ANALYSIS OF MARS ORBITAL IMAGES

Richard Han
gtID: 903757484
Georgia Tech
rhan70@gatech.edu

ABSTRACT

In this paper, I explore the application of a manifold learning technique called epsilon-ISOMAP and a dimensionality reduction technique called principal components analysis (or PCA) on a dataset consisting of orbital images of Mars in order to capture significant features of the images. I also identify some images considered outliers based on the adjacency matrix used in epsilon-ISOMAP. Finally, I apply several classification models, some linear and some non-linear, and compare their performance.

***K*eywords** Mars · ISOMAP · PCA · Clustering · Classification

1 Problem Statement

NASA's Planetary Data System Imaging Atlas allows users to search for images based on the content within the image such as craters. This leads to the machine learning problem of classifying images. In my project, I would like to explore features of the images using epsilon-ISOMAP and PCA as well as to accurately classify the images using various classification models. I am particularly interested in the HiRISE dataset of orbital images of the surface of Mars.

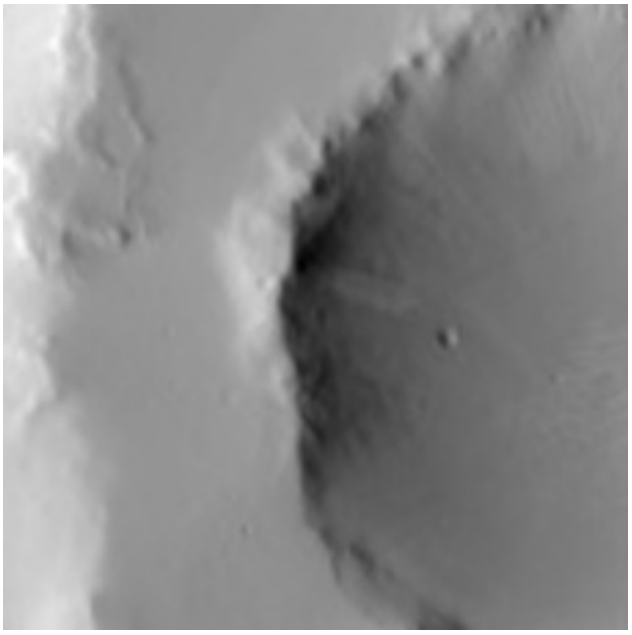
In [1], the authors trained a convolutional neural network using Earth images and transfer learning to classify the HiRISE images. They achieved very high accuracy rates. In this paper, I apply KNN, logistic regression, linear SVM, kernel SVM, and a neural network; I then compare their performance and draw some conclusions about the geometry of the data set.

2 Data Source

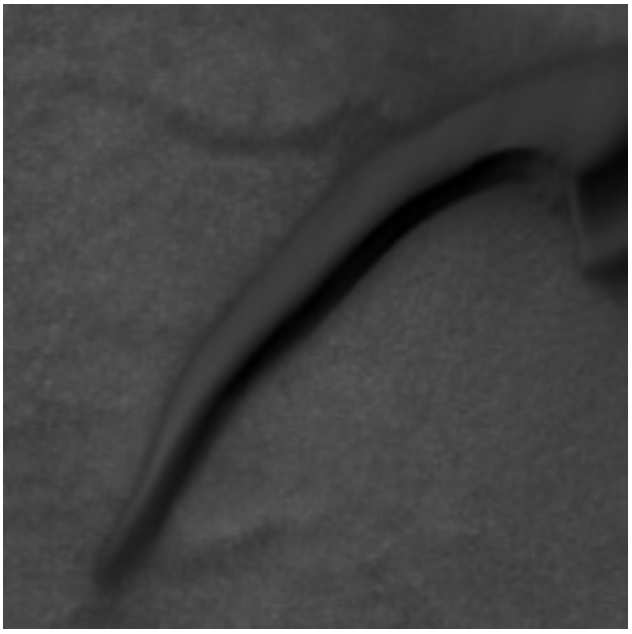
The data consists of 3820 (227 by 227 pixels) grayscale images of landmarks such as craters, dunes, and streaks. There are a total of 7 classes: 'other', 'crater', 'dark dune', 'streak', 'bright dune', 'impact', and 'edge'. There are no images that have the label 'impact', and there are only 37 images that are labeled 'streak'. In contrast, more than half the images are labeled 'other'. The images labeled 'edge' have black edges in them so that only part of the surface is showing.

The data set can be found here: dataset

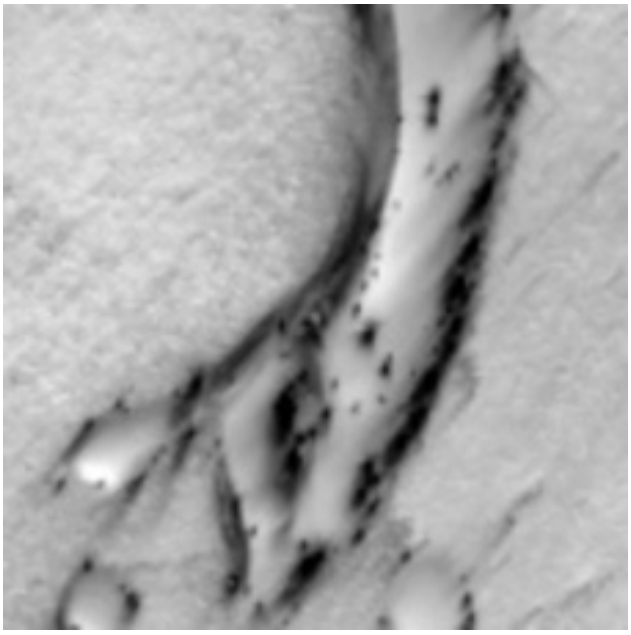
Here is an example of an image of a crater:



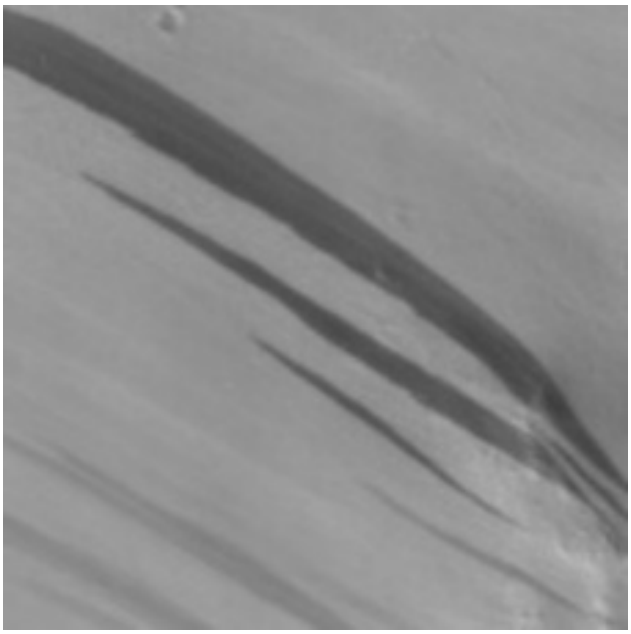
Here is an example of an image of a dark dune:



Here is an example of an image of a bright dune:



Here is an example of an image of a streak:



3 Epsilon-ISOMAP

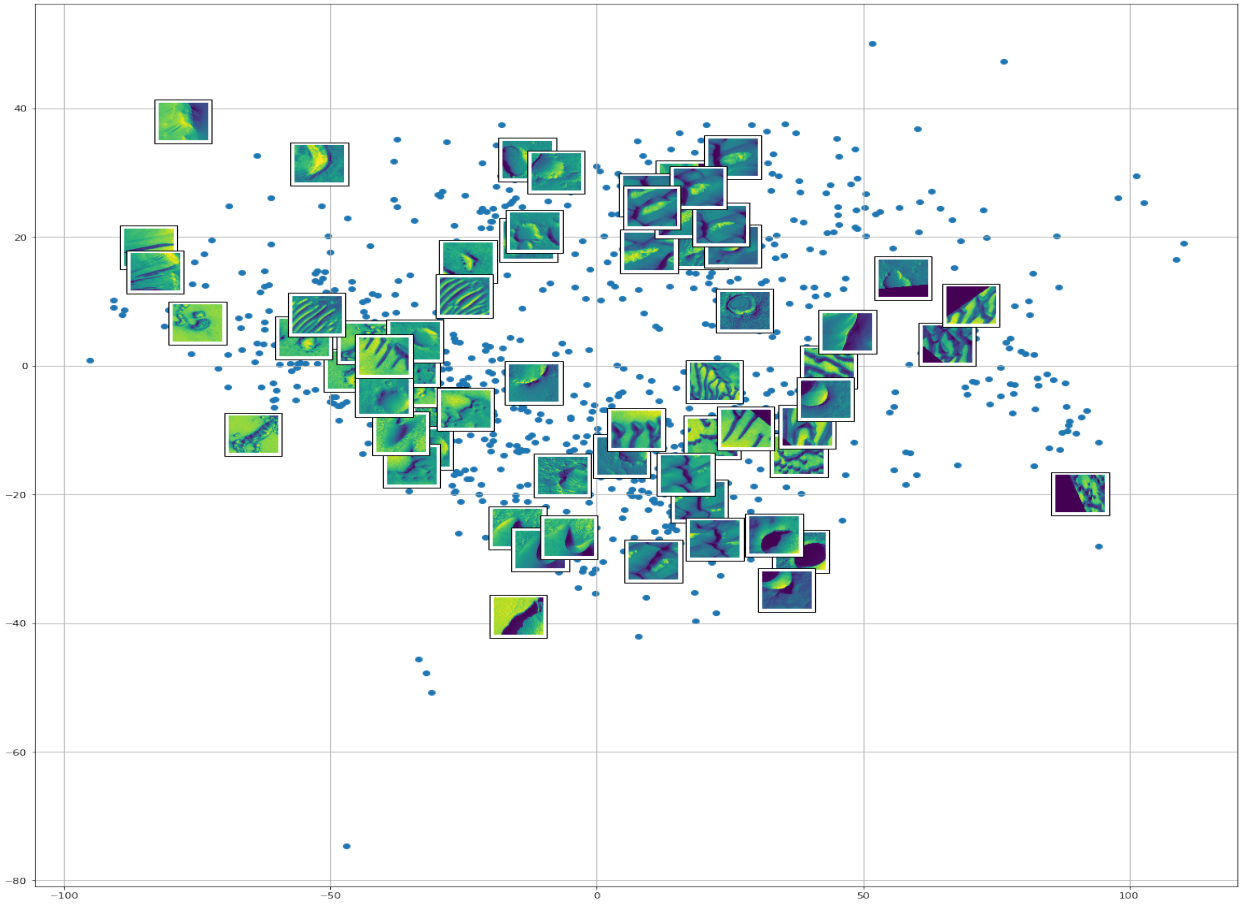
3.1 Methodology

I would like to apply epsilon-ISOMAP to the images consisting of just the four categories ‘crater’, ‘dark dune’, ‘bright dune’, and ‘streak’ since we have an idea of what the images represent. In doing so, I would like to see if epsilon-ISOMAP can cluster the images according to the four known categories. Secondly, I would like to apply epsilon-ISOMAP to a set of images classified as ‘other’; this is particularly interesting because we don’t have any labels for what’s in the images, and how epsilon-ISOMAP clusters them should reveal some underlying patterns in the images. One of the questions that will arise in this approach is what epsilon should be.

There are four main steps to epsilon-Isomap. First, we build the weighted adjacency matrix by considering if the euclidean distance between two data points is less than or equal to epsilon and setting that distance as the weight for the edge joining the two points. Otherwise, the weight is 0. Then, we use a shortest path algorithm to find the shortest distance between each pair of data points to form the shortest distance matrix D. The third step is to find the matrix C using D and find the eigendecomposition of C to get the first k largest eigenvalues and corresponding eigenvectors. We then have the scores for each principal component, allowing us to plot the data points in a lower-dimensional space. The main ideas behind ISOMAP is to preserve the manifold structure of the data in a high-dimensional space through the use of an adjacency matrix and the idea of a geodesic or 'walking distance' between points while reducing the number of dimensions to represent our data; this reduction of dimension is done through the eigendecomposition of the matrix C.

3.2 Results and Evaluation

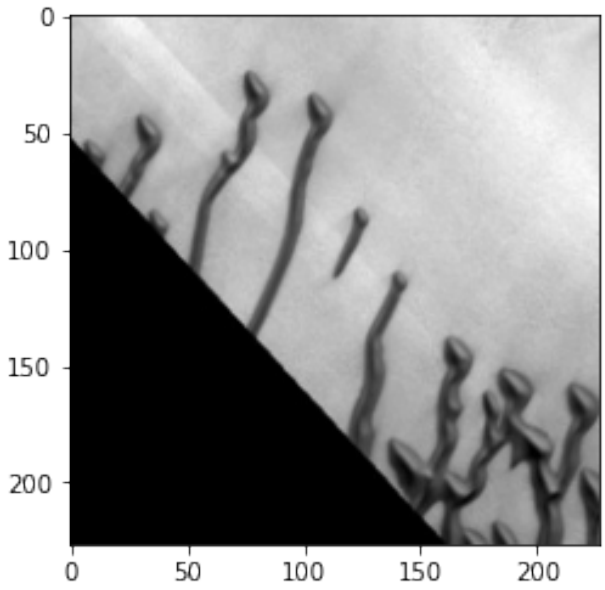
There were 970 images in the group consisting of images labeled 'crater', 'dark dune', 'bright dune', and 'streak'. I had to play around with the epsilon value to get a good clustering, and I settled on a value of $\epsilon = 50$. Here is a scatter plot of the images in the low-dimensional space:



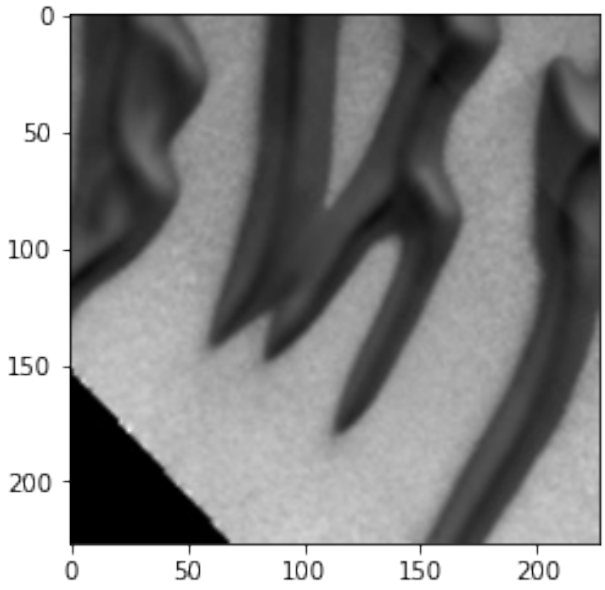
Some of the images are displayed in the scatter plot. I do see similarities between the images that are close together, indicating that ISOMAP is doing a good job clustering the different categories.

Here's a plot of the adjacency matrix A:

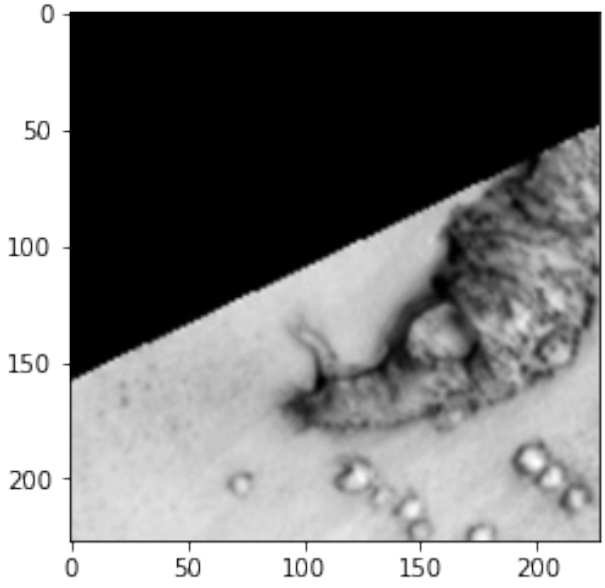
Note that there are some isolated nodes. Here are images of what some of these outliers look like:

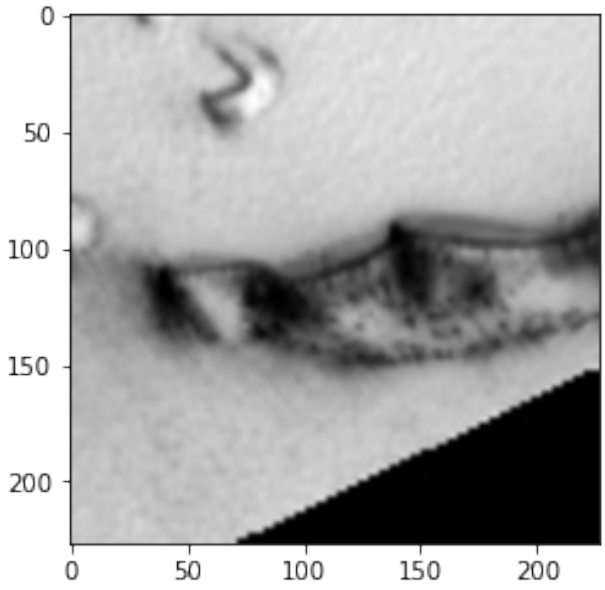


This one looks like small trails of dried up blood–very creepy. Here's one that looks like smooth pitchforks–almost artificial-looking:

