CSE 546: Homework 3

Due on November 23, 2016

Brian de Silva

0 Collaborators and Acknowledgements

1 PCA and reconstruction

1.1 Matrix Algebra Review

1. Recall that for $A \in \mathbb{R}^{n\times d}$ and $C \in \mathbb{R}^{d\times n}$, $(AC)_{ij} = \sum_{k=1}^{d} a_{ik}c_{kj}$. Also, $(B^T)_{ij} = B_{ji}$. Hence

$$(AB^T)_{ij} = \sum_{i=1}^{d}(A)_{ik}(B^T)_{jk} = \sum_{i=1}^{d} a_{ik}b_{kj}.$$

Plugging this into the definition of the trace gives

$$\begin{aligned}\mathrm{Tr}(AB^T) &= \sum_{i=1}^{n}\left(\sum_{k=1}^{d}(AB^T)_{ii}\right)\\ &= \sum_{i=1}^{n}\left(\sum_{k=1}^{d} a_{ik}b_{ik}\right).\end{aligned}$$

Similarly,

$$(B^TA)_{ij} = \sum_{k=1}^{n} b_{ki}a_{kj}$$

and, by switching the order of addition,

$$\begin{aligned}\mathrm{Tr}(B^TA) &= \sum_{i=1}^{d}\left(B^TA\right)_{ii}\\ &= \sum_{i=1}^{d}\left(\sum_{k=1}^{n} b_{ki}a_{ki}\right)\\ &= \sum_{i=1}^{n}\left(\sum_{k=1}^{d} b_{ki}a_{ki}\right)\\ &= \mathrm{Tr}(AB^T).\end{aligned}$$

2. The outer equality follows from the definition of the trace:

$$\begin{aligned}\mathrm{Tr}(\Sigma) = \mathrm{Tr}\left(\tfrac{1}{n}X^TX\right) = \frac{1}{n}\sum_{i=1}^{d}(X^TX)_{ii} &= \frac{1}{n}\sum_{i=1}^{d}\left(\sum_{k=1}^{n} x_{ik}^2\right)\\ &= \frac{1}{n}\sum_{i=1}^{d}\|X_i\|^2.\end{aligned}$$

For the other equality, we need some standard linear algebra results. Since Σ is symmetric and real, it has a real orthogonal eigendecomposition. That is to say, there exists an orthogonal matrix Q and a diagonal matrix Λ with diagonal entries $\lambda_1, \lambda_2, \ldots, \lambda_d$ such that

$$\Sigma = Q\Lambda Q^T.$$

Using our result from part 1, we have

$$\mathrm{Tr}(\Sigma) = \mathrm{Tr}\left(Q\Lambda Q^T\right) = \mathrm{Tr}\left((Q\Lambda)Q^T\right) = \mathrm{Tr}\left(Q^TQ\Lambda\right) = \mathrm{Tr}(\Lambda) = \sum_{i=1}^{d}\lambda_i.$$

