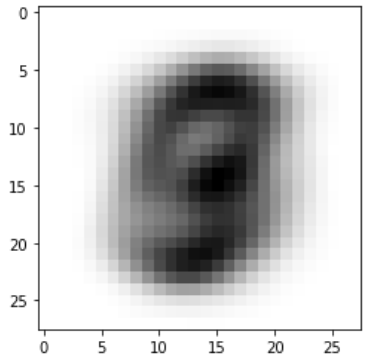
Alper Canberk Balcı
2017400087

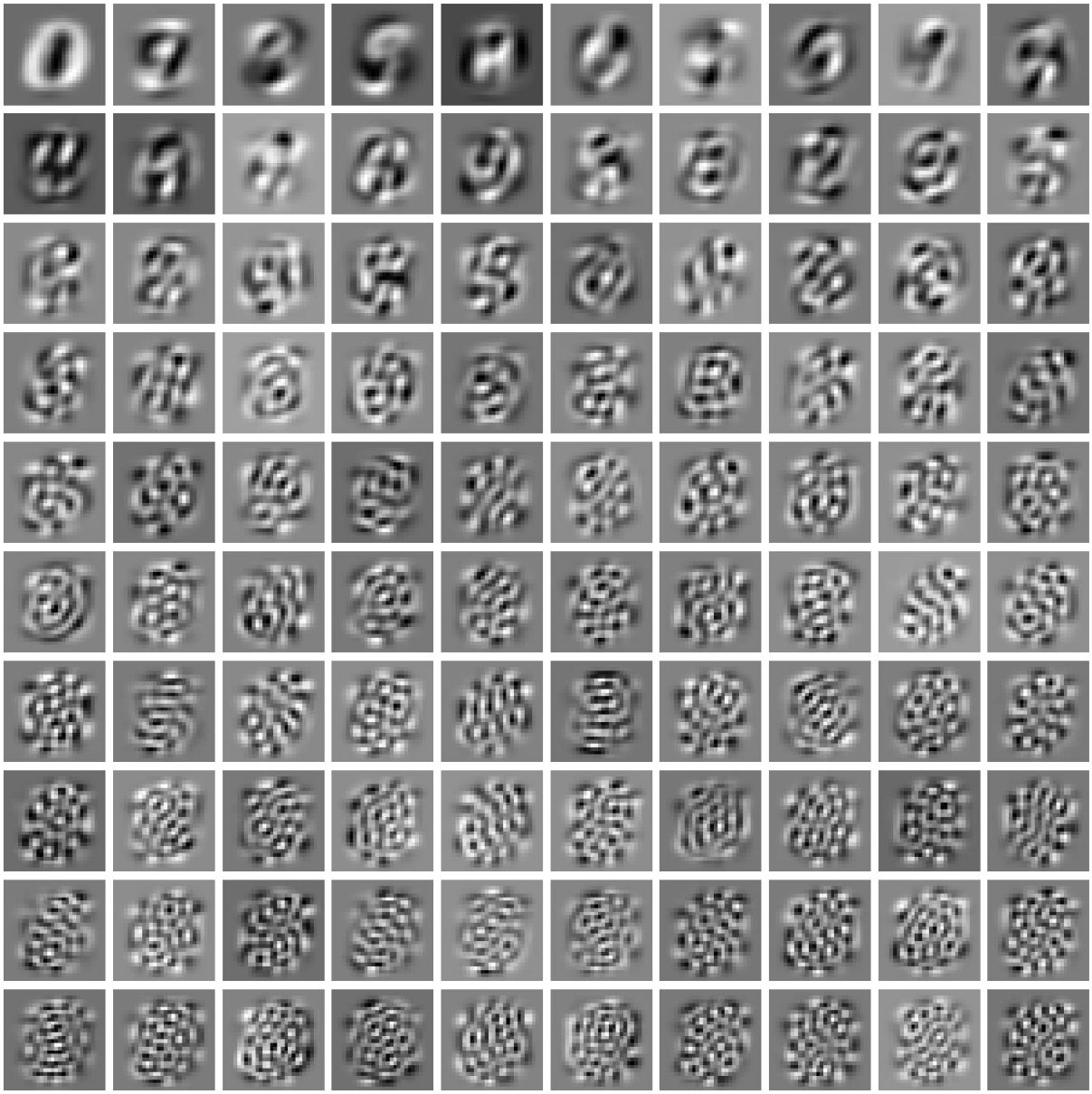
CMPE 481 Project 3

1 - Plot 10 sample digit images per digit class.