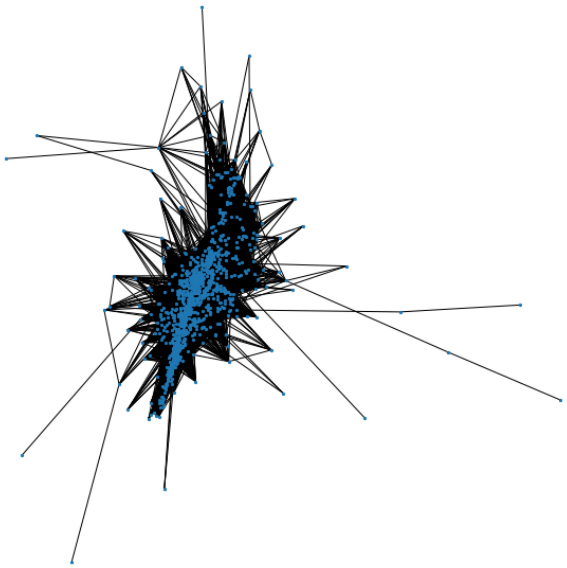


And, here are a couple more:

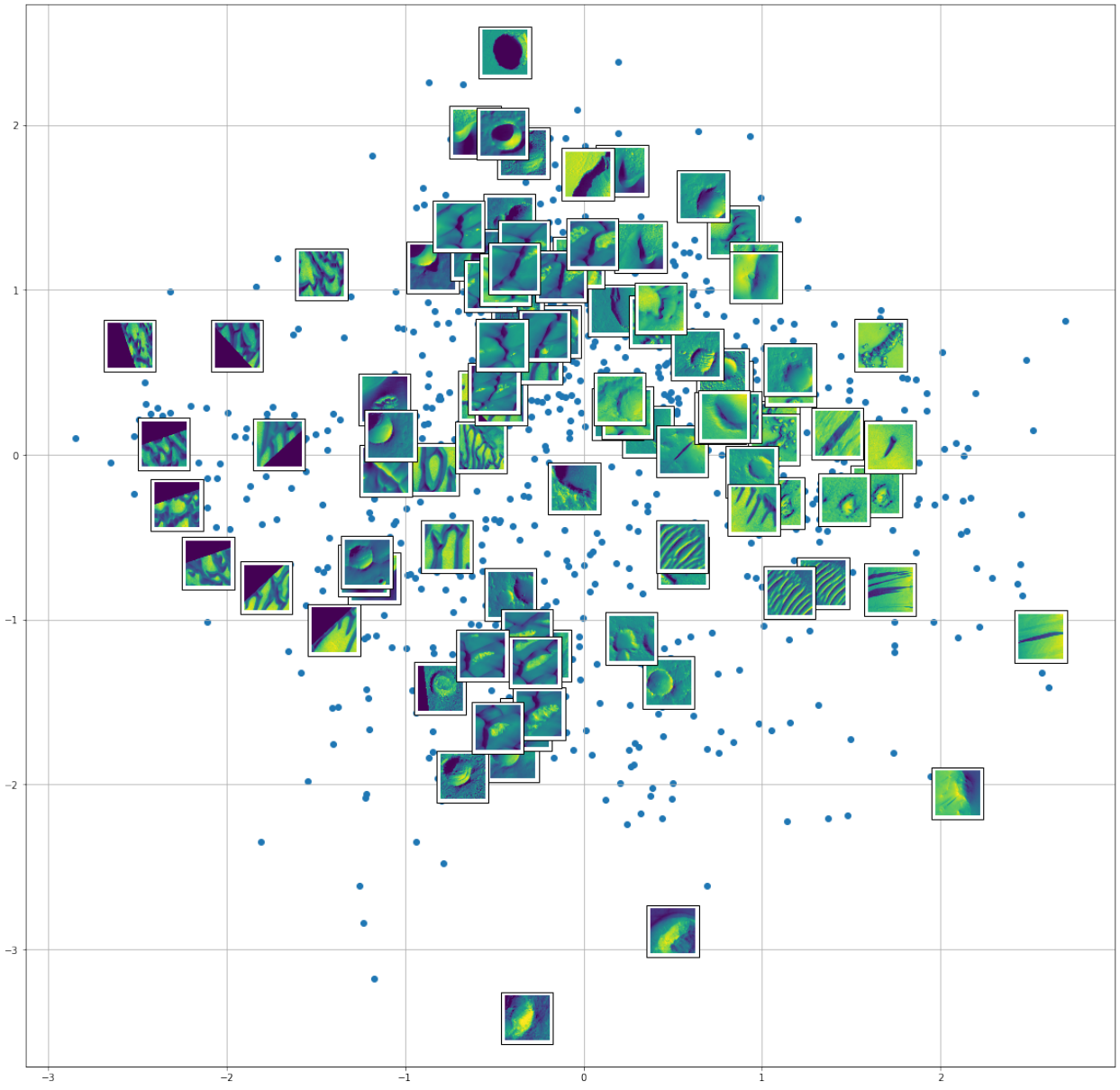




After removing the isolated nodes, here's what the adjacency matrix looks like:

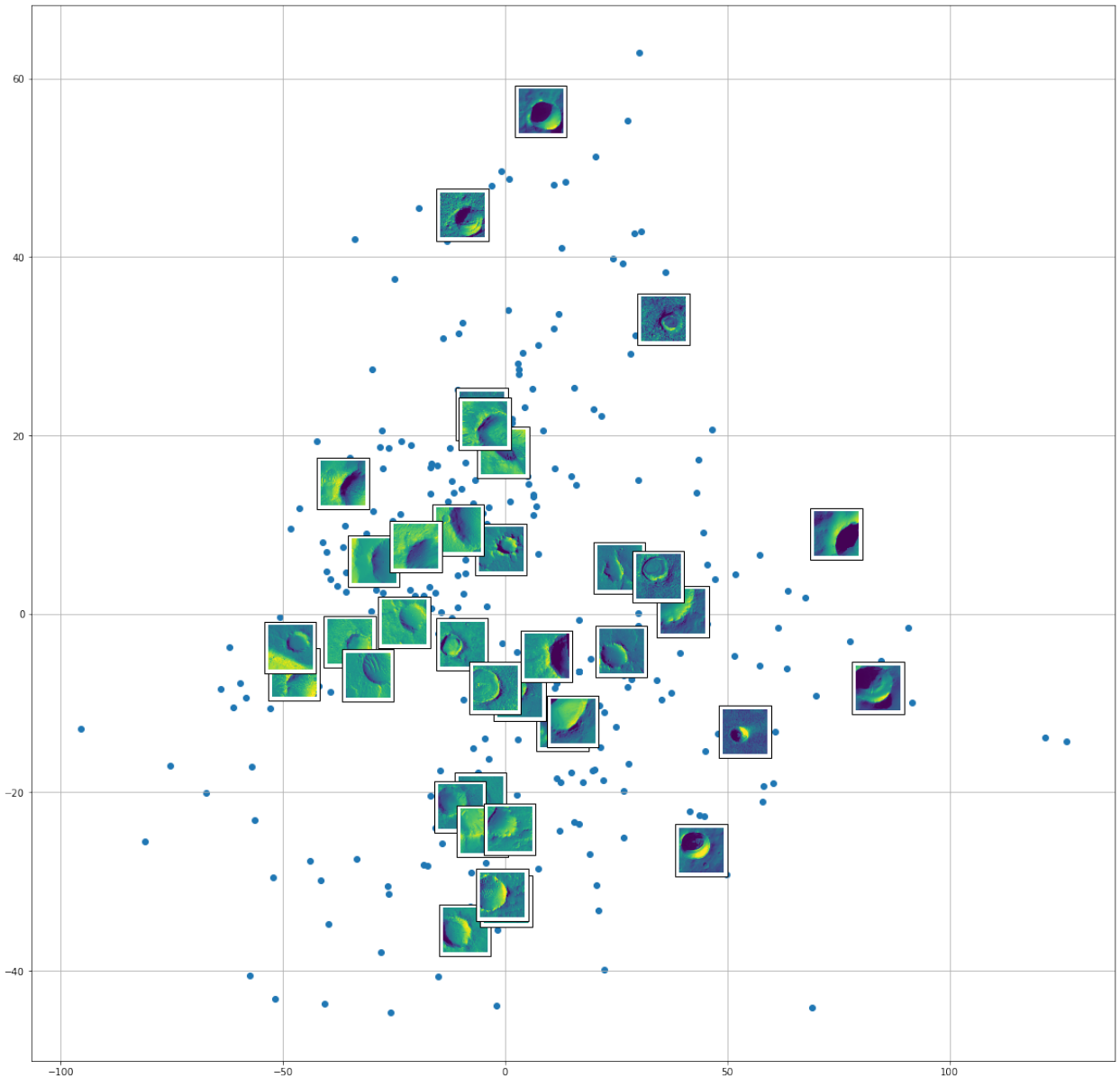


I also performed PCA on the same dataset, and here is the PCA clustering:

