**Dimension Reduction Workshop** **CS 480**

*When you are finished, share your Python code with me. If you are using Google Colab, share your code with:* `lins.brian@gmail.com`

In this workshop, we will use principal component analysis to reduce the dimension of some images, and then use the k-nearest neighbors algorithm (with $k = 1$) to classify the images. First you will need to enter the following code (which you can also copy from today's notes on the website).

```
import tensorflow as tf
import matplotlib.pyplot as plt
import numpy as np

# Load the data
mnist = tf.keras.datasets.mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

# Find the principal components
X = np.array([image.flatten() for image in x_train])
Q = np.cov(X.T)
d, W = np.linalg.eigh(Q)
```

1. Complete the following function.

   ```
   def nearest(xs, ys, z):
       # xs is a list of numpy arrays
       # z is a numpy array with the same shape as each x in xs.
       # ys is a list of labels of the same length as xs.
       # returns the y value corresponding to the nearest x to z.
   ```

2. Compute a matrix $Z$ by flattening the images in the test data like we flattened the training data to make $X$. Then compress the data in $X$ and $Z$ by computing $X_k = XW_k$ and $Z_k = ZW_k$ where $W_k$ is the matrix containing the last $k$ columns of $W$.

3. Use the first 100 rows of $X_k$ as the `xs` in the function `nearest(xs, ys, z)` and use the first 100 entries of `y_train` for the `ys`. Apply the nearest function to every row of $Z_k$ and find the proportion of the predictions that are correct when $k = 100, 50, 25, 10, 5$. Which dimension is the most accurate?