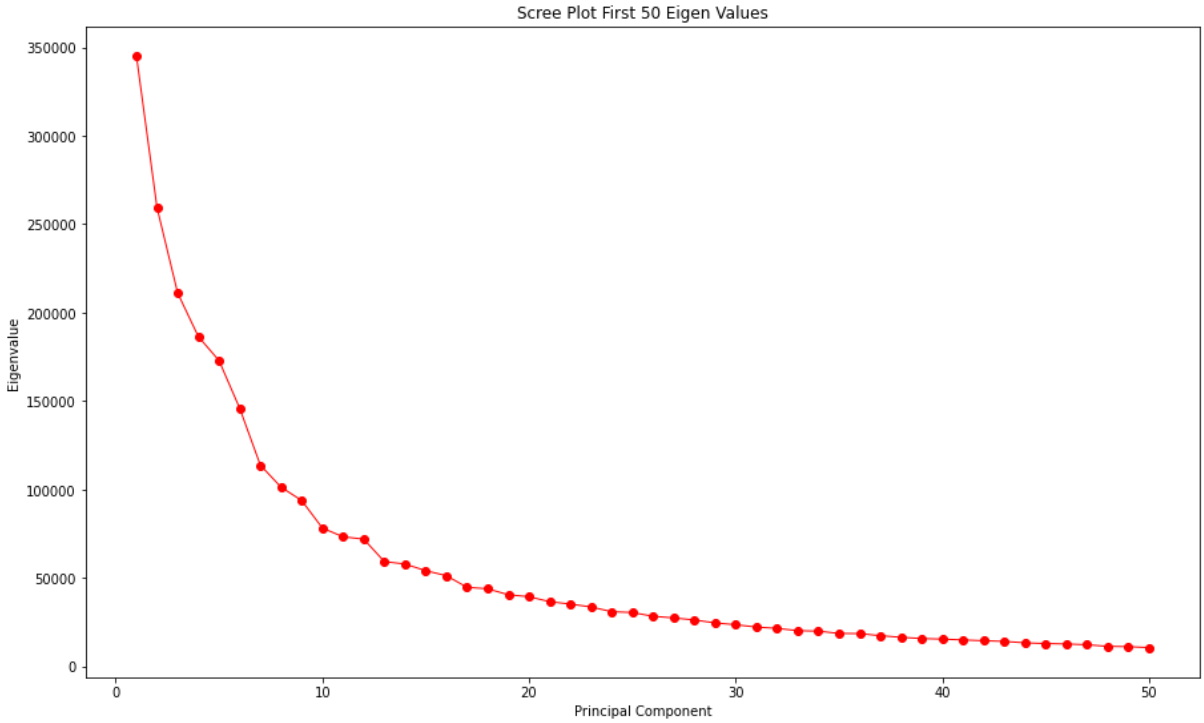
2- Using the MNIST training set: 1) plot the mean image, 2) eigenvectors, and 3) eigenvalues



