Seminar Data Science Coursework C

Sharwin Bobde (5011639),
Aditya Kunar (5074274),
Pavel Hoogland (4450892)

March 2020

1 Task 1 Principle Component Analysis

1.1 Description

Principle Component Analysis is a method that serves to reduce the dimensionality of a dataset by doing a transformation of the data to obtain a set of linearly uncorrelated 'Principal Components'. Where the First Principal component has the highest variance and each subsequent component has maximum variance relative to the previous one.

1.2 Implementation

The First algorithm in order to compute the Principal Components is based on the following steps: (1)

- Computing the 'center' of the points by computing the mean $\frac{1}{m}\sum_{i=1}^{m} x_i$
- Compute the set of centered points Y in $\mathbb{R}^{n\times m}$ from the data X with $y_i = x_i - \bar{x}$
- Compute the covariance matrix C through $C = \frac{1}{m}YY^T$
- Compute eigenvalues λ_i and eigenvectors u_i of C such that $Cu_i = \lambda_i u_i$

Alternatively for $m << n$ the *method of snapshots* can be used. The advantage of this method is that you only need to do the eigenvalue decomposition for a $m \times m$ matrix instead of a much larger $n \times n$

For this second algorithm you do the following steps: (2)

- Compute the center and centered points as in the First Algorithm.
- Compute the $m \times m$ Gram matrix G through $G = Y^TY$
- Compute eigenvalues λ_i and eigenvectors ϕ_i of $\frac{1}{m}G$ such that
- Compute the basis vectors u_i through a mapping $u_i = \frac{1}{\sqrt{\lambda_i}}Y\phi_i$

1.3 Results

Performing algorithm (1) to 2 dimensions on an iris dataset that looks at different classes of iris plants gives the result in figure 1

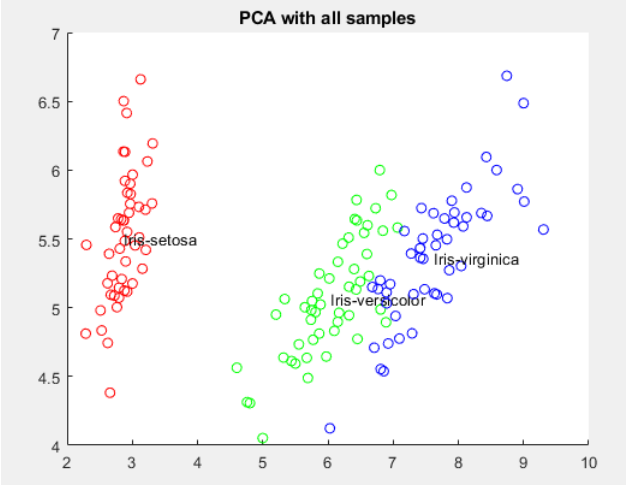


Figure 1: Algorithm (1) performed on Iris dataset

Performing algorithm (1) to 2 dimensions on an Arrhythmia dataset that analyses heart Arrhythmia gives the result in figure 2. The results show some correlation, but the distinguishability of the classes is difficult

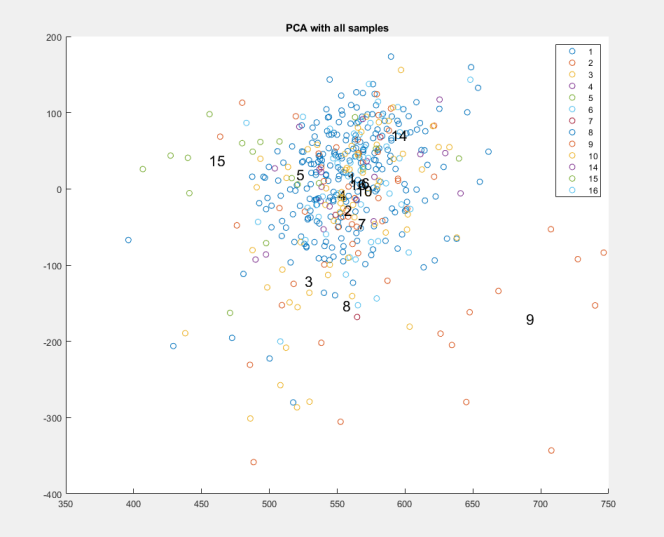


Figure 2: Algorithm (1) performed on Arrhythmia dataset

Performing algorithm (2) to 2 dimensions on the iris dataset gives the result in figure 3 The points have been given colors to distinguish the different plants from each other. From the figure it is visible that the PCA shows a fairly good seperation of the different classes.

