Assignment 1

Ruixue Zhang 20619404, Yican Shi 20618295, Mengzhen Gao 20610458

February 2017

1 Data Cleaning and Preprocessing

The code is in questionOne.m

1.1 Detect any problems that need to be fixed

Two main problems are detected in dataset A, missing data and outliers. For missing values, I count NaN elements for all the features of each sample and all the samples of features to decide how to fix it. For outliers, since this data is time series data from separate sensors, the correlation of each feature will not be high, therefore, I detect each feature individually. The particular method is that data of over the range of three times of standard deviation after subtracting mean values are seen as outliers.

1.2 Fix the detected problems

1.2.1 Fixing missing data

Samples with all features being NaN and features with all samples being NaN must be deleted. After that, I recount the number of missing data, and delete samples with most features being Nan and features with most samples being NaN through setting threshold value manually. Through setting different threshold value and counting the number of deleted values for each time, I try to find a reasonable value to make sure the number of deleted values won't be too large. At last, the thresholds are settled down as 5 features of each sample and 10000 samples of each feature. The remaining missing data are substituted by the mean values of each feature.

1.2.2 Fixing outliers

After detecting outliers, I directly replace them with mean values of each feature. One specific point is that when calculating mean values, outliers are not involved to be calculated with other normal values together.

1.3 Data normalization

I perform both Min-max normalization, as equation 1 [1] and z-score normalization, as equation 2 [2]. The histograms of feature 9 and 24 under different conditions are respectively shown as Figure 1. The shapes of the figures in three conditions remain same, also do the number of samples in every intervals. The only difference is presented on x axis with different domain, [-1000,1500] and [-2000,2000] corresponding to without normalization, [0,1] and [0,1] to min-max normalization, and [-4,4] and [-4,4] to z-score normalization. According to the definition of min-max normalization and z-score normalization, it's easily to analysis why domains are scaled to these range.

$$x' = \frac{x - min(x)}{max(x) - min(x)} \tag{1}$$

$$z = \frac{x - \mu}{\sigma} \tag{2}$$

where μ is the mean of the population and σ is the standard deviation of the population.

Autocorrelation of feature 9 and 24 before and after normalizations are performed, shown as Figure 2. It shows that normalizations have no effect on auto-correlation. Theoretically, normalization can be seen as linear transformation and under a linear transformation of variables, correlation will not change if the slope is positive. The slope in max-min normalization is