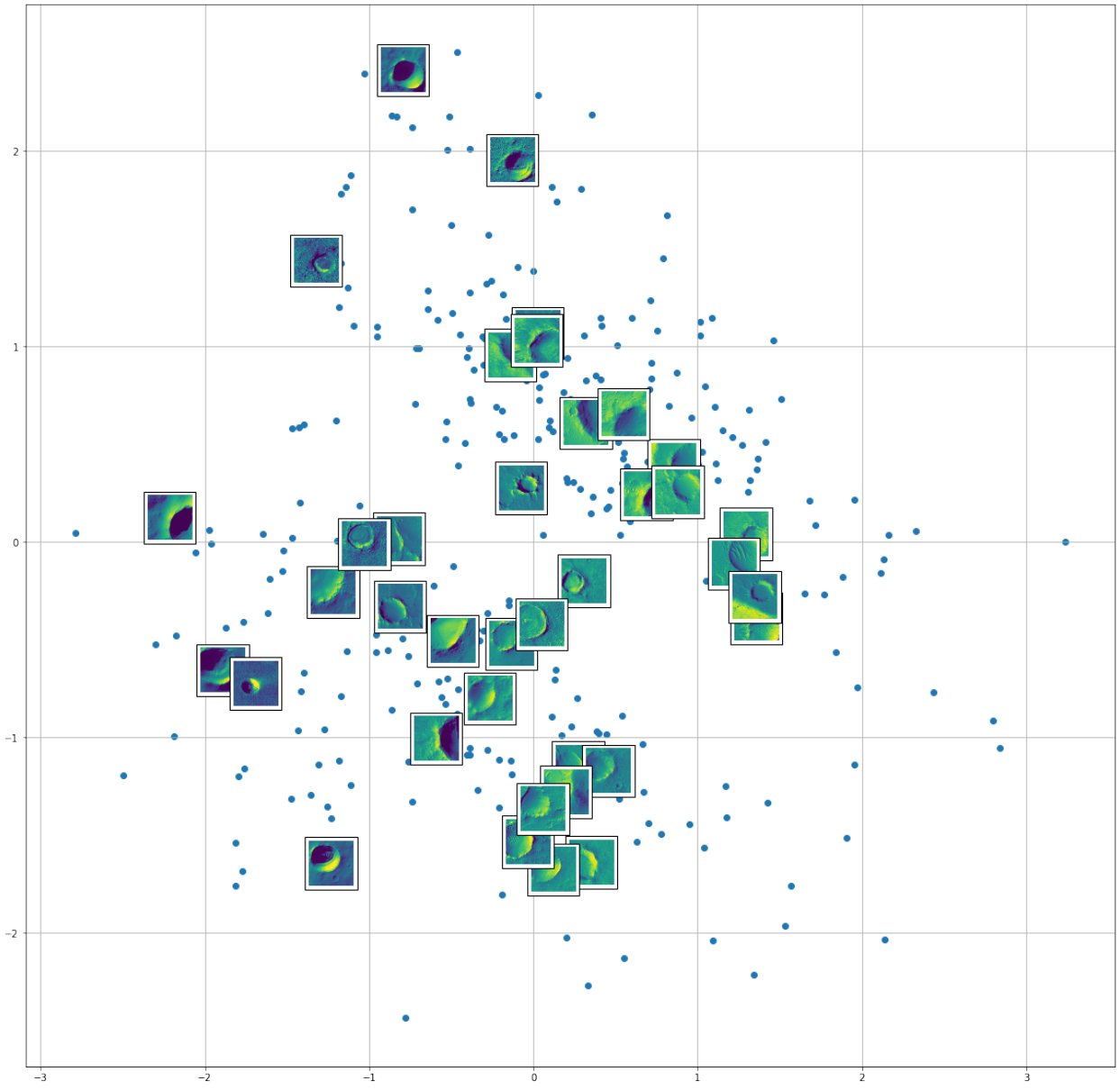


It appears to do an equally good job of clustering the images. I will now apply epsilon-ISOMAP and PCA to groups of images that consist solely of craters, solely of dark dunes, solely of bright dunes, and solely of streaks to see if epsilon-ISOMAP outperforms PCA.

For the 369 crater images, I used an epsilon value of 50. Here is the scatter plot for epsilon-ISOMAP on the crater images:

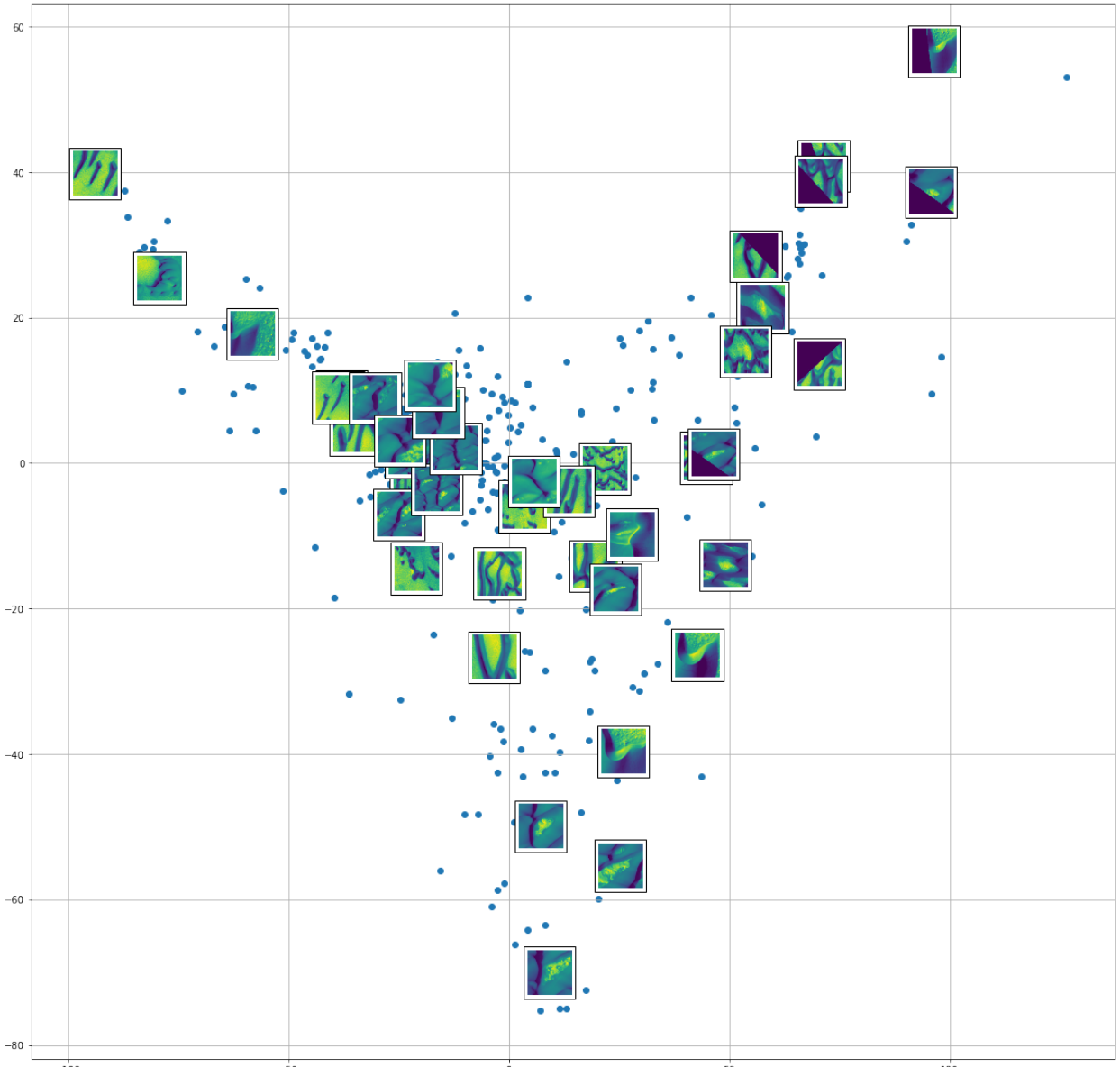


Here is the scatter plot for PCA on the crater images:

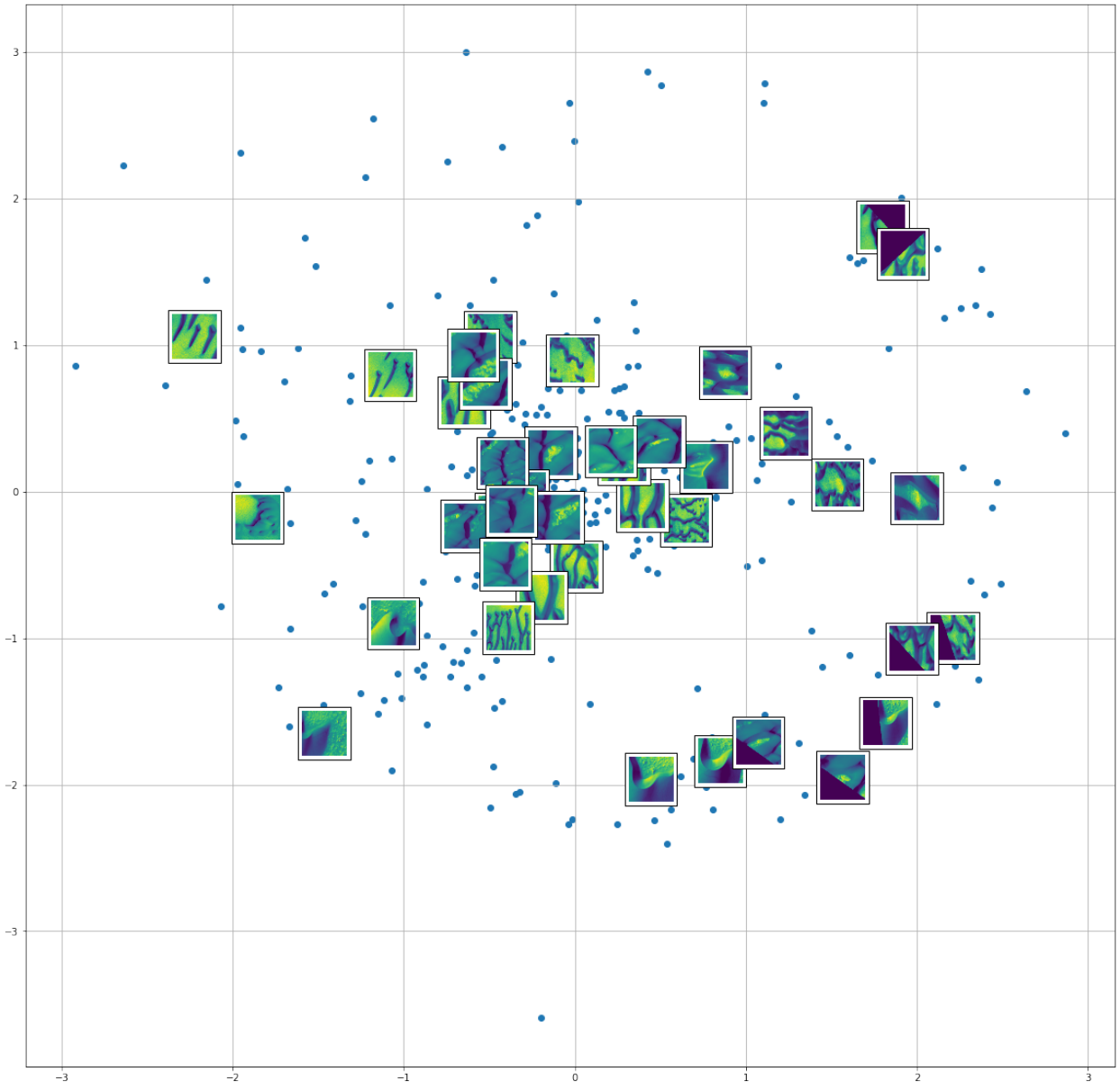


Both epsilon-ISOMAP and PCA seem to be making identical clusterings of the images.

For the 408 dark-dune images, I used an epsilon value of 45. Here is the scatter plot for epsilon-ISOMAP on the dark-dune images:

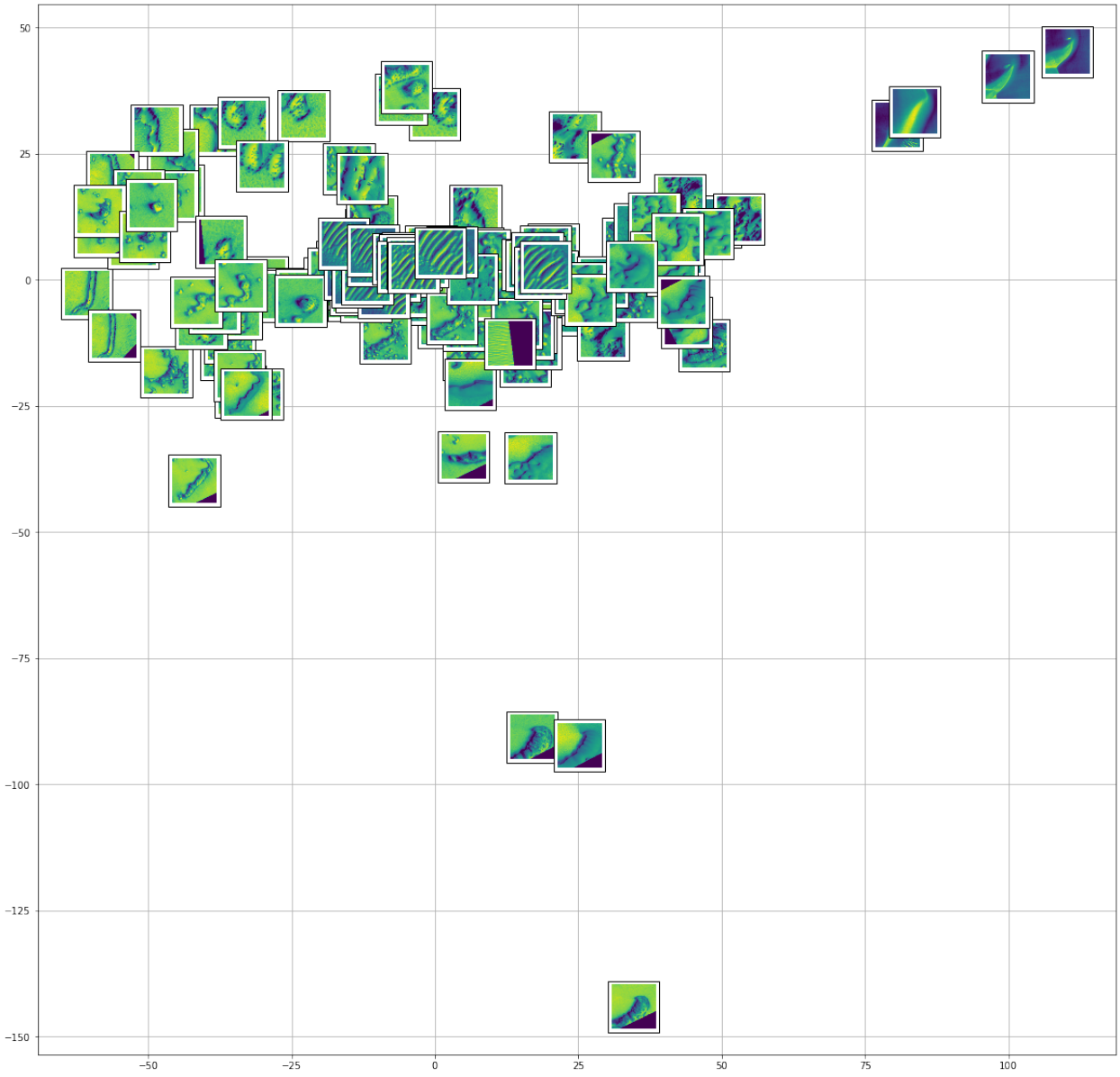


Here is the scatter plot for PCA on the dark-dune images:

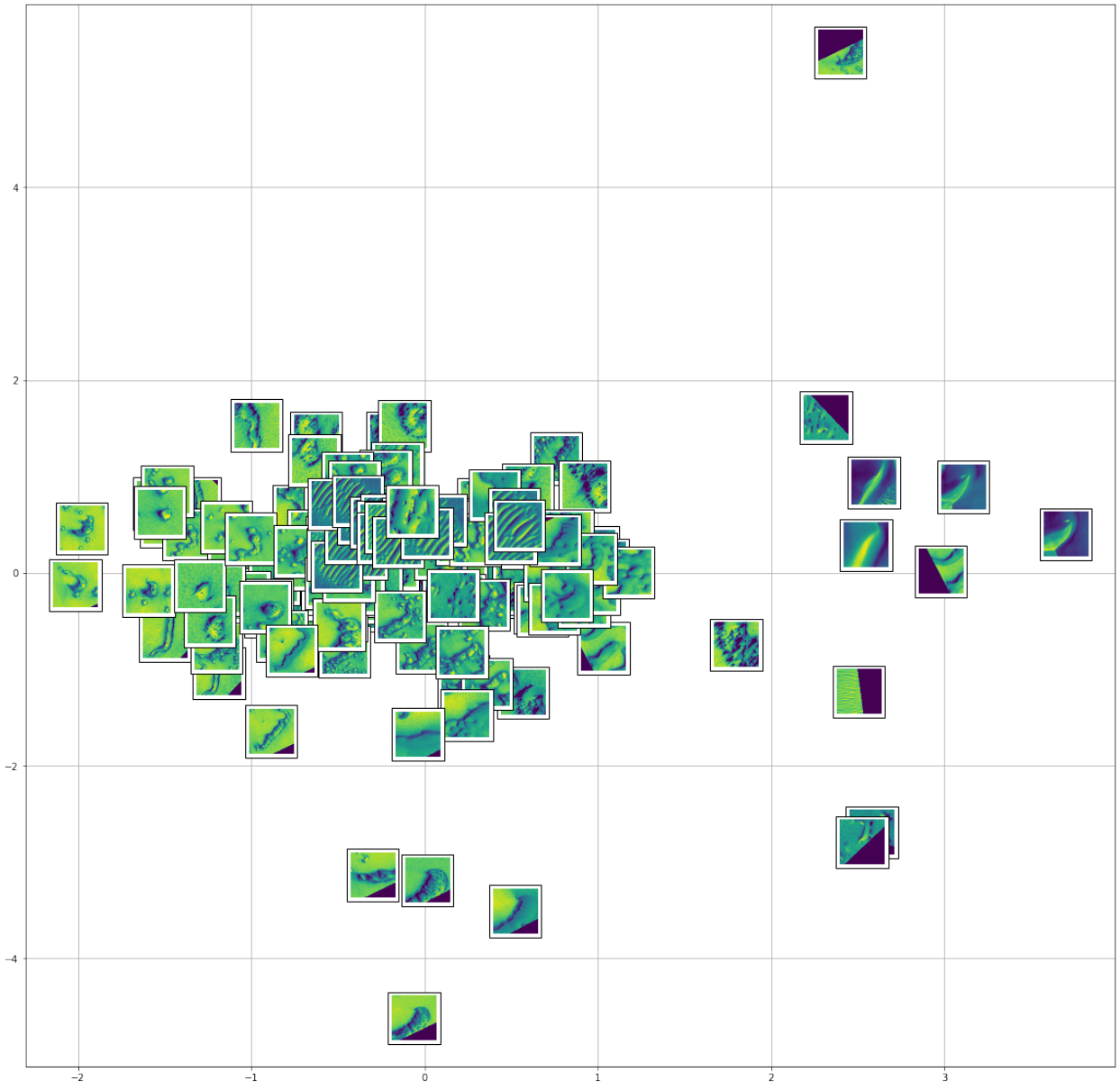


Epsilon-ISOMAP and PCA seem to be making similar clusterings of the images with epsilon-ISOMAP doing a slightly better job.

For the 156 bright-dune images, I used an epsilon value of 55. Here is the scatter plot for epsilon-ISOMAP on the bright-dune images:



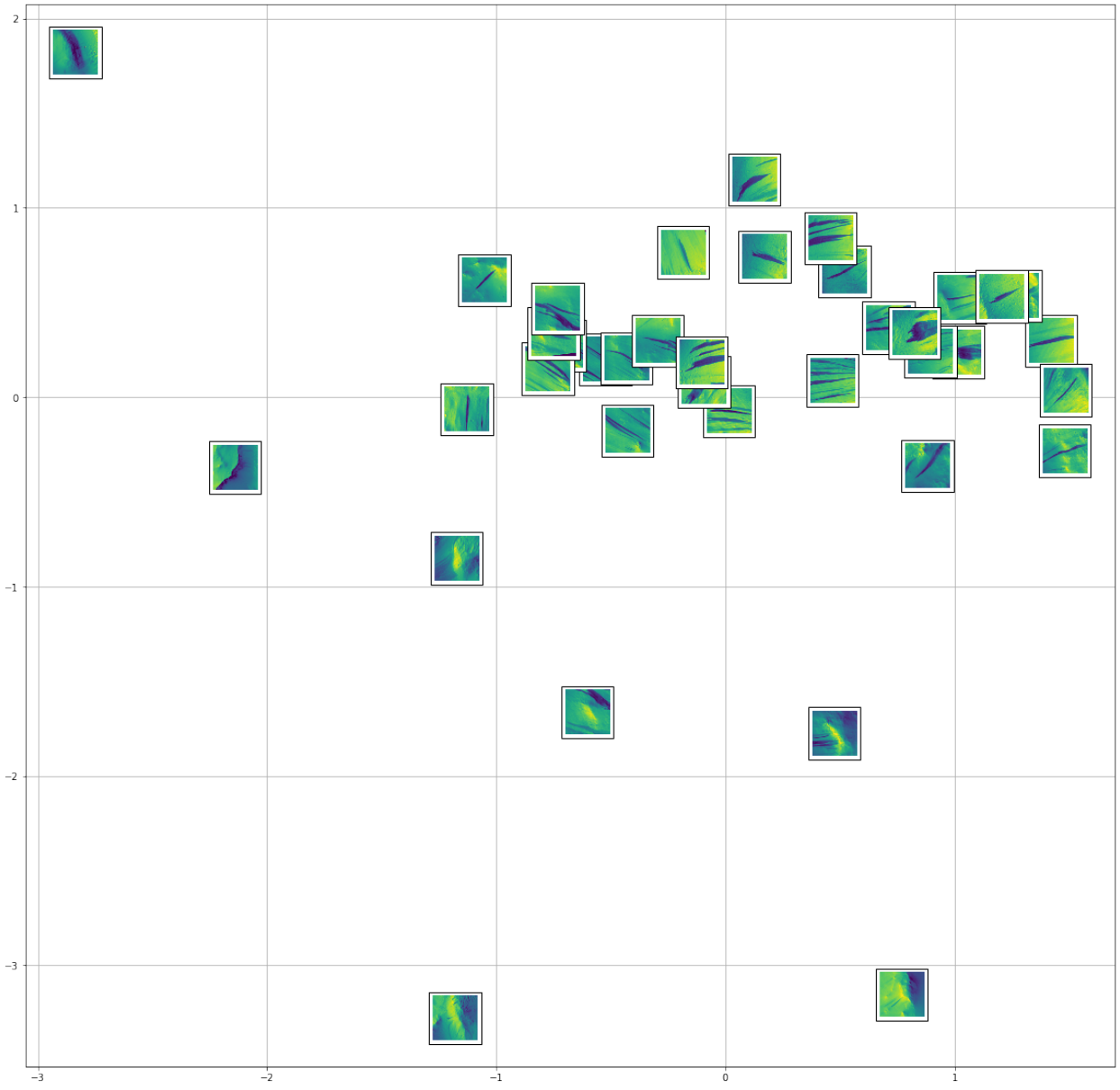
Here is the scatter plot for PCA on the bright-dune images:



For the bright-dune images, epsilon-ISOMAP does a bit better than PCA.