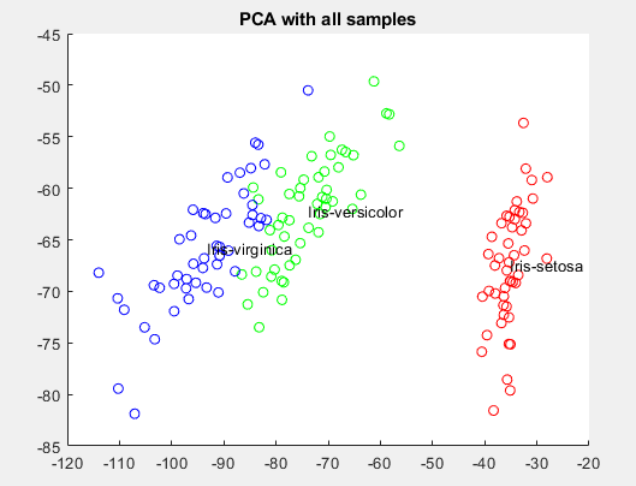


Figure 3: Algorithm (2) performed on Iris dataset

Performing algorithm (2) to 2 dimensions on the Arrhythmia dataset gives the result in figure 4

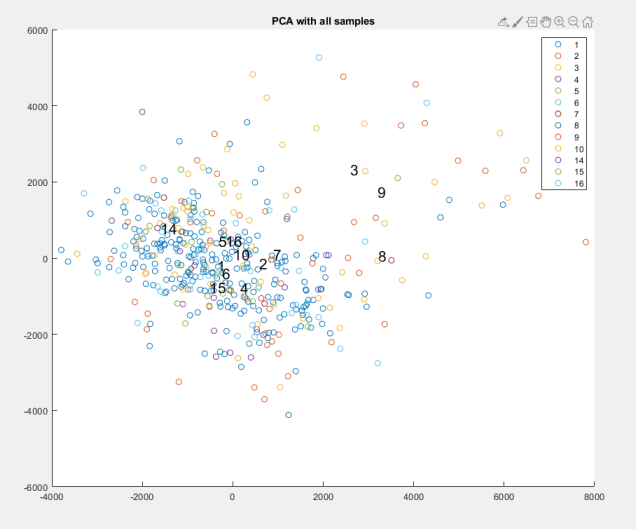


Figure 4: Algorithm (2) performed on Arrhythmia dataset

1.4 Tests for correctness

The following tests were performed:

1. We made sure that the dot product of the eigen vectors was 0 to check that they were indeed orthogonal.
2. We compared our results with that of the built-in functionality provided by Matlab.
3. We computed the covariance of the output matrix matrix after the transformation to see if the features were no longer mutually correlated.
4. We compared both methods with each other as well. When all samples were used we observed similar structures but inverted.

2 Task 2 Approximations To PCA

2.1 Snapshot PCA

When a dataset is too big it is computationally expensive to compute the covariance matrix or the Gram matrix. Moreover, the methods stated above have enormous space complexity due to covariance matrix and eigenvalue decomposition. One way of solving this is to Compute the mappings from higher dimentions to lower dimentions only using a small sample set from the actual data. The question arises that how one should sample these points for computing the mappings, we have shown two methods of doing this.

2.1.1 Methods

In this section we have presented our methodology and the results for this task. We compute PCA using the Gram matrix method for the following. We will show the results for the MNIST dataset and Arrhythmia dataset. The Mnist dataset has 60,000 instances with 784 features each. We could not show the complete MNIST dataset mapping using normal PCA because of memory constraints, he have shown the class distribution in Fig. 5. The class distribution of the Arrhythmia dataset and its PCA mapping is shown in Fig. 6, the dataset has 452 instances with 278 features. It is important to note that class labels 11, 12 and 13 for Arrhythmia were not present in the dataset because they were removed by the dataset creators.

