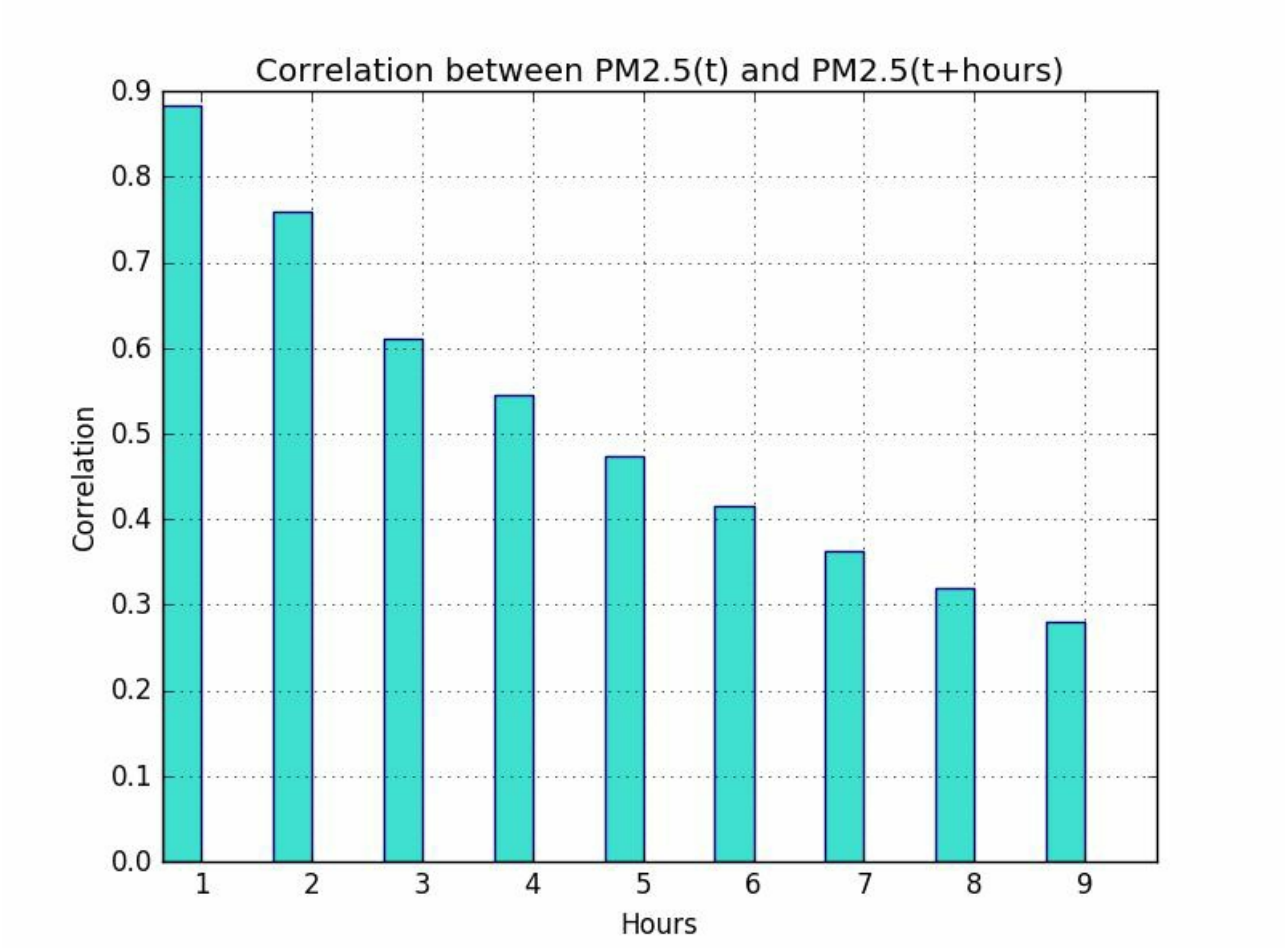
1. Gradient descent computation code

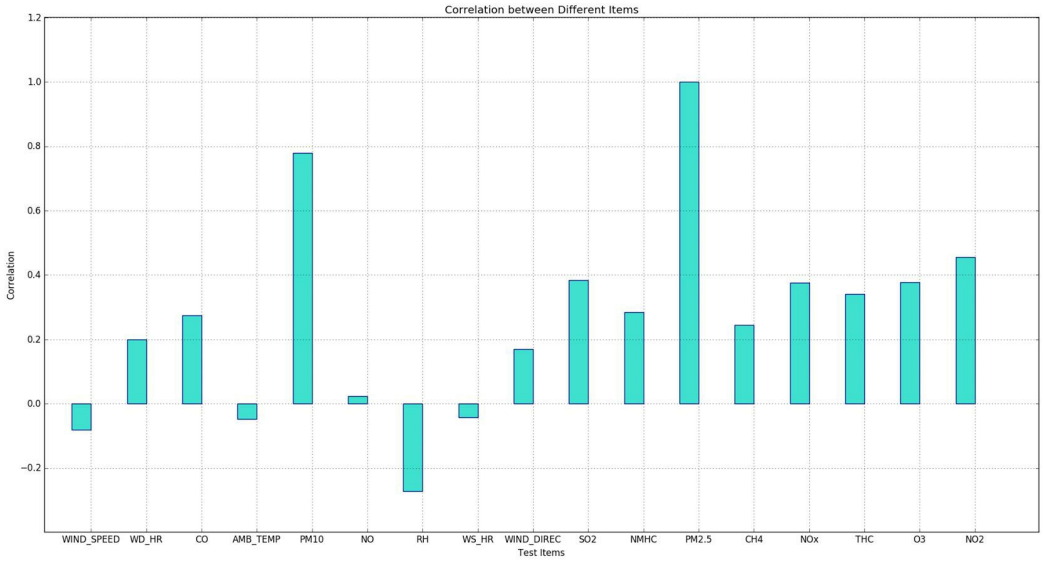
```
for k in range(train_iterations):
    # Compute gradient descent(with/without regularization)
    for i in range(num_features):
        gradsum = 0.0
        for j in range(num_inputs):
            gradsum += (output[j] - np.dot(weight, input_matrix[j])) *
(input_matrix[j][i])
