CS 189: Homework 7

William Guss
26793499
wguss@berkeley.edu

May 2, 2016

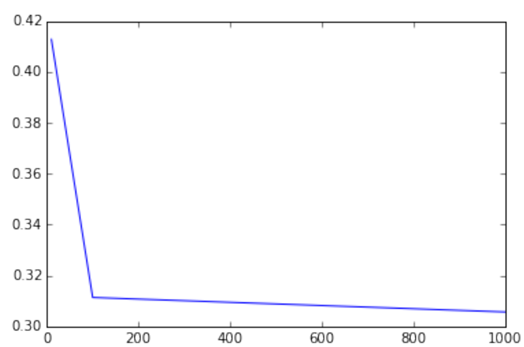
1. **KMeans**

   (a) After running K-Means Figure 1 is a sample plot of each class. Furthermore running K-means with different initializations leads to different loss; the classes are distributed differently each time.

2. **Joke Recommender System**

   (a) *Warm Up.* We tried the average value reccomendation system with some success! We got around 37% error as seen in the iPython notebook.

   (b) *k-NN.* We tried using k-NNS and below is a plot of the errors we received. k-NNs did best at $k = 1000$. Awesome!



   (c) *Latent Factor Model.* We apply the latent factor model by setting all the values in the matrix to 0 where there are NaNs. We then perform SVD yeilding

$$R = USV^T, \quad u_i = U_i, \quad v_j = (SV^T)_j^T \tag{1}$$

   Then we let $R_{ij} = \langle u_i, v_j \rangle$. Below is a plot of how well the MSE of this approximation does with $S = diag(s_1, \ldots, s_d, 0 \ldots)$ as $d \to 100$.

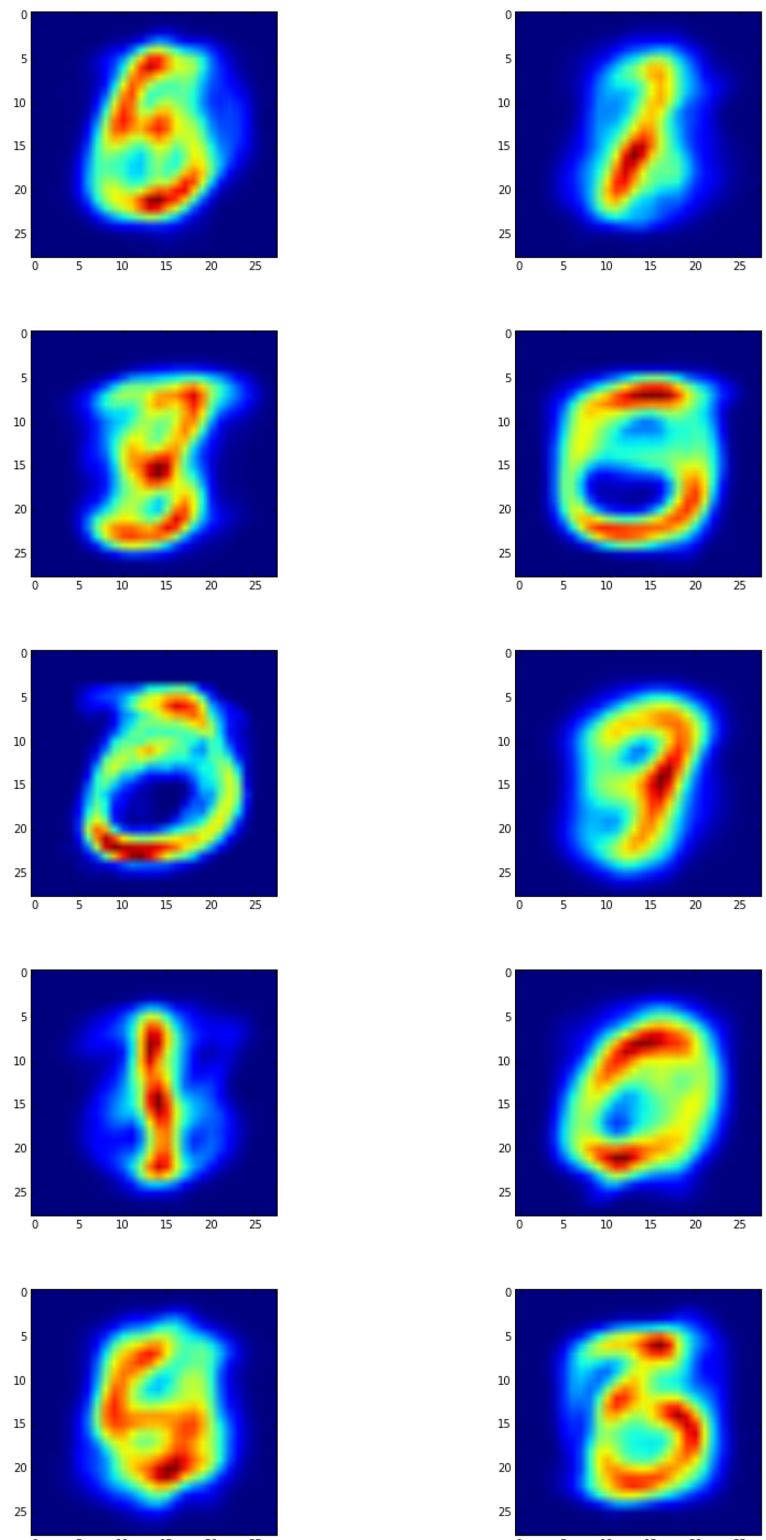
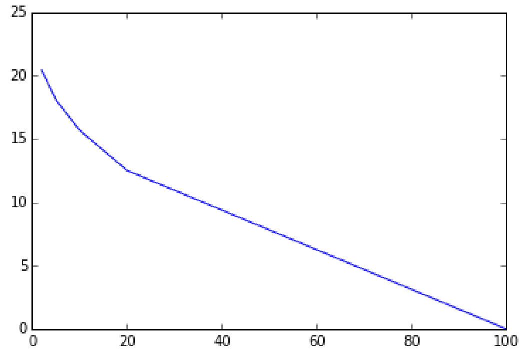
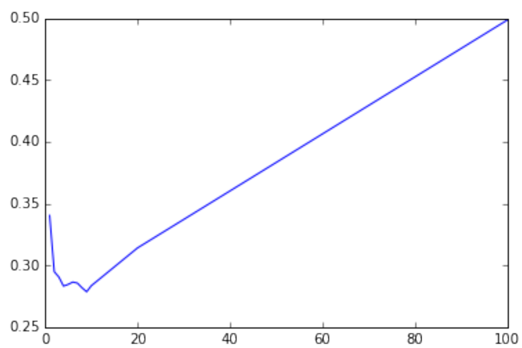