- **Snapshot selection using Random Sampling:**
  A natural direction was to sample data points randomly. This lead to a simple implementation but with certain limitations. The results for MNIST can be seen in Fig. 7. There was no constraint on having at least 2 points of each class to find the Gram matrix. this with a combination of high class imbalance lead to its failure to produce results for the Arrhythmia dataset. Moreover, we can see that the class distribution for sampling 0.07% and 0.1% is highly imbalanced. in Fig. 7 (a, b).

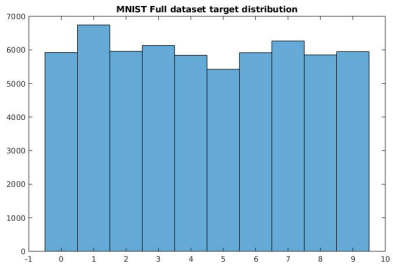
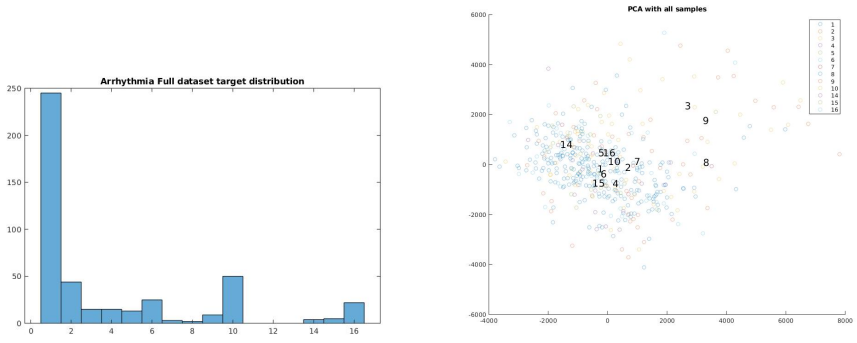


Figure 5: Class label distribution for the MNIST dataset



(a) (b)