        gradient[i] = (gradsum * (-2) + 2 * reg * weight[i]) / num_inputs
        grad_sqr_sum[i] += gradient[i] ** 2

    # Update the paramaters, using adaptive learning rate (adagrad)
    weight = weight - rate * gradient #/ (grad_sqr_sum ** 0.5)
```

2. Method description

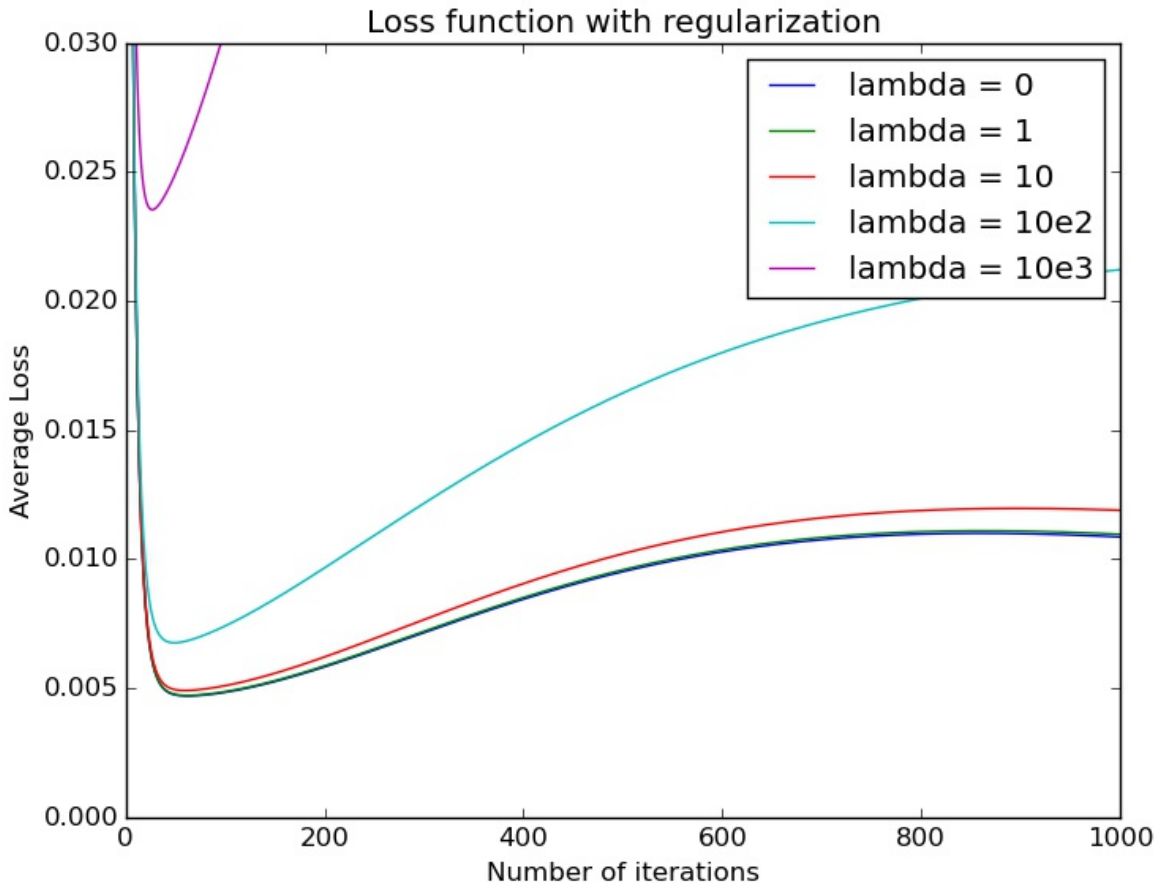
曾經嘗試過許多不同的feature組合，例如前九個小時所有的污染粒子分佈值、前九個小時的PM2.5 + PM10值(因為它與PM2.5的關聯性較大)、前n個小時的PM2.5值(n = 1~9)等等。最後以前九個小時的PM2.5值作為feature，能夠在public data set獲得最高的分數。