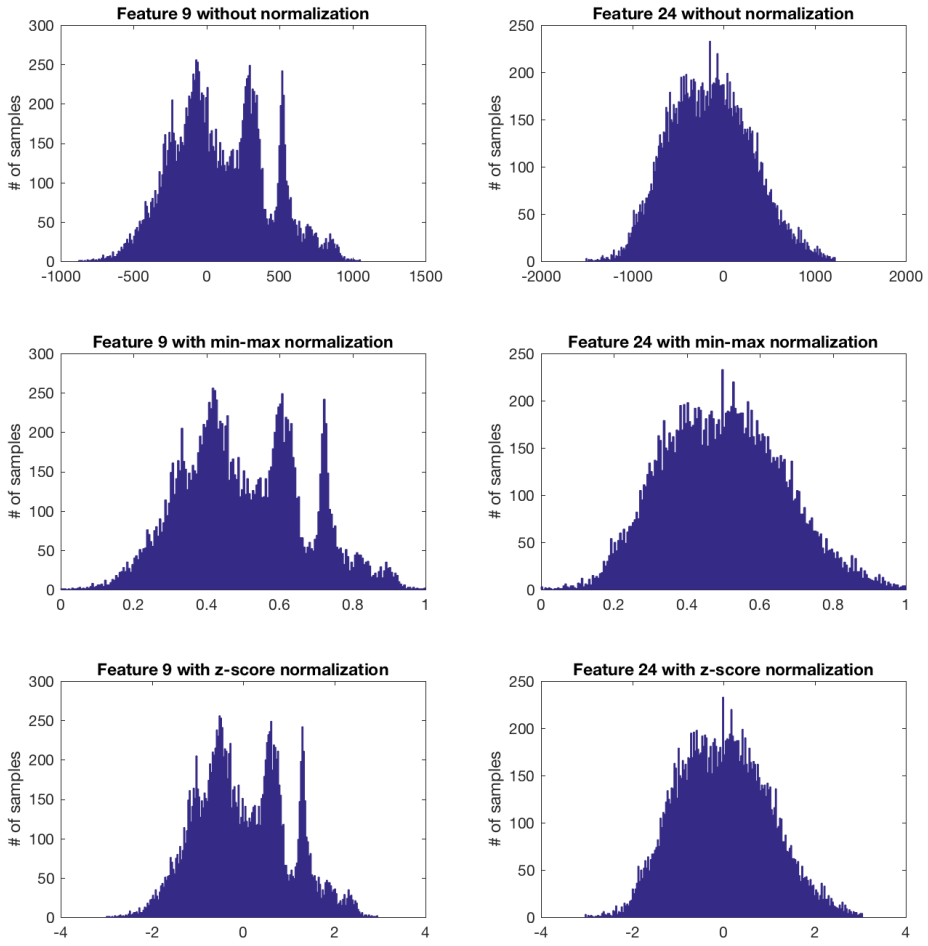

Figure 1: Histograms of feature 9 and 24 without normalization, with min-max normalizatio and with z-score normalization

$1/(max-min)$ and in z-score normalization is the standard deviation, which are both definitely positive. Hence, the observation fits the theoretical analysis.

2 Feature Extraction

The code is in questionTwo.m

2.1 Compute the eigenvectors and eigenvalues

1) Normalize the data by subtracting mean.
2) Calculate covariance matrix.
3) Use eig function to directly return eigenvectors and eigenvalues.

2.2 Two-dimension representation of the data based on 1st and 2nd principal components

The pca function in matlab can directly return the principal component coefficients, in which the first and second columns can be applied with the data to get 1st and 2nd principal components. With the two main features, a two dimensional representation of the data can visually plotted as Figure 3. Five clusters are separated in principle, presenting as five different colors. One color represents one class.

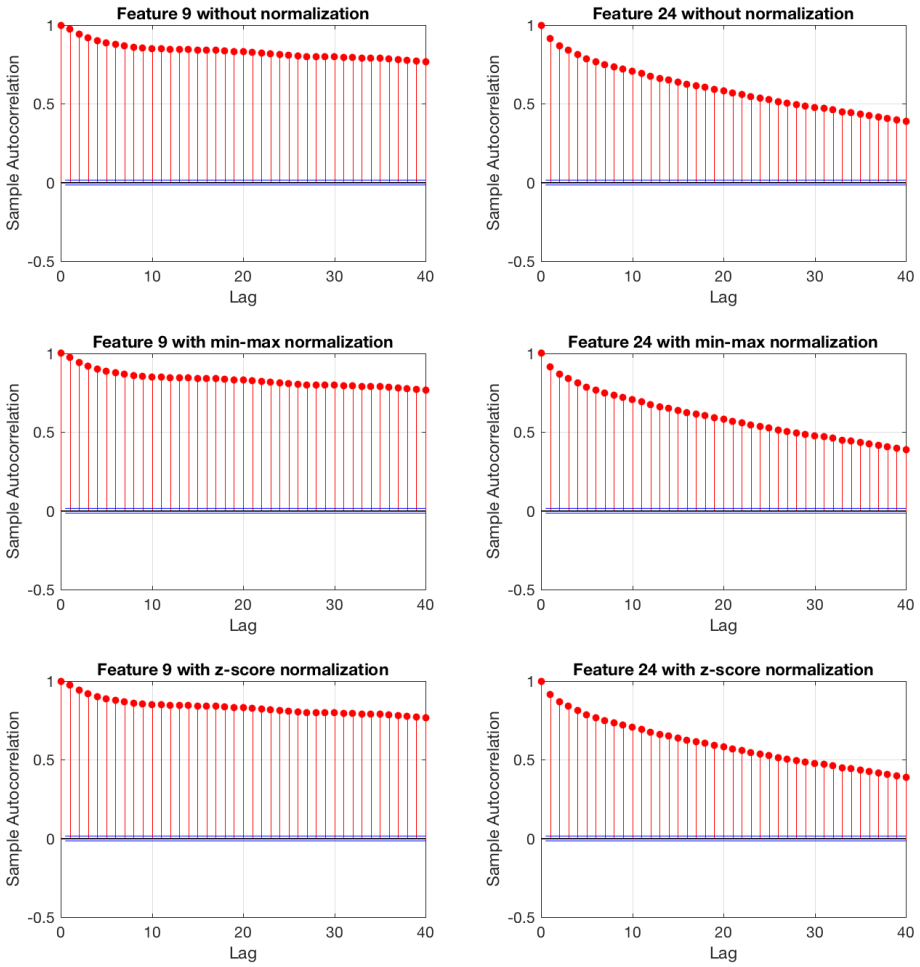

Figure 2: Autocorrelation of feature 9 and 24 without normalization, with min-max normalizatio and with z-score normalization

2.3 Two-dimension representation of the data based on 5th and 6th principal components

In this sub-question, we use similar method as last question, plotting two-dimension representation of the data based on 5th and 6th principal components as Figure 4. Since 5th and 6th principal components represent less important than 1st and 2nd components, Figure 4 shows more distributed data points and a worse performance to do the classification.

