IN3050/IN4050 Mandatory Assignment 3: Unsupervised Learning

Name: Niklas Alexsander

HOW TO RUN

To run the jupyter notebook you simply have to open the notebook and run the code in the same order as its read.
I would recommend using the "Run"-button in jupyter notebook :)

Principle of Maximum Variance: what is PCA supposed to do?

1. Variance is the difference/spread of a given data set from its mean value.

2. Covariance is how variables change together, if they are 'moving' in the same way with different parameters.

3. To calculate the covariance matrix of a given set X (Centered) we take the set X multiplied by itself transposed multiplied by the 1/N where N is the number of elements. 1/N(X.T * X).

4. The meaning of the principle of maximum variance is that the PCA is created in a way that makes the first k components get the maximized variance, and the last k elements as the ones with the minimized variance.

5. If we can maximize the spread of the data, we may be able to preserve useful information! This information can be used to visualize, compress and decompress our data.

6. The principle does not always apply. For example when all the PCA components have high variance, there will be no 'guarantee' that we get a small loss of data in the compression. Knowledge of the domain is really important with PCA.

Plotting before and after PCA

data_with_labels1

When first plotting the data set we can see that the scattered plots are close to a 'linear regression line' close to the x (x-axis) value of 4. But when we run the pca first and then apply the labels to see what has changed, we see that if we think about a linear regression line in our pca the line would be in the center of 0! The pca-algorithm managed to find the separation of the plotted points classes without knowing/'training' with the labels!

data_with_labels2

Here the results are totally different, the plots are all over the axis, and not really separable by using the x-axis of 0. But if we change the P[:,0] to [P:,1] the plots gets more even over the x-axis and we are able to see a better difference between the plotted labels at x-axis 0. The spread of P[:,0] looks a bit 'left-sided' compared to the more balanced P[:,1].

Considering using both eigenvectors at the same time does look like a fair challenge to find any correlation between the two vectors. By looking at the first vector we can see that it's more centered around 0, more grouped points, compared to the plots of the second vector. So I would assume that the greater spread of vector 2 would give a better result, and it does when we take a look at the labels.

Case study 1: PCA for visualization

While plotting I used the feature x=1 and y=3 (feature 2 and 4). While looking at the visualization of the two features before pca we see that there is a clear difference between the green-blue'ish and yellow class to the purple class. My thoughts before PCA was that this difference should be visible also after the PCA.

When running the PCA and plotting the results the purple class gets pushed all the way to the left by itself, a good reflection of the plot before. The green-blue'ish class is together with the yellow class and is close to being differentiated by the x-axis point 1. There is a clear difference between the three classes even after the PCA!

Case study 2: PCA for compression

Inspecting the reconstructed data

After compressing and decompressing the 'image' the face is still present, not quite as sharp as the original, a lot darker but still visible. So most of the data got 'reconstructed' after decompression, but it's no 'lossless' recreation. It seems that the most important information/data about the image got 'through' the PCA since the facial features and structure is still present after compression and decompression.

Evaluating different compressions

When compressing and decompressing an image with more and more dimensions the decompression delivers an image that is closer and closer to the original, meaning that most of the data gets reverted back to the original, ofc not totally lossless.

K-Means Clustering

By increasing the k-value of K-Means the algorithm starts with k number of centroids at random locations in the data set, and starts iterating, going closer and closer to a 'cluster' in the data. With each cluster added the k-means algorithm 'finds' new 'groups' in the data. It is therefore important to have knowledge of the domain before picking the amount of clusters. The k=5 doesn't look any more wrong than the correct k=3. However, increasing the cluster number and seeing some clear difference could be a pointer in the direction that there is maybe another way of looking at the data.

Quantitative Assessment of K-Means

As the k-value increases, so does the accuracy as well, with an exception at k=4 which could be an effect of the given data set. The reason for the increasing accuracy comes from what the k-value actually does. the increase of k means that we are adding a centroid/cluster/label to the dataset, which gives the classifier more labels to chose from and more information to work on, not just a 0 or 1 labeling, and thus the chance of finding a cluster that fits the data increase.