Figure 6: Arrhythmia dataset class distribution and full PCA

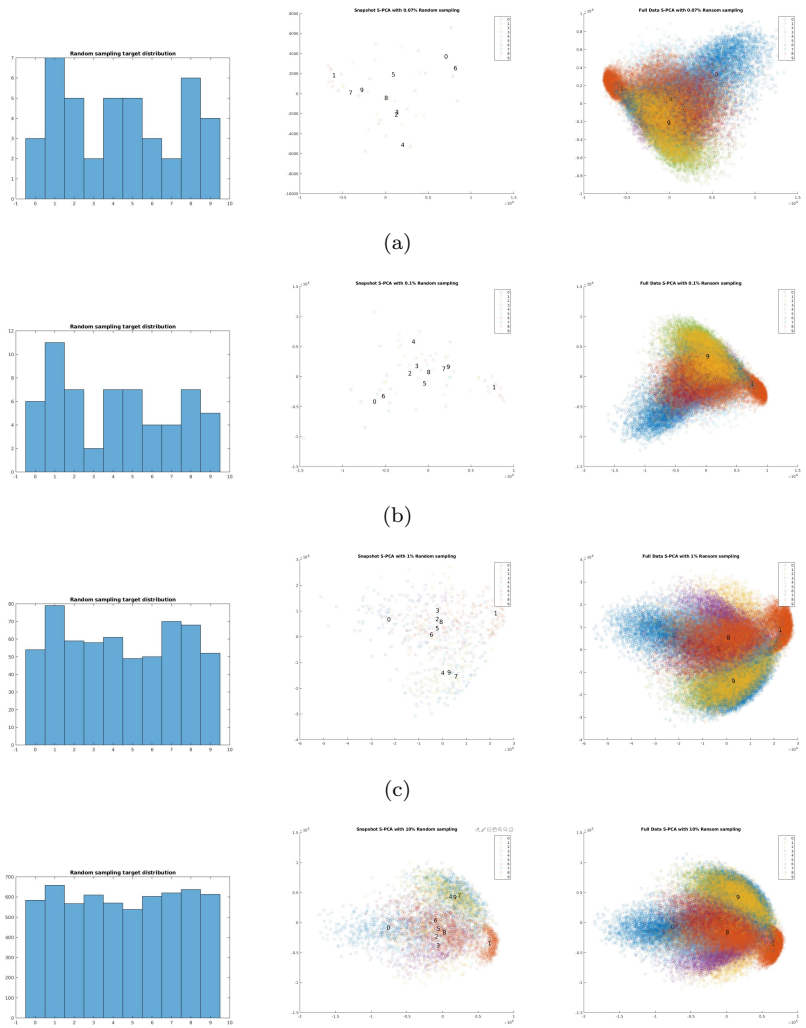


Figure 7: Snapshot PCA on MNIST dataset using *Random Sampling*. Every row shows the class distribution of the samples, the samples selected in the snapshot for calculating the mapping and the 2D embedding of the full dataset. The sampling rates are a) 0.07%, b) 0.1%, c) 1%, d) 10%. *(Class labels are plotted at the means and the data samples are hollow circles with alpha 0.3).*

- **Snapshot selection using Consistent Sampling:**
  As an improvement we devised an algorithm to sample consistently from each class. If the class distribution leads to sampling less then 2 samples then it automatically selects 2 samples. We can see that the class distribution for sampling 0.07% and 0.1% for MNIST in Fig. 7 (a, b) is very balanced as opposed to the random sampling.

In both cases we observe that as we take a bigger snapshot of the data, we get a better 2D embedding. We also see that with greater sampling the mean of more distinguishable classes becomes more prominent and the variance decreases, as is expected.

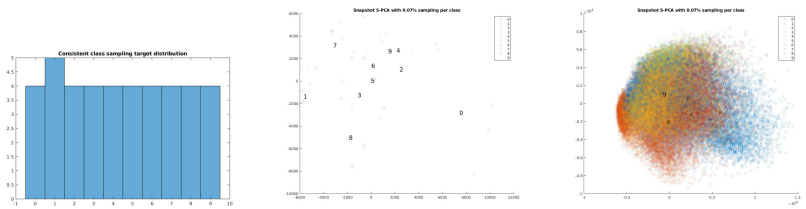
2.2 Nystrom Method

The Nystrom method is an approximation to PCA where we sample a subset of landmark features. Therefore since we utilize fewer features, we end up with a covariance matrix of lesser rank. Since computing the eigen value composition of a matrix is $O(n^3)$, reducing the size of the matrix significantly reduces computational complexity.

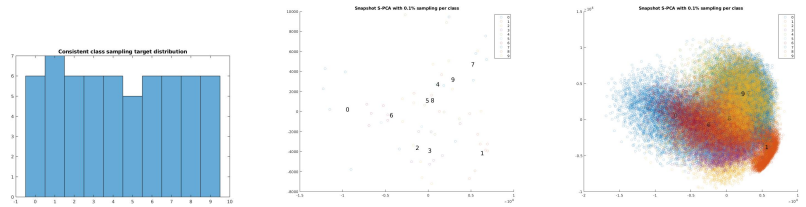
2.2.1 Implementation

The algorithm for the Nystrom method is as follows:

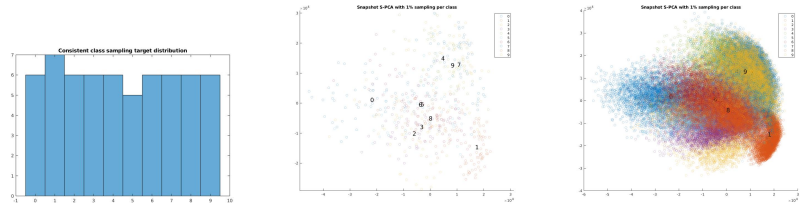
1. Select some number of features k which we want to use for the analysis through a sampling technique.
2. Based on the indices of those selected features in the matrix, reorder the columns (if the features are columns) of the matrix to incorporate them in the beginning of the matrix.
3. Compute the covariance of this reordered matrix and use the rectangular component limited to include the first selected k column features.
4. Compute the eigen value decomposition of the (kxk) part of the rectangular component and sort it's eigen vectors in decreasing order with respect to the eigen values.
5. Compute the dot product of the lower part (n-kxk) of the rectangular component with eigen vectors and eigen value matrix computed in the previous step and store it.
6. Concatenate the eigen vectors of upper part of the rectangular matrix(kxk) with the result computed in the step previously to from a (nxk) projection matrix
7. Reorder the rows of the projection matrix to represent the original ordering of columns of the input matrix.
8. Compute the dot product of the data with the projection matrix to get the final output matrix in the reduced dimensional space.



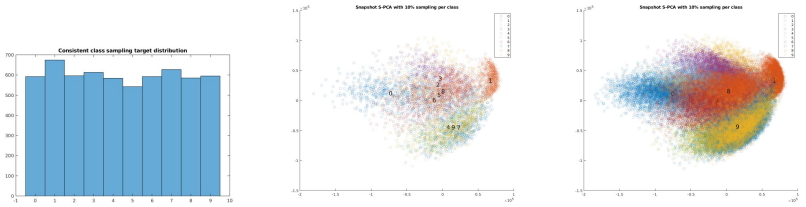
(a)



(b)

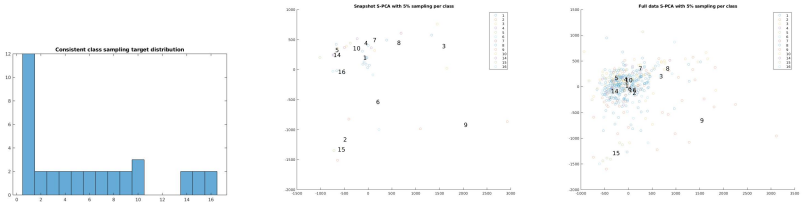


(c)

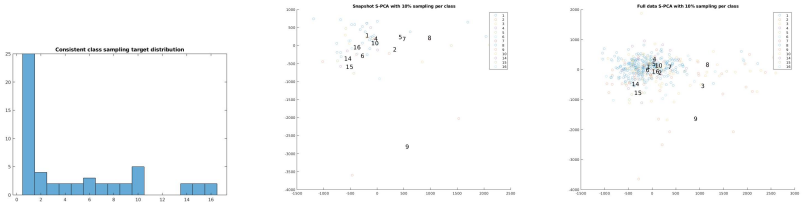


(d)

Figure 8: Snapshot PCA on MNIST dataset using *Consistent* Sampling. Every row shows the class distribution of the samples, the samples selected in the snapshot for calculating the mapping and the 2D embedding of the full dataset. The sampling rates are a) 0.07%, b) 0.1%, c) 1%, d) 10%. *(Class labels are plotted at the means and the data samples are hollow circles with alpha 0.3).*



(a)



(b)

Figure 9: Snapshot PCA on Arrhythmia dataset using Consistent Sampling. Every row shows the class distribution of the samples, the samples selected in the snapshot for calculating the mapping and the 2D embedding of the full dataset. The sampling rates are a) 5%, b) 10%,. *(Class labels are plotted at the means and the data samples are hollow circles with alpha 0.4).*

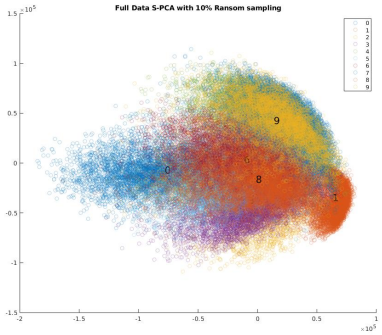


Figure 10: MNIST 2D Embedding

2.2.2 Results

In figure 11 we show a plot of the Nystrom method applied on the iris dataset by taking a subset of 3 features. We realize that performing the Nystrom method on data with such a low dimensionality is unnecessary, but we still want to show this for illustration purposes of the effect of choosing this method on a small dataset.