Alper Canberk Balcı
2017400087



Alper Canberk Balcı
2017400087

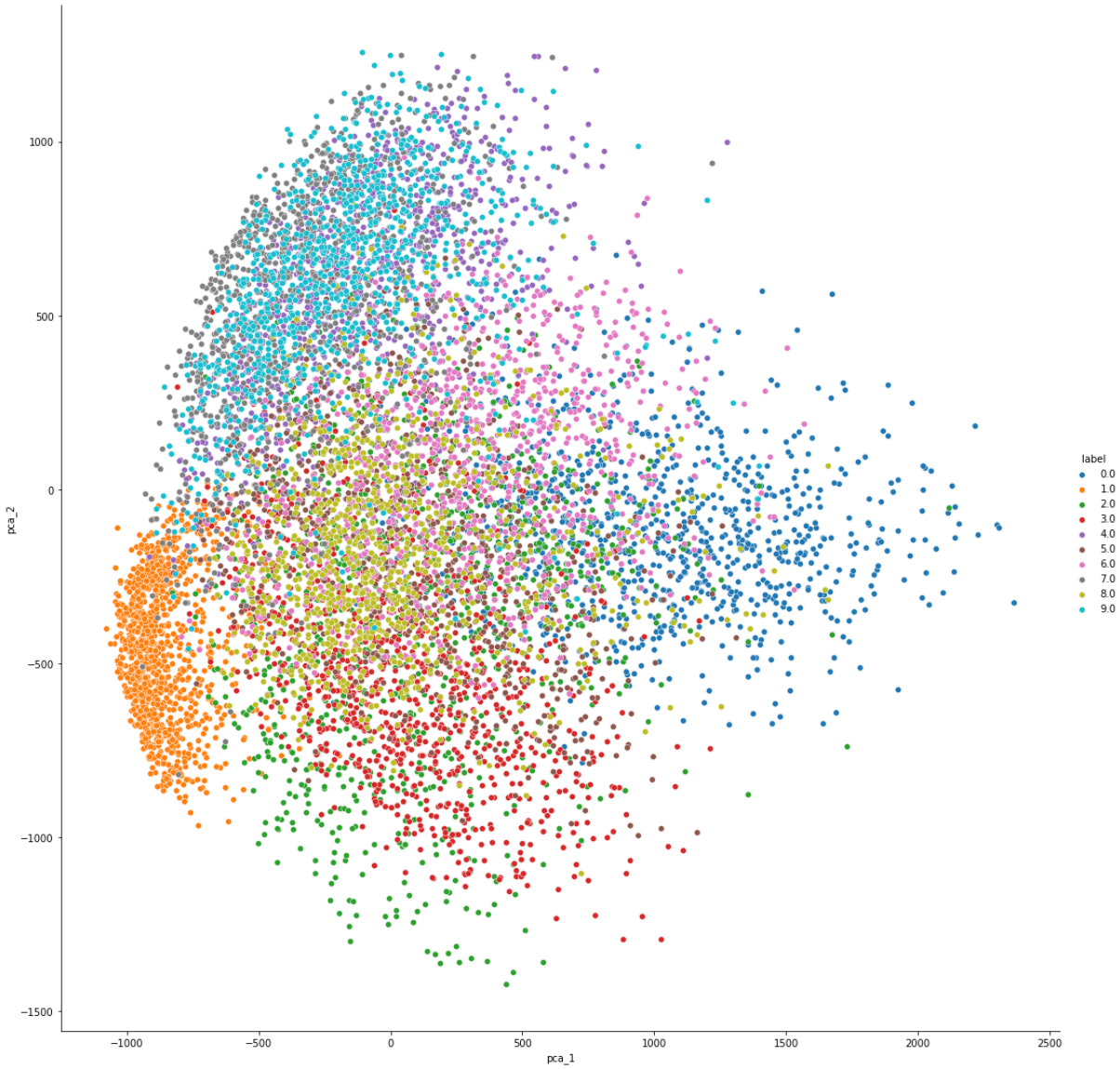


3- Using the MNIST test set, reduce the dimensionality of features to two for PCA visualization. Plot the digits in 2D

Image below, in the next page, to keep quality of the image. Here, I wrote observations on the plot.

