Fashion MNIST
KNN classification with PCA data

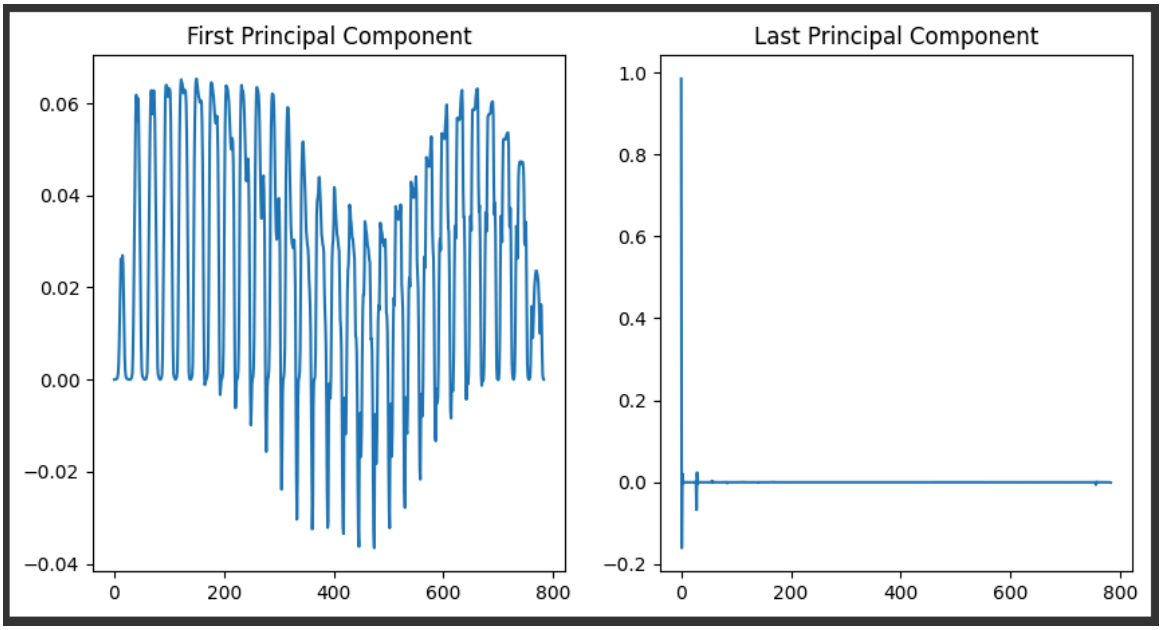
Name: Swathi Baskaran

PCA Overview: Principal component analysis (PCA) is a dimensionality reduction technique that can be used to reduce the number of features in a dataset. It does this by finding a new set of features, that are linear combinations of the original features. The first PC captures the most variance in the data, and each subsequent PC captures the remaining variance.

1. PCA prioritizes features that explain the most prominent variations in the data. In the context of the Fashion MNIST dataset, the PCs would capture the most important variations in the images, such as the overall brightness, the presence of certain textures, or the shapes of the clothing items.

The plot in the output shows that the first principal component (PC) has a large positive value in the center of the range and then tapers off towards the edges. This suggests that the first PC captures the overall brightness of the images. The last PC has a much smaller magnitude and appears to oscillate around zero. This suggests that the last PC captures less important variations in the images, such as noise.

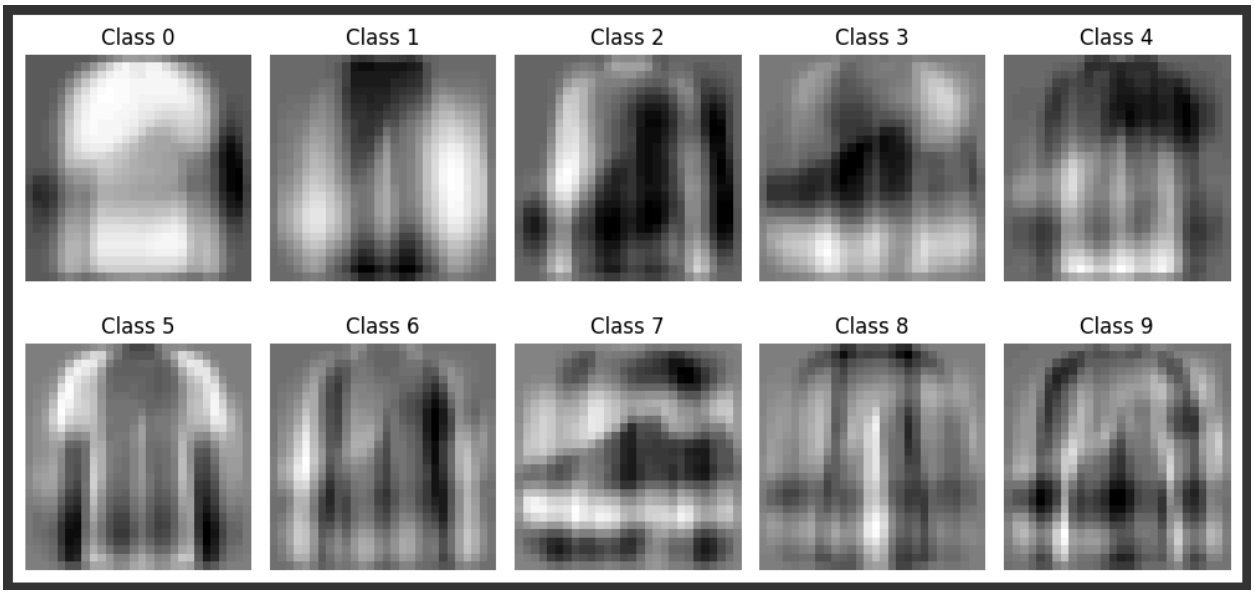
Here is the screenshot of the plots:



A brighter image, on average, will have higher pixel intensities across all 784 pixels. When PCA computes first PC, it essentially finds a weighted linear combination of the original features (pixels) that best captures the spread of the data points. Since brighter images tend to have higher average pixel values, giving positive weights to all pixel features in first PC makes sense. This explains the large positive value in the center of the first PC plot. As we move towards the edges of the plot (corresponding to considering individual pixel features more heavily), the influence of any single pixel on the overall brightness diminishes. This leads to the tapering off of the first PC values towards the edge.

The lower magnitude of last PC indicates it captures features with less variance compared to first PC. Since most clothing items likely have a dominant background (e.g., white background for a shirt), the pixel intensities wouldn't deviate significantly from the average brightness captured by first PC. The high-frequency oscillations in last PC suggest it might be capturing residual variations with smaller magnitudes, like Texture variations within the clothing material, small wrinkles, or subtle imperfections. Since these details likely have minimal impact on distinguishing broad clothing categories (e.g., trousers vs. sandals), they contribute less to the overall data variance and get captured by later principal components with lower magnitudes like fast PC.

2. Plot for the first of each of the digital garment classes:



Each class might have some inherent visual properties that get captured when the data points from that class are projected onto the principal components. The reconstructed image for a particular class would then reflect these inherent properties based on the weights assigned to the principal components during projection. The reconstructed image for a class like "t-shirt" might emphasize brightness and overall shape, as these could be prominent features for t-shirts in the Fashion MNIST data. Similarly, the reconstructed image for "trousers" might focus on capturing elongated shapes along the vertical dimension.

3. Used the PCA representation of the Fashion MNIST dataset as the new dataset. From Project 1, used Kernard stone algorithm to split train and test data with optimal N= 5810, then performed kNN classification on the new data with k=20 with metric "Manhattan".

The overall accuracy = 86.18 %
Accuracy in Project 1 = 84.96 %
So the accuracy is overall improved with PCA.

Here is the class-wise accuracy. This also looks better overall.

Accuracy Project 2	**86.18%**
Accuracy Project 1	**84.96%**
	Accuracy
Class 0	89.33%
Class 1	96. 05%
Class 2	79.83%
Class 3	86.97%
Class 4	78.15%
Class 5	83.78%
Class 6	61.01%
Class 7	95.46%
Class 8	94.37%
Class 9	95.29%

4. Overall PCA helped with my classification. It improved my classification from 84.96% to 86.18%. That's actually quite a good improvement. The accuracy of each class also looks better. Out of 10 classes, 8 classes are above 80% and all the classes are above 78%.

The raw pixel data in Fashion MNIST might contain redundant or noisy information that can hinder KNN's performance. PCA helps remove this redundancy by focusing on the most informative variations in the data. In essence, PCA might have extracted features like overall brightness, shapes, and edges, which are likely more relevant for distinguishing clothing categories.
PCA projects the original data into a lower-dimensional space while preserving the maximum variance. This can help in focusing on the most informative features and reducing the impact of noise and redundant information present in the original high-dimensional data. PCA transforms the features into orthogonal (uncorrelated) components. This can help in improving the

performance of KNN, especially when the original features are highly correlated. By reducing multicollinearity, PCA can make the data more suitable for KNN classification. With fewer dimensions after PCA, there's a lower risk of overfitting the model to the training data. This can lead to better generalization performance on unseen data, resulting in improved accuracy.

There's often an optimal number of components that maximizes the information retained while minimizing the dimensionality. Experimenting with different values of **n_components** and evaluating the performance can help identify the optimal number for your dataset. Accuracy may decrease if you choose a value of n_components that is too low or too high for your dataset, leading to either underfitting or overfitting. So I used the **n_components = 188.**

I tried n_components= 784 and got 81.17% accuracy. With very high dimensionality (all 784 components), KNN can suffer from the "curse of dimensionality." This means that distances between data points become less meaningful in high dimensions, making KNN less effective. The, I tried with 350 exact half 784, that gave out accuracy about 83.84%. Then tried with 200, 196 and those gave 85. 26% and 85.66% respectively. I tried values above and below 188, then concluded with 188 as it gave the best accuracy comparatively.

n_components	Accuracy
784	81.17%
350	83.84%
200	85.26%
196	85.66%
189	85.70%
188	86.18%
187	85.99%
180	85.70%

As for the class wise performance, overall there is increase. Classes 0, 2, 3, 4, and 6 show more significant improvement. This suggests that PCA might have been particularly effective in capturing the key features for distinguishing these clothing items (e.g., shirts, shoes, sandals). Classes 1, 5, 7, 8, and 9 have minimal improvement. This indicates that the original features (pixels) were likely already sufficient for accurate classification of these classes.

In conclusion, using PCA with an appropriate number of components can significantly improve KNN classification by extracting relevant features and reducing noise. The extent of improvement can vary depending on the class and the discriminative nature of the features already present in the raw pixel data.

Link for the code for Project 2: Project 2 -Link