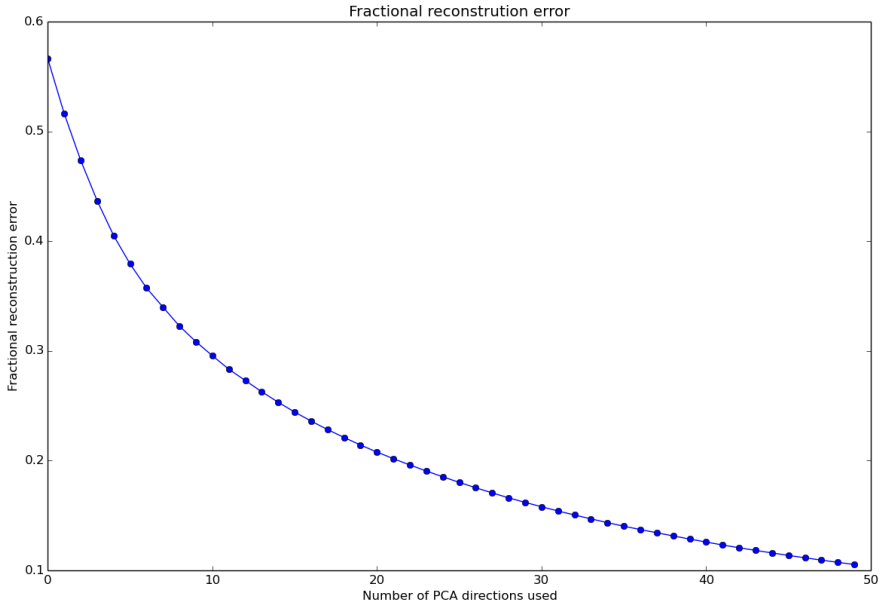

Figure 1: Fractional reconstruction error using the top 1 through 50 principal components

1.2 PCA

1. Without centering the data, the eigenvalues we obtain for Σ are

$$\begin{aligned}\lambda_1 &= 2476871.877922\\ \lambda_2 &= 285468.253236\\ \lambda_{10} &= 81022.153964\\ \lambda_{30} &= 23690.383772\\ \lambda_{50} &= 11076.827129.\end{aligned}$$

   The sum of the eigenvalues is $\sum_{i=1}^{784} \lambda_i = 5709846.669117$. Notice that the first eigenvalue accounts for almost half of the total of all the eigenvalues.

2. We plot the fractional reconstruction error retaining 1 through 50 of the top principal directions in Figure 1.

3. The first eigenvector captures the average image of the dataset. The first eigenvalue gives some indication of how close the images are to this average one. All of the images have a large amount of blank space in common, and so they are, in a sense, well-approximated by the average image. This explains why the first eigenvalue is so large compared to the rest.

4. We show the fractional reconstruction error for dimensions 2 through 50 in Figure 2.

1.3 Visualization of the Eigen-Directions

1. Visualizations of the top 10 eigenvectors are shown in Figure 3.

2. As I mentioned in a previous problem, the first eigenvector captures the average of all the images. The other eigenvalues appear to contain information about patterns encountered in many of the digits. The

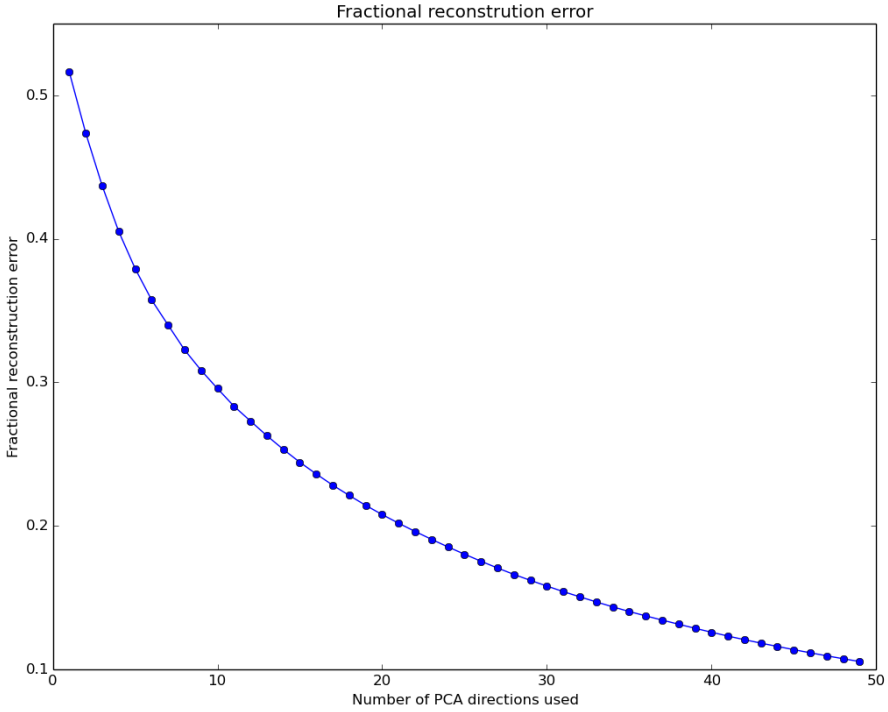

Figure 2: Fractional reconstruction error using the top 2 through 50 principal components