2.4 Use the Naive Bayes classifier to classify

Naive Bayes classifier is used to classify 8 sets of dimensionality reduced data (using the first 2, 4, 10, 30, 60, 200, 500, and all 784 PCA components) with same approach as last question. The classification error is directly returned by function classify. And the retained variance is obtained from principal component variances, which is another return value by function pca in matlab. The versus result is plotted in Figure 5. The expression of the retained variance is [3]

$$\sum_{i=1}^{m} \lambda_i / \sum_{i=1}^{d} \lambda_i \geq \tau < 1 \tag{3}$$

where m is the number of eigenvalues of R we choose and d is the dimension of original sample. Obviously, when m is approaching to the original dimension, the retained variance is more close to 1. In comparison, the classification error is obtained from function classify, which is

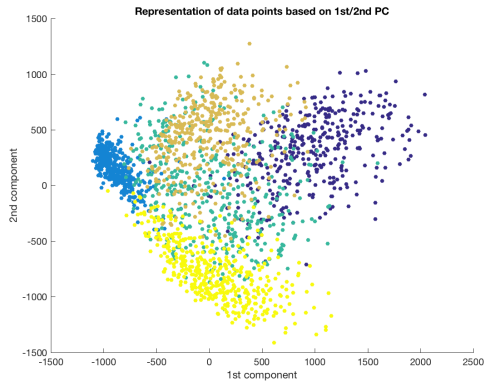

Figure 3
Two-dimension representation of the data based on 1st and 2nd principal components

Figure 4
Two-dimension representation of the data based on 5th and 6th principal components

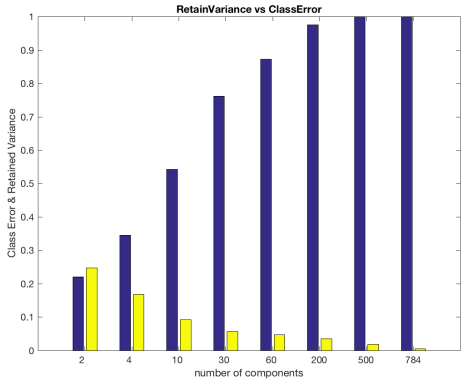

Figure 5
Classification error for 2, 4, 10, 30, 60, 200, 500, and 784 PCA components against the retained variance

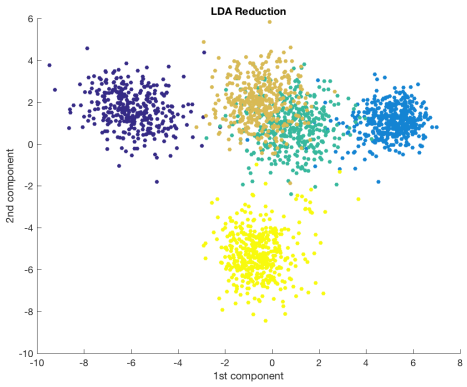

Figure 6
Two-dimension representation of the data based on 1st and 2nd LDA components

based on training test. With more features, the distribution will fit training set more precisely. Also, if no features are redundant, more features will contain more variance and give more information to classify. Hence, the classification error will be lower and lower with the number of features increasing.

2.5 Use LDA to reduce the dimensionality

In this section, we add drtoolbox to the folder and set parameter type to 'LDA' in function compute_mapping to implement LDA reduction. Then, we display data points with the first 2 LDA components in Figure 6. We can see the clusters are almost perfect seperated.

Compared with PCA, LDA is a more reasonable method to do classificaiton. According to Wikipedia[4], LDA explicitly attempts to model the difference between the classes of data. PCA on the other hand does not take into account any difference in class, and factor analysis builds the feature combinations based on differences rather than similarities. This analysis corresponds to the images.

3 Nonlinear Dimensionality Reduction

The code is in questionThree.m.