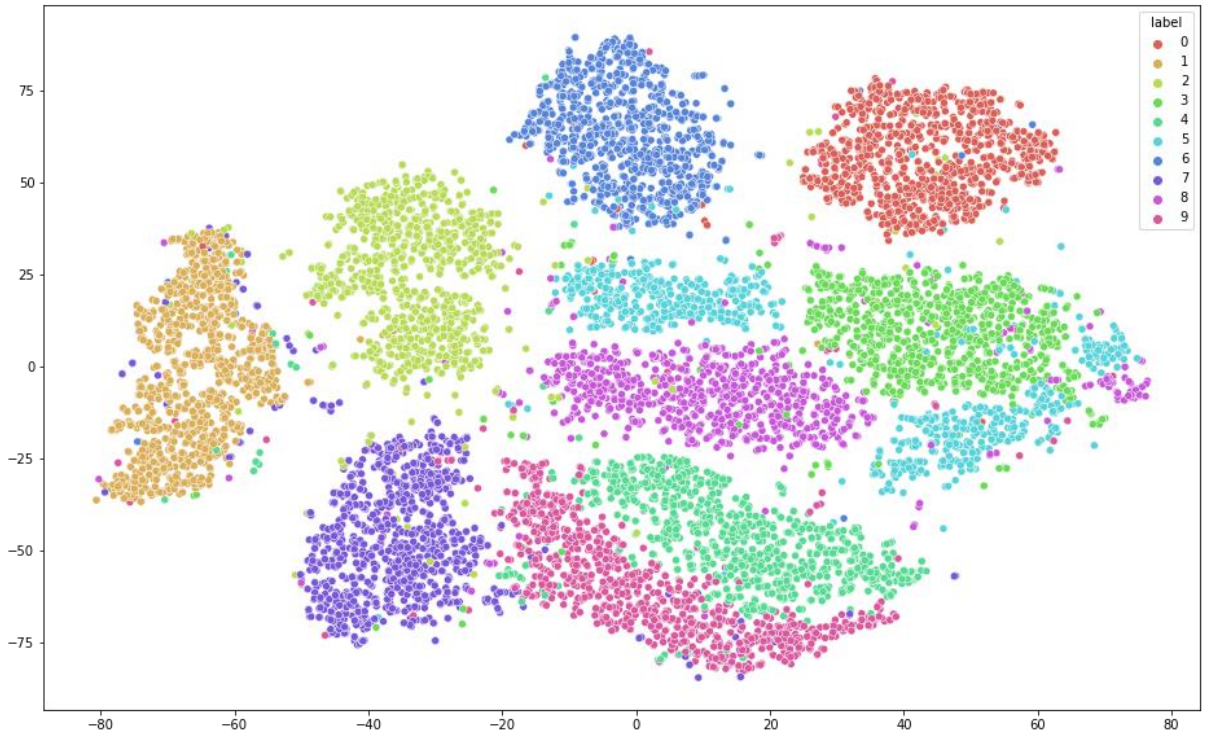
PCA is not as strong as t-SNE in visualization manner, but, we can see obvious patterns, next to not so obvious other patterns for digits dataset MNIST. We can see that 0 and 1 are at the opposite ends of the graph. As someone who worked on even neural networks I can say that, anyone who worked on regression methods can simply argue that 1 and 0 are the opposite of each other in terms of pixel brightness, as they easily fit on each other like a filter. 8 is in the center and close to everybody. There are other similarities between digit representations by drawers as well. On the other hand, we can argue that some of the digits' drawings have bigger variance than others. For example, 1 has very dense representation on the data.

Alper Canberk Balcı
2017400087



4- Use t-SNE to visualize MNIST test set (used sklearn):

Alper Canberk Balcı
2017400087



5- Explain the fundamentals of the t-SNE approach and compare it to the PCA. What are the advantages and disadvantages of t-SNE approach compared the PCA?

t-distributed Stochastic Neighbor Embedding is a tool to visualize a high dimensional data with low dimensions, using feature extraction. T-SNE will maximize the distance in two dimensional space between observations that are most different in a high dimensional space. Because of this, observations that are similar will be close to one another and may become clustered. This way seeing the different clusters in the data becomes easier. The basic concept is that t-SNE finds a way to project data into a low dimensional space so that the clustering in the high dimensional space is preserved. How?

To simply explain, it puts the points in a random order, from there, t-SNE moves these points, a little bit at a time, until it has clustered them. A point wants to move closer to its partners in that cluster, but at the same time, there are other points from other clusters, which are far away from the point we want to move, so they push back the point, while its partners from its cluster attract the point. At each step, a point is attracted to points it is near, and repelled by points it is far from. The details of the measurement of the distance and finding similarities are complex.

On the other hand, PCA is also a tool like t-SNE for dimension reduction. It works on the high dimensional space such that it finds the features(dimensions) with the most contribution to variance via covariance matrix and its eigen values, which represent the variance, and eigen vectors, which are the directions of the variance. Then PCA projects the data onto these principal component vectors (eigen vectors). This way we can see not only the clusters in the data on 2D or 3D, but also in higher dimensions that are significantly lower than the original data is in, that cannot be visualized due to the fact that we live in a three dimensional space (tragic).

PCA helps lowering the dimensions of a data, and it can be accompanied with t-SNE to visualize the data better. They could complement each other. T-SNE is more powerful but more complex method for visualization but, lowering the dimensions first helps immensely to t-SNE.