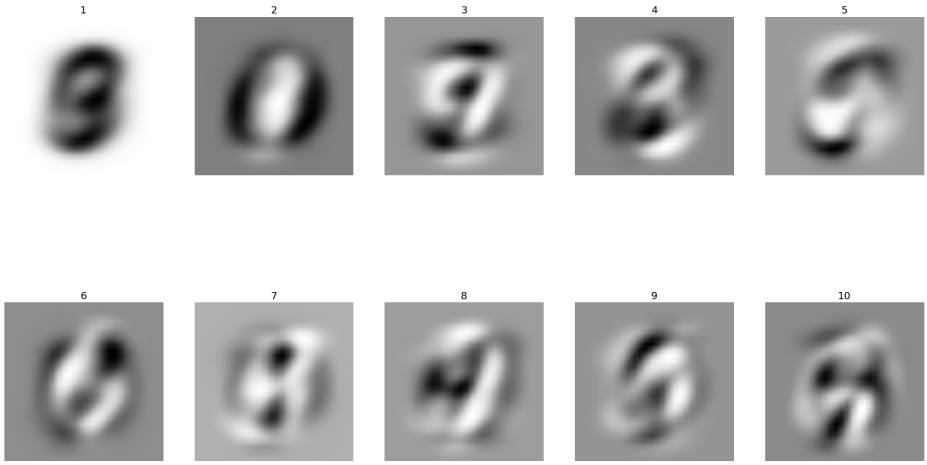

Figure 3: The eigenvectors corresponding to the 10 largest eigenvalues of Σ. Grey corresponds to values of 0, black to positive values, and white to negative values.

Figure 4: Five different images from the MNIST dataset

second eigenvalue, for example, appears to encapsulate the roundness that many digits exhibit (i.e. 0,3,6,8, and 9).

1.4 Visualization and Reconstruction

1. Figure 4 shows five images from the MNIST dataset (the first five, in fact).
2. Figure 5 shows the reconstructions produced using two and five eigenvectors, respectively. Figure 6 visualizes the reconstructions utilizing 10, 20, and 50 eigenvectors, respectively.
3. The two-eigenvector reconstruction is not very good. Under this projection some of the digits look almost identical. Using just five eigenvectors, one can already make out most of the digits. With 10 eigenvectors these digits become even clearer, though the 4 appears to be difficult to represent. The 20-eigenvalue approximation is crisper than the previous ones, but the increase in quality it starting to slow. Adding another 30 eigenvalues makes it very easy to discern the digits. Overall, adding more eigenvalues improves the projection quality, but the returns are diminishing.

2 Let's get to state of the art on MNIST!

2.1 Least Squares

The code for this problem is contained in `hw3-2.py`.

1. To obtain the results presented in this section I used the following parameters: a mini-batch size of 10, no regularization, stochastic gradient descent with a learning rate of $10^{-5}/(2\sqrt{t+1})$, where t is the iteration, and initial weights all 0. I used the Fourier features with a kernel bandwidth that is $1/2$ the approximate mean distance between points in the dataset. In order to approximate this distance I randomly sampled 100 pairs of points from the training data and took the mean of their Euclidean distances from one another. To obtain the plots below I ran SGD for 10 epochs.