3. Regularization

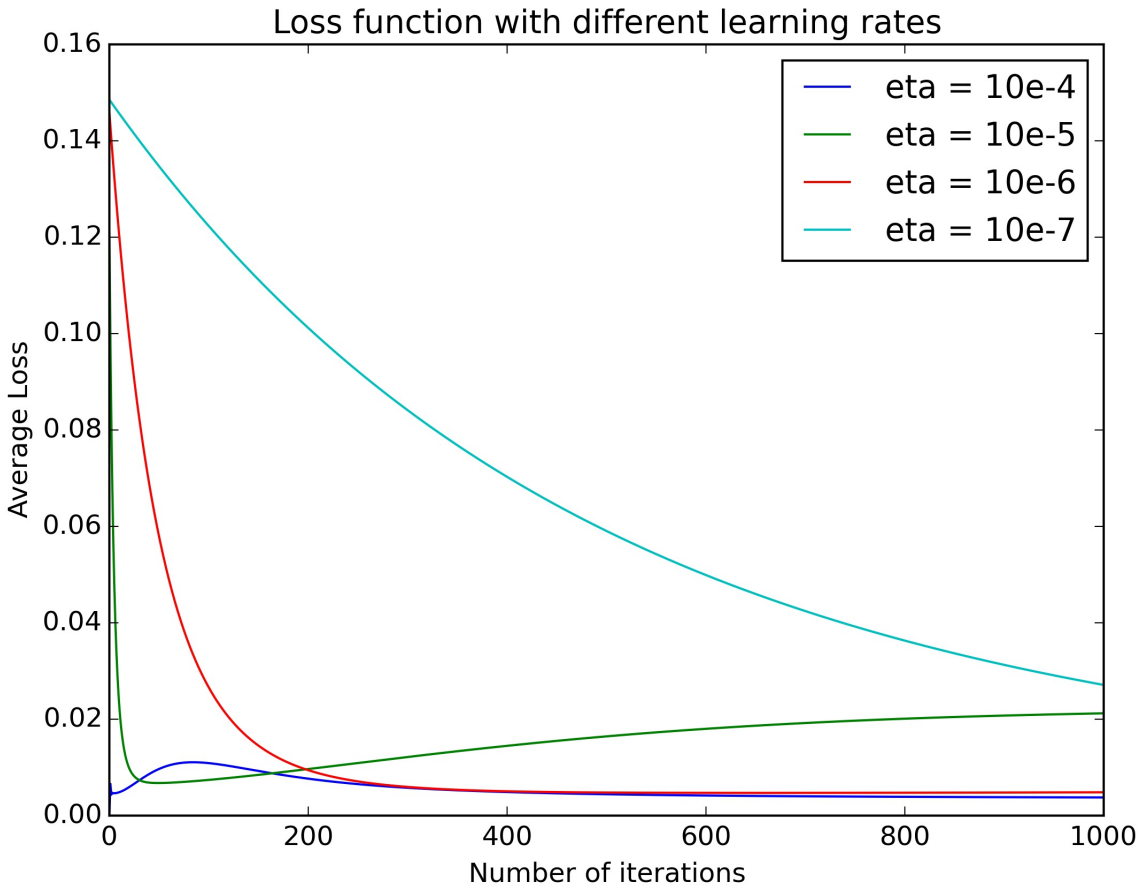
我嘗試以learning rate = 10e-5, regularization參數λ = 0, 1, 10, 100, 1000, 10000進行training，但預測結果並沒有多大差別。以下是regularization對training error的影響