Figure 1: The mean centers of each class.



Applying these approximations to the validation set we get the following graph indicating that 10 latent dimensions optimizes the proper embedding of the data.



(d) We will now attempt to do L_1 regularized gradient descent on the Frobenius MSE. Notice that $L(u_i, v_j)$ is symmetric in its calculation of the gradient so we skip the second derivations. First we calculate the gradient with respect to a single u_i.

$$\begin{aligned}
\frac{\partial L}{\partial u_i} &= \frac{\partial}{\partial u_i} \sum_{(k,l)\in S} (v_l^T u_k - R_{kl})^2 + \frac{\partial}{\partial u_i} \lambda \sum_{k=1}^{d} u_k^T u_l \\
&= \frac{\partial}{\partial u_i} \sum_{k\in S_1} \sum_{l\in S_2} (v_l^T u_k - R_{kl})^2 + 2\lambda u_i \\
&= \frac{\partial}{\partial u_i} \sum_{l\in S_2} (v_l^T u_i - R_{il})^2 + 2\lambda u_i \\
&= 2 \sum_{l\in S_2} (v_l^T u_i - R_{il}) v_l + 2\lambda u_i
\end{aligned}$$

Using the aforementioned symmetry we get

$$\begin{aligned}
\frac{\partial L}{\partial u_i} &= \sum_{l\in S_2} (v_l^T u_i - R_{il}) v_l + \lambda u_i \\
\frac{\partial L}{\partial v_j} &= \sum_{k\in S_1} (v_j^T u_k - R_{kj}) u_k + \lambda v_j
\end{aligned} \tag{2}$$

Now we calculate the minimizers

$$\begin{aligned}
\frac{\partial L}{\partial u_i} = 0 &= \sum_{l \in S_2} (v_l^T u_i - R_{il}) v_l + \lambda u_i \\
0 &= \lambda I u_i + \sum_{l \in S_2} (v_l v_l^T u_i - \sum_{l \in S_2} v_l R_{il} \\
\sum_{l \in S_2} v_l R_{il} &= \left(\lambda I + \sum_{l \in S_2} v_l v_l \right) u_i \\
u_i &= \left(\lambda I + \sum_{l \in S_2} v_l \otimes v_l \right)^{-1} \sum_{l \in S_2} v_l R_{il}
\end{aligned}$$

This gives the following alternating minimizer method for every i, j alternate the following.

$$\begin{aligned}
u_i &= \left(\lambda I + \sum_{l \in S_2} v_l \otimes v_l \right)^{-1} \sum_{l \in S_2} v_l R_{il} \\
v_j &= \left(\lambda I + \sum_{k \in S_1} u_k \otimes u_k \right)^{-1} \sum_{l \in S_1} u_k R_{kj}
\end{aligned} \tag{3}$$