Alper Canberk Balcı
2017400087

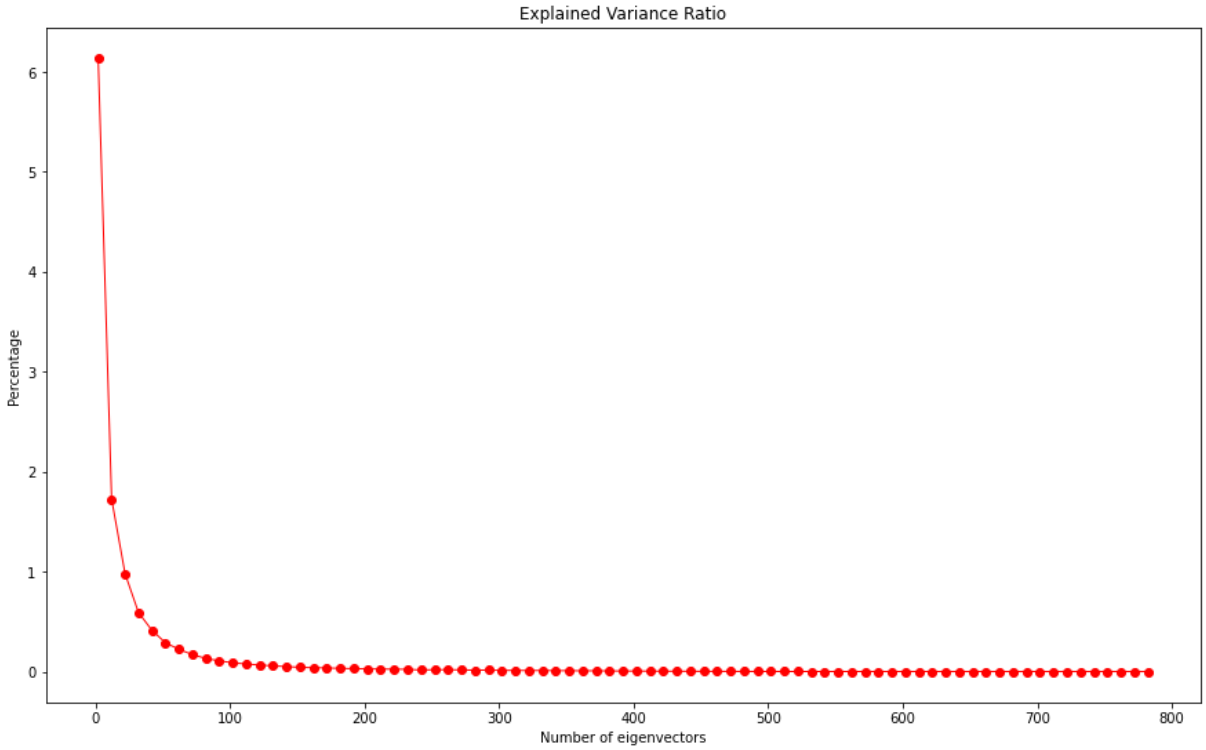
PCA is a linear dimensionality reduction technique, however, t-SNE is a non-linear one. PCA moves the data as a whole. It tries to preserve the global structure of the data. However, t-SNE tries to preserve the local structure of data. PCA is a deterministic algorithm, however, t-SNE is a non-deterministic or randomized algorithm.

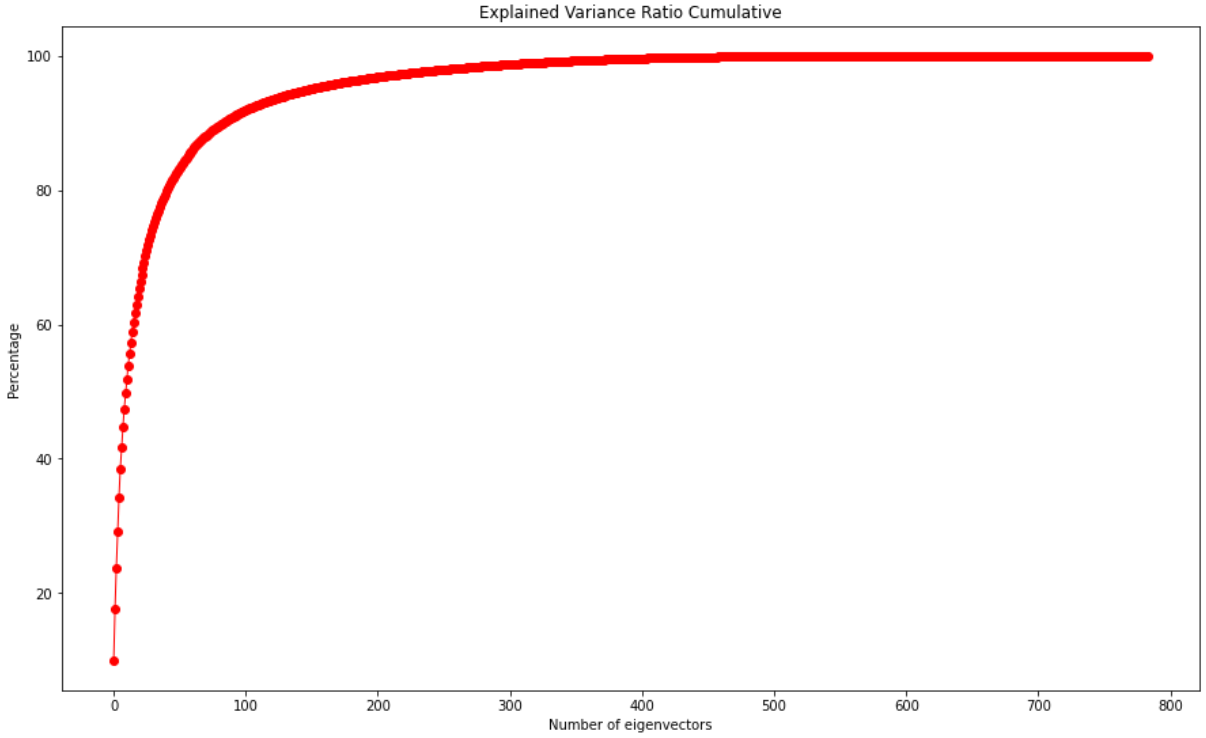
With PCA we can find an optimal number of eigen vectors to transform the data into a low dimensional one, as marginal increase in additional eigen vectors decrease with high number of it. This way we can preserve variance among the most important features. T-SNE cannot do this. It can preserve distances. But, it is better in dimensionality reduction and visualization than PCA.