(a) Reconstruction with two eigenvectors

(b) Reconstruction with five eigenvectors

Figure 5: Reconstruction of MNIST images using two and five eigenvectors

(a) Reconstruction with 10 eigenvectors

(b) Reconstruction with 20 eigenvectors

(c) Reconstruction with 50 eigenvectors

Figure 6: Reconstructions of MNIST images using different numbers of eigenvectors

3 SVMs: Hinge loss and mistake bounds

1. To show that $\ell((x,y),w) = \max\{0, 1 - w\cdot x\}$ is convex with respect to w, we will need two observations. First, for any $a, k \in \mathbb{R}$ with $k \geq 0$, we have that

$$\max\{0, ka\} = k \max\{0, a\}.$$

In the case $ka < 0$, both sides of the equality are 0. If $ka \geq 0$, then $a \geq 0$ and the equality still holds. Next, for any $a, b \in \mathbb{R}$, we have

$$\max\{0, a+b\} \leq \max\{0, a\} + \max\{0, b\}.$$

If both a and b are negative, then the above is an equality. If exactly one of a and b is negative and $a + b \leq 0$ then the inequality clearly holds. If both are nonnegative then it is also an equality.

Recall that a function f is convex if for any $t \in [0,1]$ and for any x, y in its domain, $f(tx + (1-t)y) \leq tf(x) + (1-t)f(y)$. We will show that $\ell((x,y),w)$ has this property with respect to w. Let $w_1, w_2 \in \mathbb{R}^d$ be arbitrary and let $t \in [0,1]$. Then $0 \leq (1-t) \leq 1$, and so

$$\begin{aligned}
\ell\left((x,y), tw_1 + (1-t)w_2\right) &= \max\{0, 1 - y(tw_1 + (1-t)w_2)\cdot x\} \\
&= \max\{0, 1 - tyw_1 \cdot x - (1-t)yw_2 \cdot x\} \\
&= \max\{0, 1 + (t-t) - tyw_1 \cdot x - (1-t)yw_2 \cdot x\} \\
&= \max\{0, [t - tyw_1 \cdot x] + [(1-t) - (1-t)yw_2 \cdot x]\} \\
&\leq \max\{0, t - tyw_1 \cdot x\} + \max\{(1-t) - (1-t)\text{`}yw_2 \cdot x\} \\
&= t\max\{0, 1 - yw_1 \cdot x\} + (1-t)\max\{1 - yw_2 \cdot x\} \\
&= t\ell((x,y), w_1) + (1-t)\ell((x,y), w_2).
\end{aligned}$$

Therefore $\ell((x,y),w)$ is convex with respect to w.

2. By its definition it is clear that $0 \leq \ell((x,y),w)$. If $y_i = \text{sgn}(w \cdot x_i)$ then $y_i w \cdot x_i \geq 0$, implying that $1 - y_i w \cdot x_i \leq 1$. Hence $\ell((x_i, y_i), w) = \max\{0, 1 - y_i w \cdot x_i\} \leq 1$. Combining these bounds we get

$$0 \leq \ell((x_i, y_i), w) \leq 1$$

for correctly classified points.

3. Observe that if we misclassify a point (so that $y_i = -\text{sgn}(w \cdot x_i)$) then $y_i w \cdot x_i \geq 0$. Hence $\ell((x_i, y_i), w) \geq 1$ for misclassified points. Let $I \subset \{1, 2, \ldots, n\}$ be the indices corresponding to the data points which w misclassifies. It follows that $|I| = M(w)$. By the previous part we know that for correct classifications the hinge loss is bounded between 0 and 1. Putting this all together we obtain

$$M(w) = \sum_{i=1}^{M(w)} 1 = \sum_{i \in I} 1 \leq \sum_{i \in I} \ell((x_i, y_i), w) \leq \sum_{i=1}^{n} \ell((x_i, y_i), w) = \sum_{i=1}^{n} \max\{0, 1 - y_i w \cdot x_i\}.$$

Dividing both sides by n gives the desired result.

4 Fitting an SVM classifier by hand

We will use the equations

$$\hat{\boldsymbol{w}}, \hat{w}_0 = \text{argmin}\|w\|^2 \quad s.t. \tag{1}$$

$$y_1(w^T\phi(x_1) + w_0) \geq 1 \tag{2}$$

$$y_2(w^T\phi(x_2) + w_0) \geq 1. \tag{3}$$

1. The optimal vector $\hat{\boldsymbol{w}}$ is orthogonal to the decision boundary, which, in this case, should be the plane lying directly in the middle of $\phi(x_1) = (1,0,0)^T$ and $\phi(x_2) = (1,2,2)^T$. Hence $\hat{\boldsymbol{w}}$ is a normal vector for this plane and all normal vectors for this plane are proportional to $\phi(x_2) - \phi(x_1) = (0,2,2)^T$. Hence $\hat{\boldsymbol{w}} = \alpha(0,1,1)^T$ for some $\alpha \in \mathbb{R}$. In particular $(0,1,1)^T$ is parallel to $\hat{\boldsymbol{w}}$.