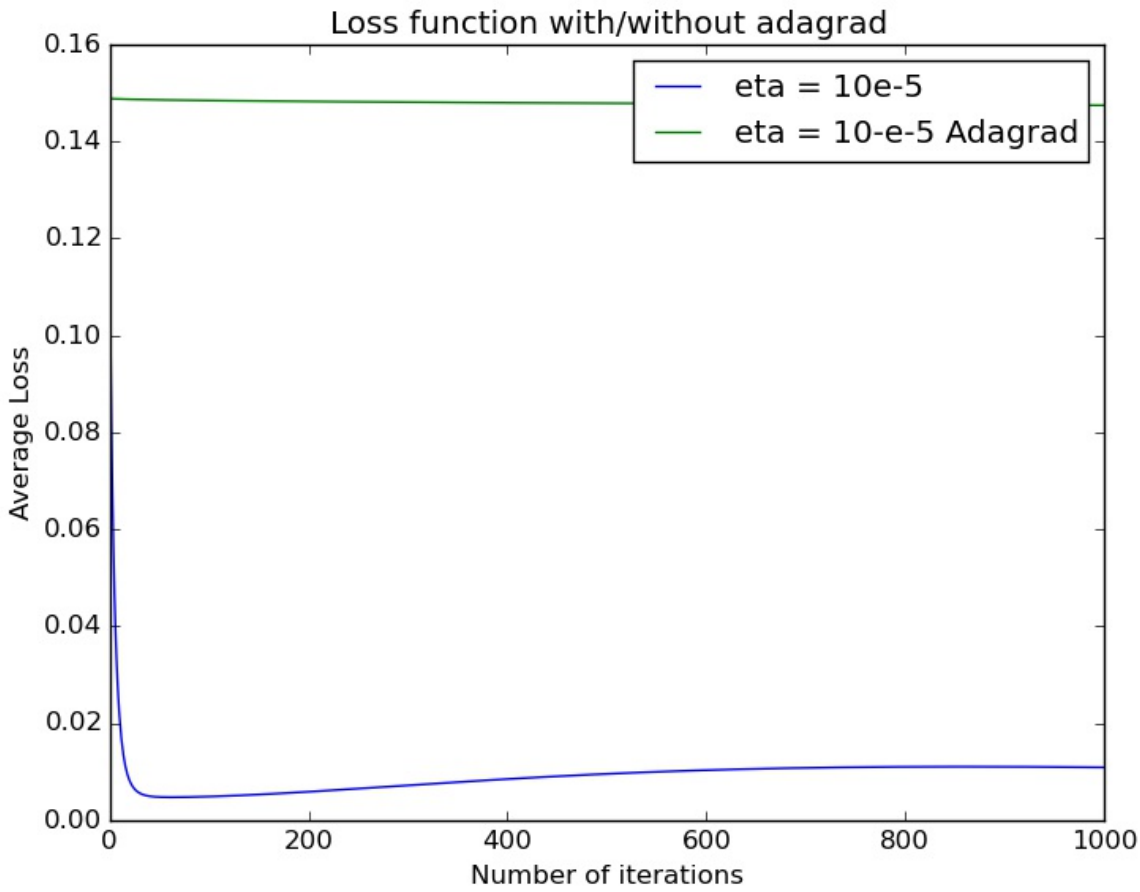
--

4. Different learning rates

Learning rate的取捨對error收斂的速度造成影響，如圖所示，越大的learning rate能夠在越前面的gradient descent iteration就取得較小的error。但是learning rate >= 10e-3會造成error發散至無限大的情形。



另外，使用Adagrad會因為一開始gradient絕對值過大而造成error收斂速度變慢（如圖所示），反而跟預期的效果相反。



Discussion

這是第一次學習操作machine learning，學到一些經驗可供未來參考：

1. feature selection：在選擇feature之前可先計算feature與output的關聯性，以選擇較適當的feature，並將結果視覺化以便觀察。
2. 使用adagrad的時機：為了避免第一次gradient平方值所佔的比例過大，可以先以一般的gradient descent取得參數之後，再使用adagrad