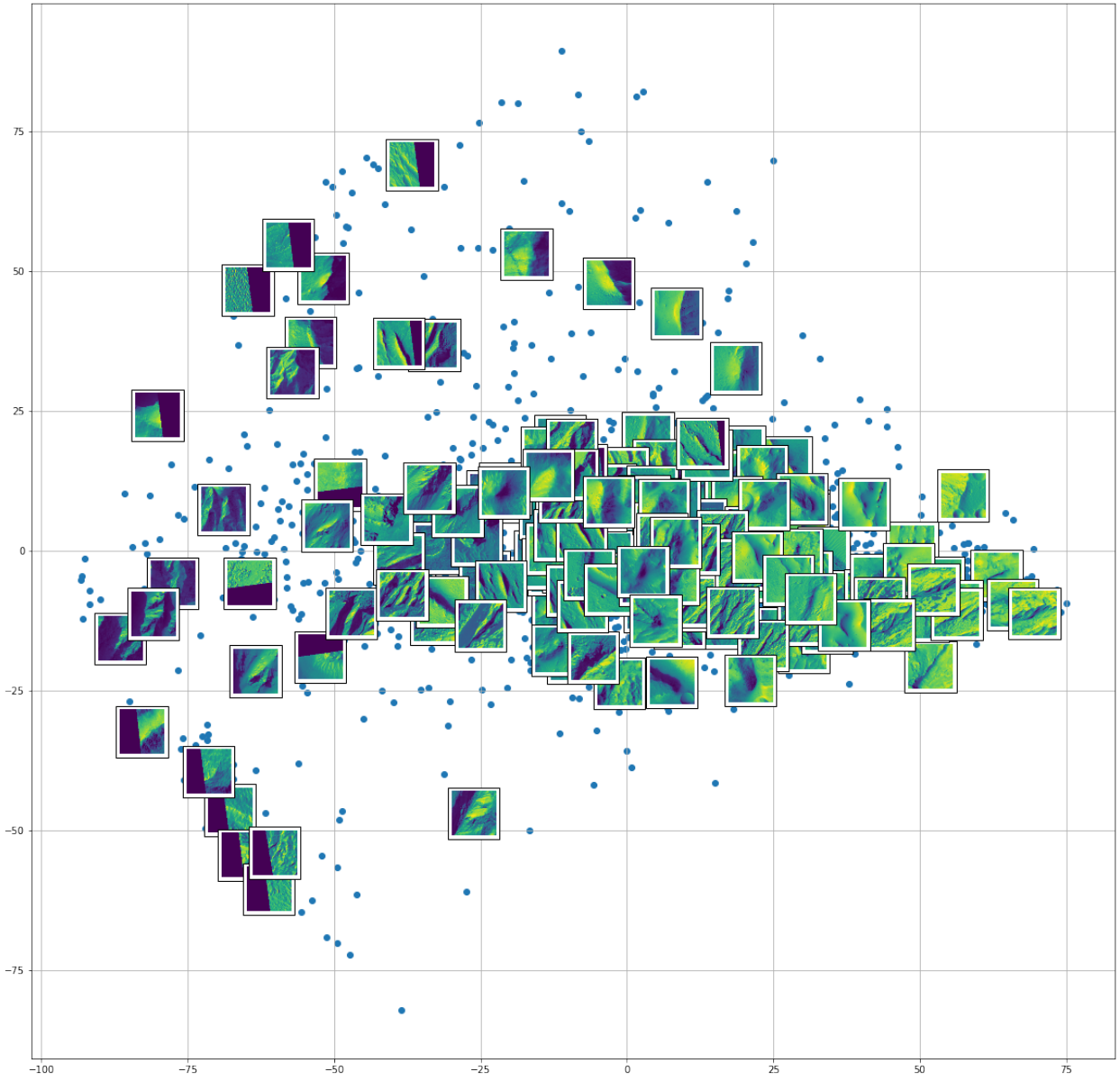
For the 37 streak images, I couldn't really apply epsilon-ISOMAP because there was only a few data points.

Here is the scatter plot for PCA on the streak images:



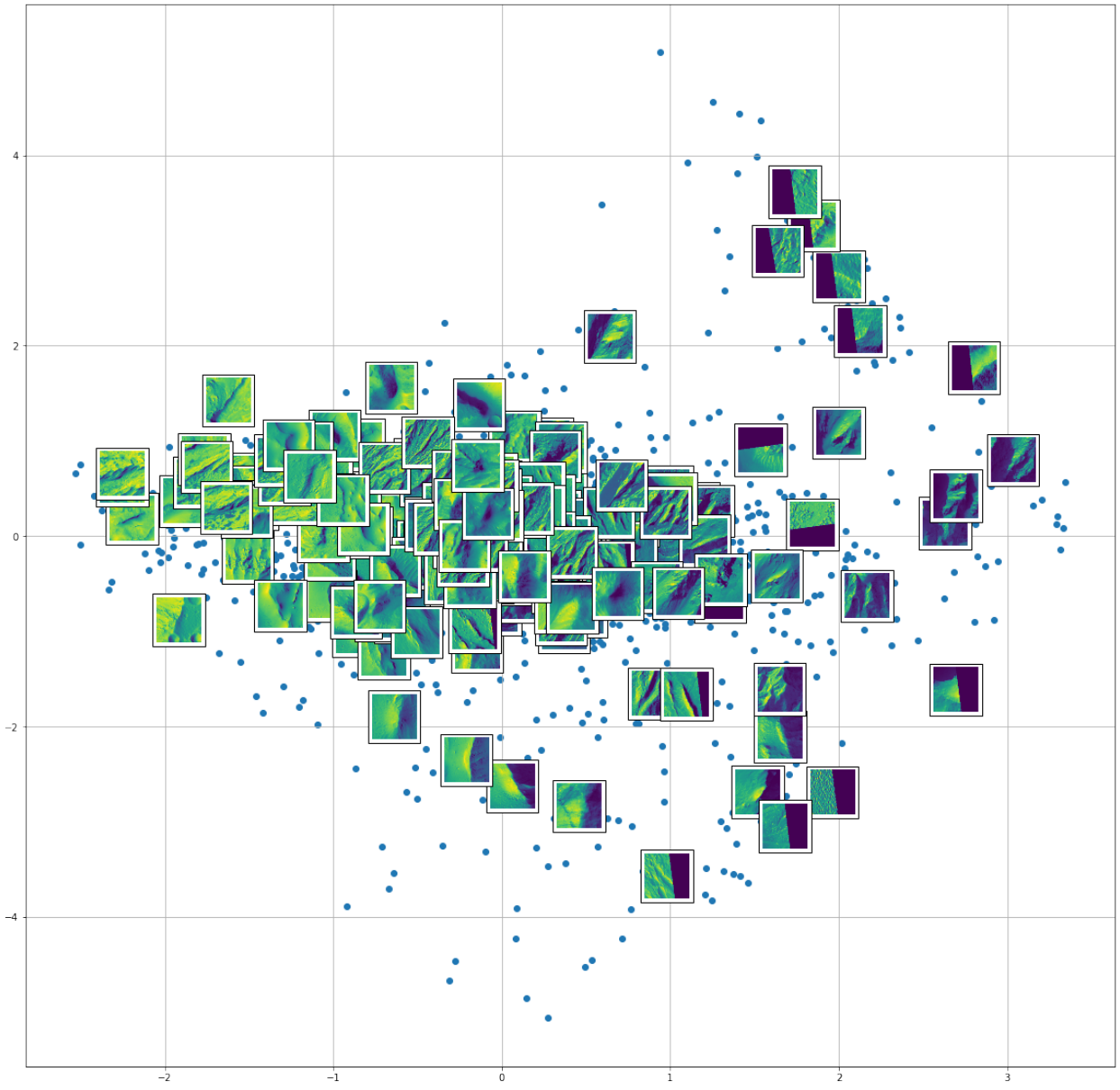
Finally, there is a set of images that are labeled 'other'. I would like to apply epsilon-ISOMAP and PCA to see if any interesting patterns or features emerge.

For the 2,018 'other' images, I used an epsilon value of 50. Here is the scatter plot for epsilon-ISOMAP on the 'other' images:



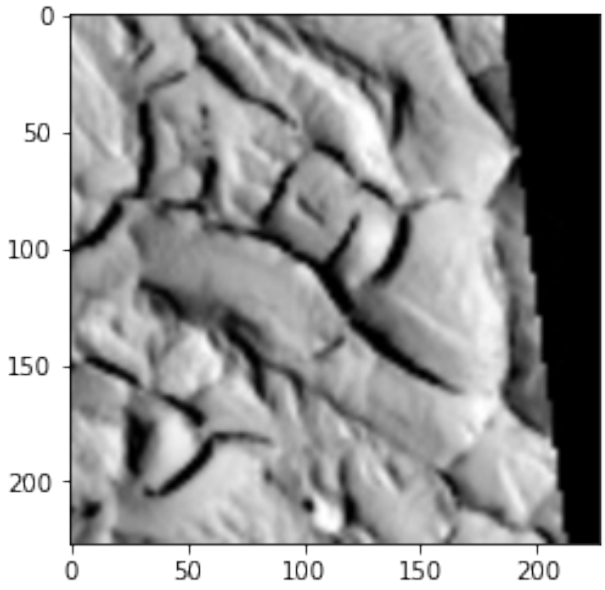
The images do seem to be clustered near images that are similar to them. For instance, on the left side, images that have straight edges appear clustered close to each other. The images in the middle appear darker, and the images on the right side appear brighter.