2. Since the decision boundary is the plane in the middle of the two new features, the margin is simply half the distance between the two points $\phi(x_1)$ and $\phi(x_2)$. I.e. the margin is

$$\frac{1}{2}\|\phi(x_2) - \phi(x_1)\| = \frac{1}{2}\|(0,2,2)^T\| = \frac{1}{2}(2\sqrt{2}) = \sqrt{2}.$$

3. Since the margin is equal to $1/\|\hat{\boldsymbol{w}}\|$, we know that $\|\hat{\boldsymbol{w}}\| = 1/\sqrt{2}$. From before, $\|\hat{\boldsymbol{w}}\| = \|\alpha(0,1,1)\| = \alpha\sqrt{2}$. Equating these two expressions, we see that $\alpha = \frac{1}{2}$, so

$$\hat{\boldsymbol{w}} = \frac{1}{2}\begin{bmatrix}0\\1\\1\end{bmatrix} = \begin{bmatrix}0\\1/2\\1/2\end{bmatrix}.$$

4. Having determined $\hat{\boldsymbol{w}}$ we are now in a position to solve for w_0. Equation (1) now simply states that we should choose w_0 to have minimal magnitude. Plugging into constraints (2) and (3), we get

$$\begin{aligned} y_1(w^T\phi(x_1) + w_0) = -1[(0,1/2,1/2)\cdot(1,0,0) + w_0] &= -w_0 \geq 1 &&\implies w_0 \leq -1 \\ y_2(w^T\phi(x_2) + w_0) = 1[(0,1/2,1/2)\cdot(1,2,2) + w_0] &= 2 + w_0 \geq 1 &&\implies w_0 \geq -1. \end{aligned}$$

   The only value for w_0 which satisfies both of these constraints is $w_0 = -1$.

5. Substituting in the values we found, we obtain

$$f(x) = \hat{w}_0 + \hat{\boldsymbol{w}}^{\boldsymbol{T}}\phi(x) = -1 + (0,1/2,1/2)\cdot(1,\sqrt{2}x,x^2) = -1 + \frac{\sqrt{2}}{2}x + \frac{1}{2}x^2.$$

   Figure 7 shows a 2d plot of this discriminant function (generated in Mathematica). The blue curve is the graph of the function and the red points are the values of f at the two data points.

5 K-Means

For this problem I chose to cluster on the 50-dimensional dataset. For the visualizations of the means I projected back up to the 784-dimensional space. It should also be noted that in order to improve numerical stability in the computation of the square reconstruction error, I first divided all the points in the dataset by the 2-norm of the largest magnitude entry. This preserves the geometry of the points, but alters the numerical value of the squared reconstruction error (but not its qualitative behavior). I made sure to rescale the means appropriately before projecting them back up to 784 dimensions.

5.1 Run the algorithm

1. I initialized K-means in a naive way by simply selecting 16 random points from the data set to be the initial means. The code for this problem is contained in `hw3-5.py`.

   (a) Figure 8 shows the squared reconstruction error as a function of iteration number of the K-means algorithm with 16 means. As I mentioned above, this is the squared reconstruction error for the rescaled projections of the data points onto the first 50 singular vectors.

   (b) Figure 9 gives the numbers of data points assigned to each of the means in descending order. Along the x-axis are the labels associated with each mean.