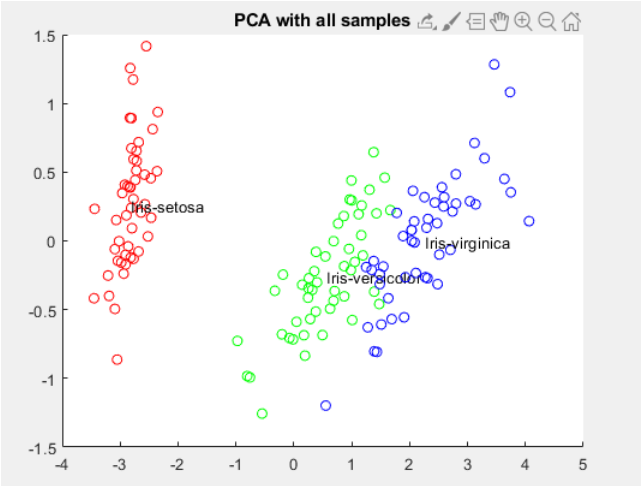


Figure 11: Nystrom method applied on Iris dataset selecting subset of 3 features

In figure 12 we show a plot of the nystrom method applied on the Arrhythmia dataset. For this we selected $l = 50$ out of the 278 features.

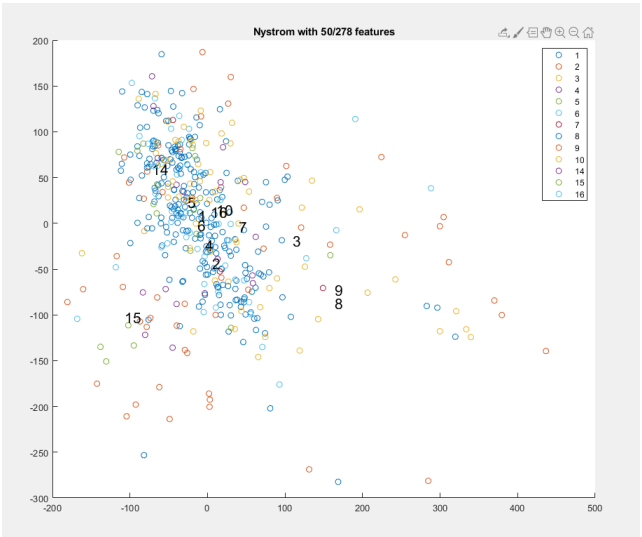


Figure 12: Nystrom method applied on Arrhythmia dataset selecting subset of 50 features

3 Task 3 Multi-Dimensional Scaling

3.1 Description

Multidimensional scaling (MDS) is used to give the user a visual indication of the dissimilarities(for e.g Squared euclidean distances) among a set of entities in a lower dimensional setting. For instance, if we were provided with a dissimilarity matrix of different cities, MDS could help the user in plotting the various cities on a two dimensional map such that, cities perceived to be different would be shown farther away and cities perceived to be similar would be positioned closer to each other.

3.2 Implementation

As for the algorithm itself, the input to the MDS algorithm is a dissimilarity matrix and the output is a set of coordinates for each observation in the reduced feature space as chosen by the user. The goal of the user when using the MDS algorithm is to minimize the stress. The stress is the overall sum of the squared differences with respect to the square root of the input dissimilarity matrix and the output dissimilarity matrix respectively. A natural consequence of reducing the dimensionality to fewer dimensions is that it leads to an increase in the overall stress value. Usually in order to visualise the data, users choose to reduce it to two or three dimensions.

3.2.1 Datasets

To implement the MDS algorithm, we chose the following datasets-:

1. Iris Dataset.
2. FIFA 19 Player Statistics Dataset.

The Iris dataset contains 3 classes of 50 instances each, where each class refers to a type of iris plant. This dataset was actually used to compare the results of PCA with that of the MDS method. We were curious to see if reducing the dimensionality using both methods produced the exact same visual representation in 2-d.

In the case of the FIFA 19 dataset, it contains the performance statistics of different players in the game. We decided to use this dataset to showcase the prototypical usage of the MDS method as a means to see the similarity/dissimilarity of players in 2-d based on their statistics. The full feature space was 7 dimensional in our case.