Here is the scatter plot for PCA on the 'other' images:

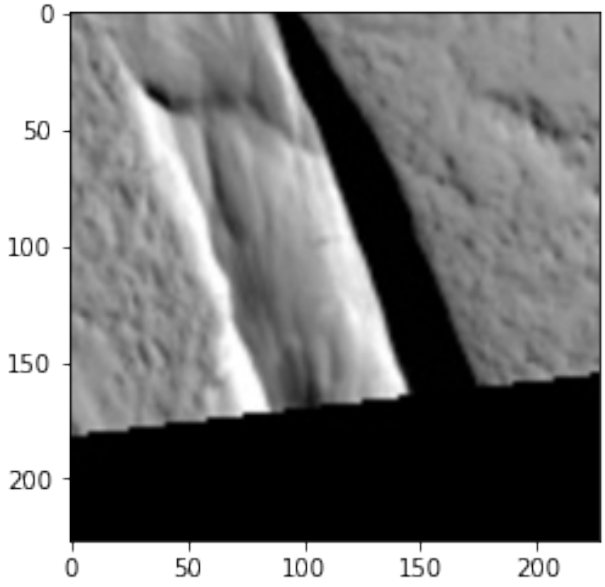


PCA appears to be performing a very similar clustering.

Using an epsilon of 55, I performed epsilon-ISOMAP and identified the isolated nodes as outliers. Here is one I found particularly interesting:

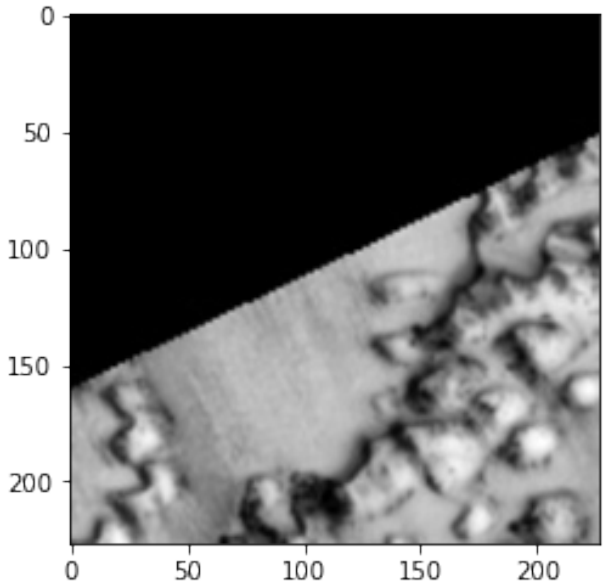


I reduced the epsilon value to get more isolated nodes. Here are some that I found particularly weird-looking:



The above image looks like that of a giant slab with strikingly straight edges.

This one looks like a bunch of bumps on human skin:



4 Classification

4.1 Methodology

Here, I would like to use classification models to distinguish between the four classes ‘crater’, ‘bright dune’, ‘dark dune’, and ‘streak’. I will apply logistic regression, k-nearest neighbors, linear SVM, kernel SVM, and neural networks.

4.2 Results and Evaluation

For KNN, I used cross-validation to choose k=3 neighbors. I got a test accuracy of 72.68%. Here are the confusion matrix and precision/recall/f1 scores:

```
[[67  2  3  1]
 [15 68  1  0]
 [ 1  0  3  0]
 [17  6  7  3]]
              precision    recall  f1-score   support

           1       0.67      0.92      0.77        73
           2       0.89      0.81      0.85        84
           3       0.21      0.75      0.33         4
           4       0.75      0.09      0.16        33

    accuracy                           0.73       194
   macro avg       0.63      0.64      0.53       194
weighted avg       0.77      0.73      0.69       194
```

For logistic regression, I got a test accuracy of 63.40%. Here are the confusion matrix and precision/recall/f1 scores:

```
[[56 11  4  2]
 [16 57  0 11]
 [ 0  1  2  1]
 [14  6  5  8]]
              precision    recall  f1-score   support

           1       0.65      0.77      0.70        73
           2       0.76      0.68      0.72        84
           3       0.18      0.50      0.27         4
           4       0.36      0.24      0.29        33

    accuracy                           0.63       194
   macro avg       0.49      0.55      0.49       194
weighted avg       0.64      0.63      0.63       194
```

For linear SVM, I used cross-validation to choose C=0.01. I got a test accuracy of 68.04%. Here are the confusion matrix and precision/recall/f1 scores:

```
[[57 11  4  1]
 [18 60  2  4]
 [ 0  1  2  1]
 [11  6  3 13]]
              precision    recall  f1-score   support

           1       0.66      0.78      0.72        73
           2       0.77      0.71      0.74        84
           3       0.18      0.50      0.27         4
           4       0.68      0.39      0.50        33

    accuracy                           0.68       194
   macro avg       0.57      0.60      0.56       194
weighted avg       0.70      0.68      0.68       194
```

For kernel SVM, I used the 'rbf' kernel and, using cross-validation, chose C=4. I got a test accuracy of 79.9%. Here are the confusion matrix and precision/recall/f1 scores:

```
[[58  7  5  3]
 [ 9 74  0  1]
 [ 1  0  3  0]
 [ 5  8  0 20]]
              precision    recall  f1-score   support

           1       0.79      0.79      0.79        73
           2       0.83      0.88      0.86        84
           3       0.38      0.75      0.50         4
           4       0.83      0.61      0.70        33

    accuracy                           0.80       194
   macro avg       0.71      0.76      0.71       194
weighted avg       0.81      0.80      0.80       194
```

For neural network, using two hidden layers of sizes 20 and 10, respectively, I got a test accuracy of 70.62%. Here are the confusion matrix and precision/recall/f1 scores:

```
[[53 15  4  1]
 [11 68  0  5]
 [ 1  0  2  1]
 [ 9  9  1 14]]
              precision    recall  f1-score   support

           1       0.72      0.73      0.72        73
           2       0.74      0.81      0.77        84
           3       0.29      0.50      0.36         4
           4       0.67      0.42      0.52        33

    accuracy                           0.71       194
   macro avg       0.60      0.61      0.59       194
weighted avg       0.71      0.71      0.70       194
```

Let's compare the performance of each model by looking at a table:

Model	Test accuracy	Average f1 score
KNN	72.68%	69%
Logistic regression	63.40%	63%
Linear SVM	68.04%	68%
Kernel SVM	79.9%	80%
Neural network	70.62%	70%

As you can see, the best-performing model, in terms of test accuracy and average f1 score, is kernel SVM, then KNN in second place, and neural network in third place. Due to the non-linearity of the data, the non-linear models kernel SVM,

KNN, and neural network perform the best while linear models like linear SVM and logistic regression perform the worst. Furthermore, the streak images had the lowest precision score for each model, and the bright dune images had the lowest recall score for each model. The low precision score for streak images might be explained by the low number of streak images in our dataset–there were only 37 of them.

5 Conclusion

For the dataset consisting of craters, dark dunes, bright dunes, and streaks together, epsilon-ISOMAP and PCA performed very similar clusterings. For the dataset consisting of images labelled 'other', epsilon-ISOMAP and PCA performed similarly. For the datasets consisting solely of dark dunes or solely of bright dunes, epsilon-ISOMAP appeared to perform slightly better than PCA. Some images were singled out as outliers based on the adjacency matrix used in epsilon-ISOMAP. Finally, several classification methods were applied to the images, and the non-linear models outperformed the linear models, suggesting that the geometry of the image dataset is non-linear.

References

[1] Kiri L. Wagstaff, You Lu, Alice Stanboli, Kevin Grimes, Thamme Gowda, and Jordan Padams. Deep mars: Cnn classification of mars imagery for the pds imaging atlas. *Proceedings of the Thirtieth Annual Conference on Innovative Applications of Artificial Intelligence*, 2018.