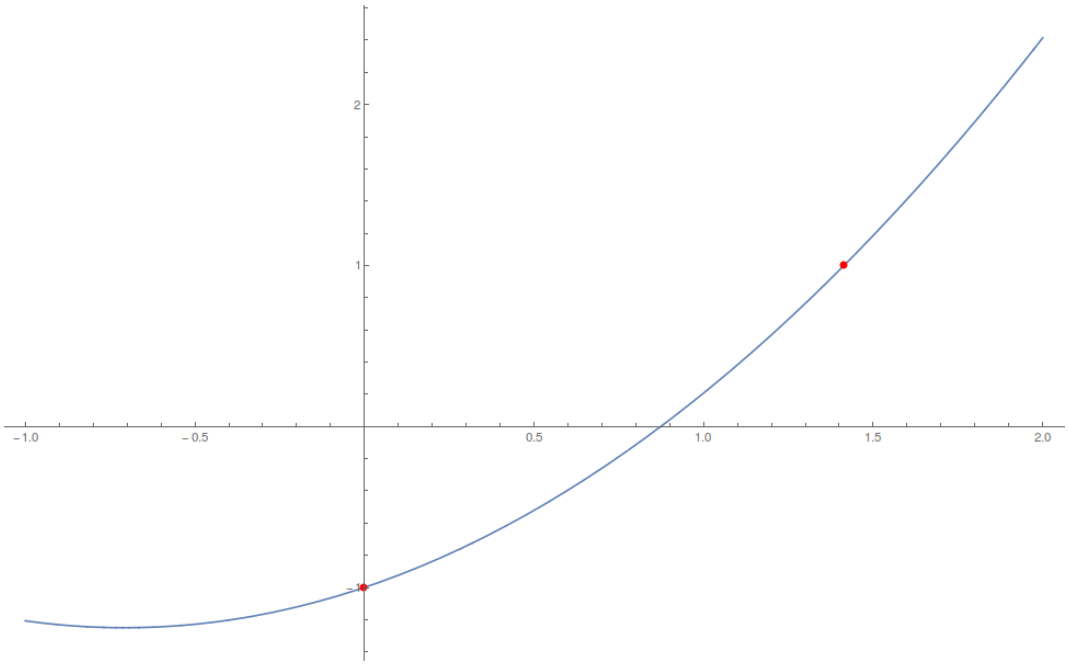

Figure 7: The discriminant function $f(x)$ (blue) and the two points in the dataset (red)

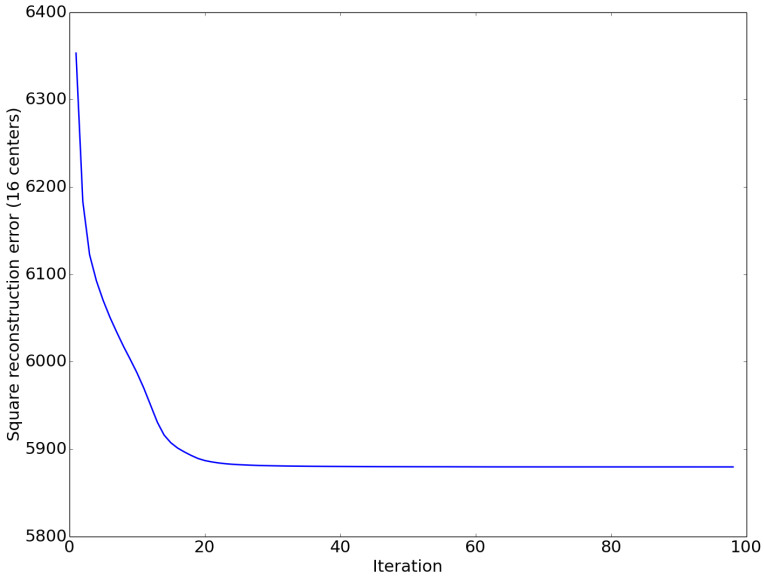

Figure 8: Squared reconstruction error as a function of iteration number for 16 means