3.2.2 Algorithm

For both these datasets, we needed to compute the dissimilarity matrix of the observations in order to feed it further into the MDS algorithm. Once we had the dissimilarity matrix, we performed the following operations-:

1. Computed the gram matrix via double centering.
2. Performed an eigen value decomposition on the gram matrix to extract the eigen values(as a diagonal matrix) and eigen vectors.
3. Sorted the eigen values and the respective eigen vectors in descending order
4. Extracted the first 2 eigen vectors and eigen values needed to reduce the dimensionality to 2 dimensions.
5. Computed the coordinates in lower dimensional feature space by performing a matrix multiplication of the eigen vectors with the square root of the eigen value matrix.

Once we had the new low dimensional feature space, we computed the dissimilarity matrix in the output space and used this to compute the overall stress as described previously.

3.2.3 Tests for correctness

We made sure that the stress value decreased as we approached the same dimensions as the input and indeed observed that it eventually converges to zero. Apart from that we made sure that all our dimensions were as we expected them to be when computing the gram matrix and the output coordinates of our observations as well. Finally, we checked if the results of our method was identical to the built in Matlab functionality.

3.3 Results

For the FIFA 19 dataset, when we reduced the feature space to 2 dimensions. We get the following visualisation as shown in figure 13. The top two eigen values of the gram matrix are 2416.4 and 799.5 respectively. And the overall stress value for this visualisation is 637.5821. From this visualization we can see that Christiano Ronaldo and P. Dybala are similar players. We also see Sergio Busqeuts who is a defensive mid-fielder is somewhat far away from all the other attacking players.

For the Iris dataset, we see the exact same visual representation as of PCA thereby confirming the fact that PCA and MDS are equivalent methods if the squared dissimilarity matrix comes from a point set in $\mathbb{R}^n$ as shown in figure 14. The stress value for the Iris dataset is 22317 which is quite high considering all the points were initially in 4-d. The top 2 eigen values of the gram matrix are 629.5 and 36.1.

Will add references to data later.

References

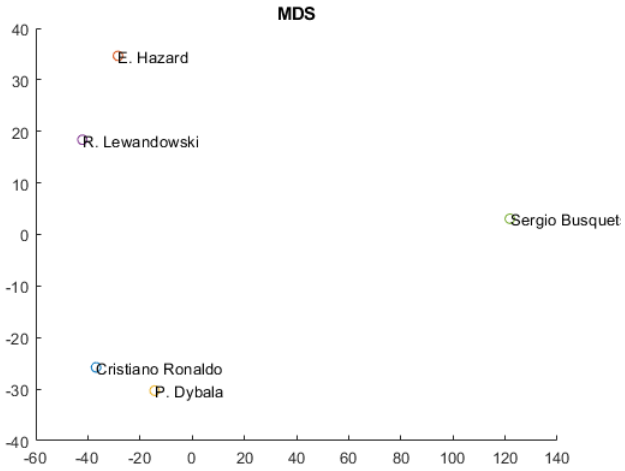


Figure 13: MDS on famous football players.

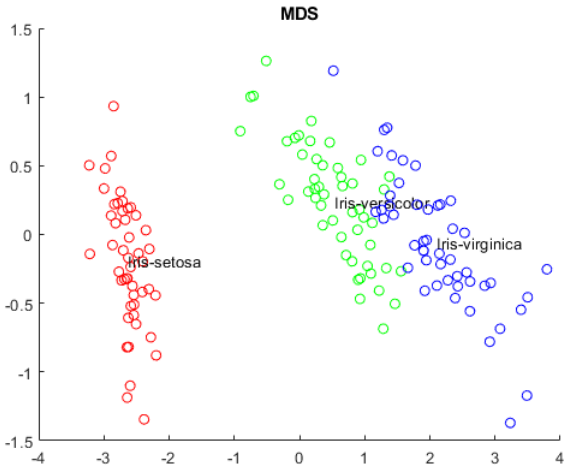


Figure 14: MDS on famous football players.