6- Reconstruction:

The explained variance ratio:

Alper Canberk Balcı
2017400087





Via elbow method, we can assume between 130-150 seems like a great choice. Also, we can take into account the leading k components that explain more than, for example, 95 percent, of the variance, and that is 149:

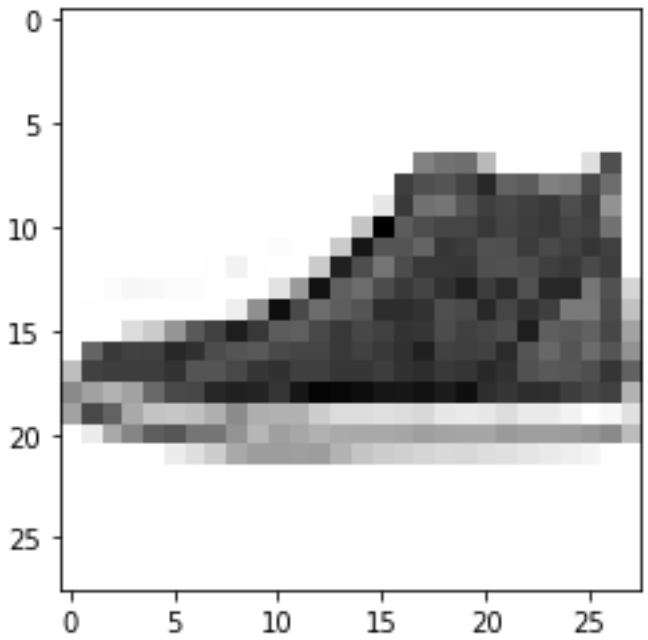
Alper Canberk Balcı
2017400087

In [114]:
```
images = []
# Explained Variance Ratio
sum_of_eigen_values = int(sum(eigen_values))
percentages = []
for i in range(2,783,10):
    percentage = 100*eigen_values[i]/sum_of_eigen_values
    percentages.append(percentage)

percentage = np.array(percentages)

cumul_percentages = []
cumul_percentage = 0
star_point = 0
for i in range(0,784):
    percentage = 100*eigen_values[i]/sum_of_eigen_values
    cumul_percentage += percentage
    cumul_percentages.append(cumul_percentage)
    if cumul_percentage < 95.01:
        star_point = i+2

mean = np.array(df_test.mean())
for i in range (2,783,10):
    final_df_test = np.matmul(eigen_vectors[:i], X[128])
    # reconstruction
    rec_ = np.matmul(final_df_test, eigen_vectors[:i])
    rec_ = rec_ + mean
    images.append(rec_)
```

In [120]:
```
print("for 95% explained variance ratio, number of eigen vectors:", star_point)
```

```
for 95% explained variance ratio, number of eigen vectors 149
```

I chose mnist fashion dataset, which can be found under misc folder.



This is the image I am using for non-digit image. I use the eigen vectors of digit dataset. So the graphs are not copy pasted again. Below is the reconstructed images

Alper Canberk Balcı
2017400087