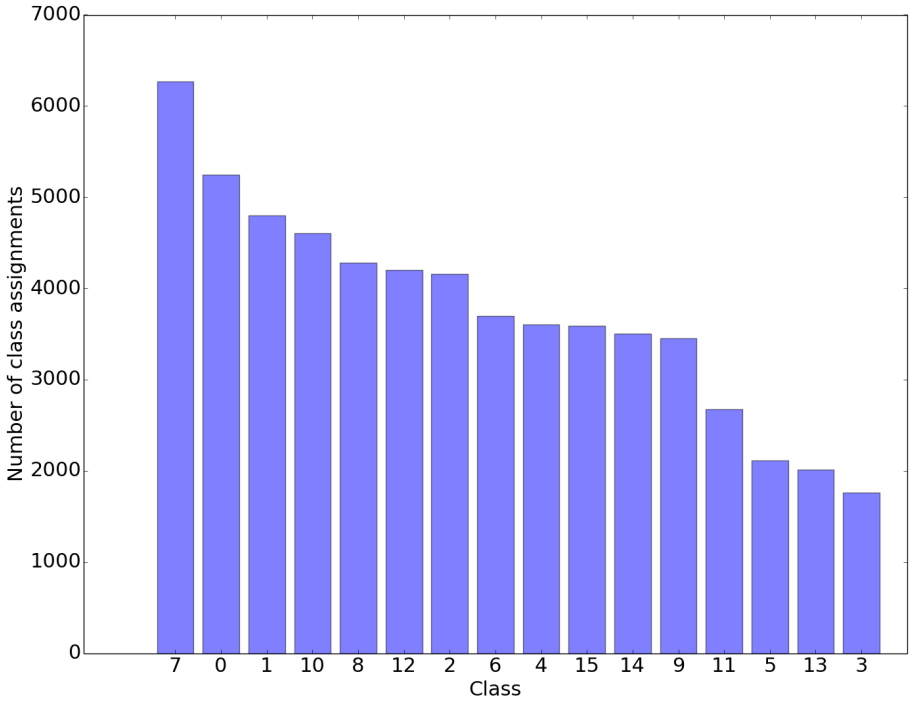

Figure 9: The number of data points assigned to each class using 16 means

Figure 10: Visualizations of the 16 means, ordered (descending) from top-left to bottom-right by the number of data points assigned to them

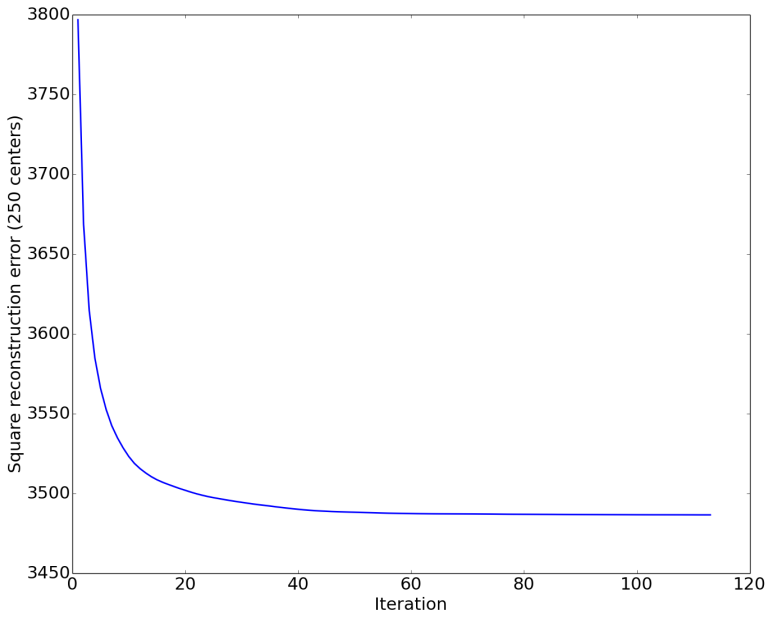

Figure 11: Squared reconstruction error as a function of iteration number for 250 means

(c) Figure 10 provides visualizations of the 16 means. The top left image corresponds to the mean with the most points assigned to it and the one in the bottom left to the mean with the least. The images are sorted in descending order in the same order as one would read text (from top to bottom, left to right).

2. The means that output by the method do in fact look as if they each correspond to specific digits or averages of two similar looking digits. Since there are more than 10 possible classes, the means correspond to specific digits and digits that look similar to one another. Some digits can be drawn in multiple ways and have multiple means corresponding to each style. The mean with the most data points assigned to it looks to be somewhere between a 9 and a 7, so it is likely that some 9's and some 7's are placed in its cluster. There are two variations of 1 present, a vertical one and a somewhat tilted one. There are also means that look to be close to both 8's and 3's, 9's and 4's and 5's and 8's. This output is a good sanity check that K-means is doing what we think it should do for this dataset.

3. I used the same initialization and rescaling procedures for 250 means as in the previous part of the problem.

   (a) Figure 11 shows the squared reconstruction error as a function of iteration number of the K-means algorithm with 250 means. As I mentioned above, this is the squared reconstruction error for the rescaled projections of the data points onto the first 50 singular vectors.

   (b) Figure 12 gives the numbers of data points assigned to each of the means in descending order. Along the x-axis are the labels associated with each mean.

   (c) Figure 13 provides visualizations of a random selection of 16 of the 250 means. The top left image corresponds to the mean with the most points assigned to it and the one in the bottom left to the mean with the least. The images are sorted in descending order in the same order as one would read text (from top to bottom, left to right).

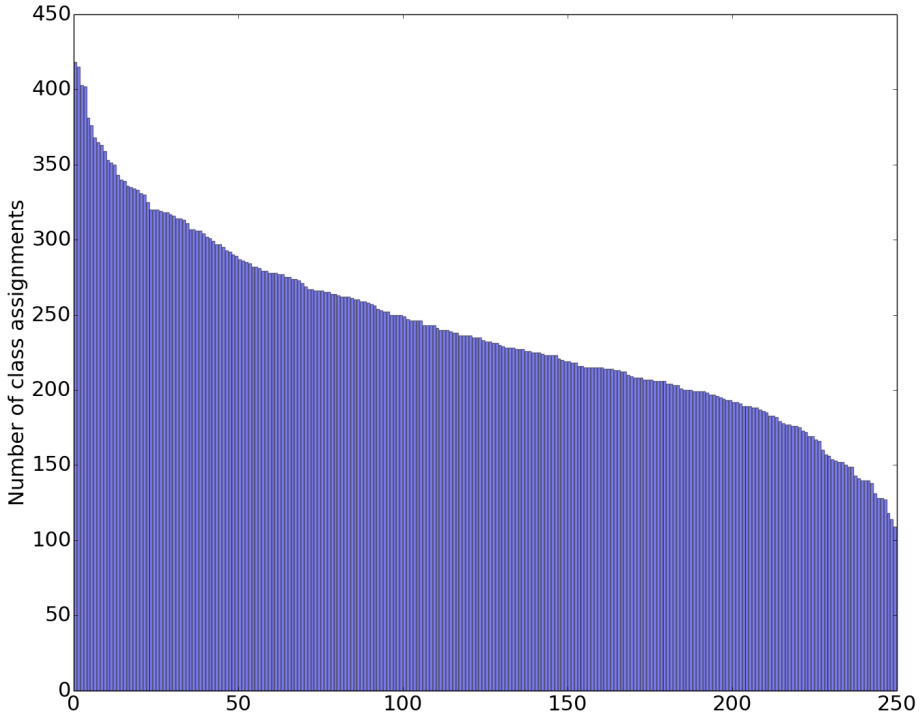

Figure 12: The number of data points assigned to each class using 250 means

Figure 13: Visualizations of the 16 randomly selected means, ordered (descending) from top-left to bottom-right by the number of data points assigned to them

5.2 Classification with K-means

1. For $K = 16$, my 0/1 losses for the training and test sets are 0.313900 and 0.310600, respectively.
2. For $K = 250$, my 0/1 losses for the training and test sets are