3.1 LLE for Hand Written Digits of 3

Locally linear embedding (LLE) is an approach, which addresses the problem of nonlinear dimensionality reduction by computing low dimensional neighborhood preserving embedding

of high-dimensional data. A data set of dimensionality n, which is assumed to lie on or near a smooth nonlinear manifold of dimensionality d (less than n), is mapped into a single global coordinate system of lower dimensionality, d. The global nonlinear structure is recovered by locally linear fits. Shown by Figur 7, we can see some patterns such that the two features extracted by LLE reflecting the slope or pose (namely left and right) of digit 3. The digit 3s in the central part is almost vertical. The digit 3(s) in the left part of the coordinates is more likely to lean to the left and vise versa.

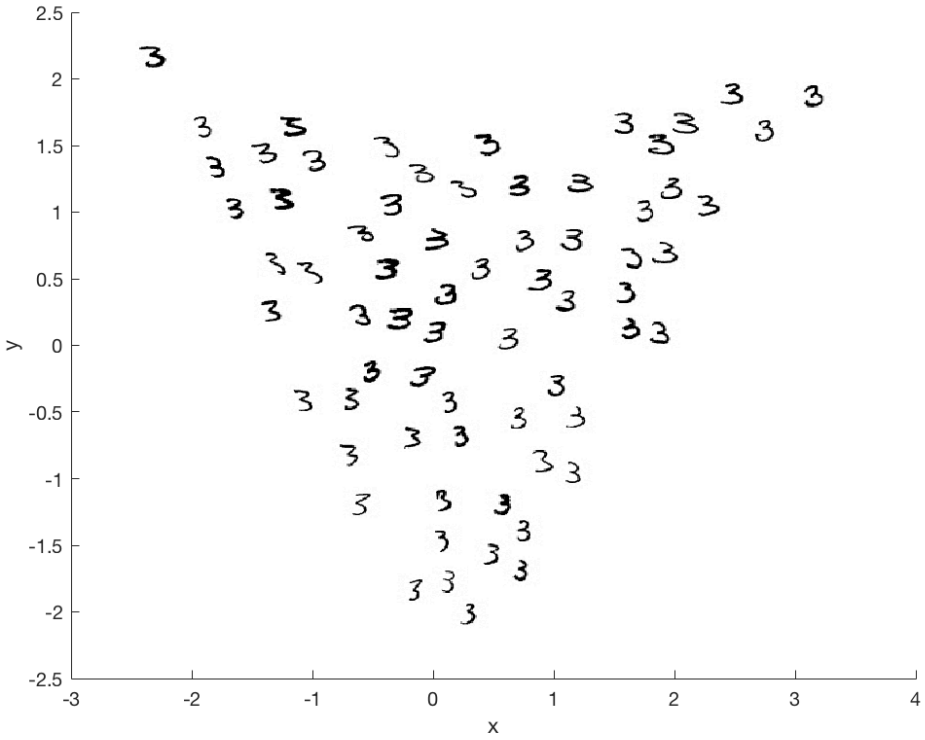

Figure 7: The first vs. second components of LLE

3.2 ISOMAP for Hand Written Digits of 3

Isomap is a nonlinear generalization of classical MDS (Multidimensional Scaling). The main idea is to perform MDS, not in the input space, but in the geodesic space of the nonlinear data manifold. In this part, shown by Figure 8, we can also conclude that the two features selected by Isomap represent the slope or pose (namely left and right) of digit 3. From upper right to lower left of the displayed coordinate area, the digit 3s tended inclination is from left to right.

It does not do better than LLE. LLE can present the original manifold structure which Isomap lacking of. The patterns being found are locally based.

3.3 Compare the Performance of PCA, LDA, LLE and ISOMAP

Firstly, in order to do the classification, we applied LLE and ISOMAP on all digits not only on digit of 3.

Secondly, Naive Bayes classifier is applied to classify the dataset based on the projected 4-dimension representations of the LLE and ISOMAP of dataset B. In this step, we use 70% of the data as training set and 30% as testing set, running the classifier function for 50 times to get an average error rate. Here fifty times is theoretically enough for random abnormal results. The average accuracies of LLE and ISOMAP are 95.82% and 95.01% respectively.

In this part, we also added code and got the average accuracies of PCA and LDA as 94.27% and 99.81% respectively, shown by Table 1. The result shows that LDA has a much better accuracy than other algorithms because it aims to find a linear combination of features that characterizes or separates two or more classes of the data and explicitly attempts to model

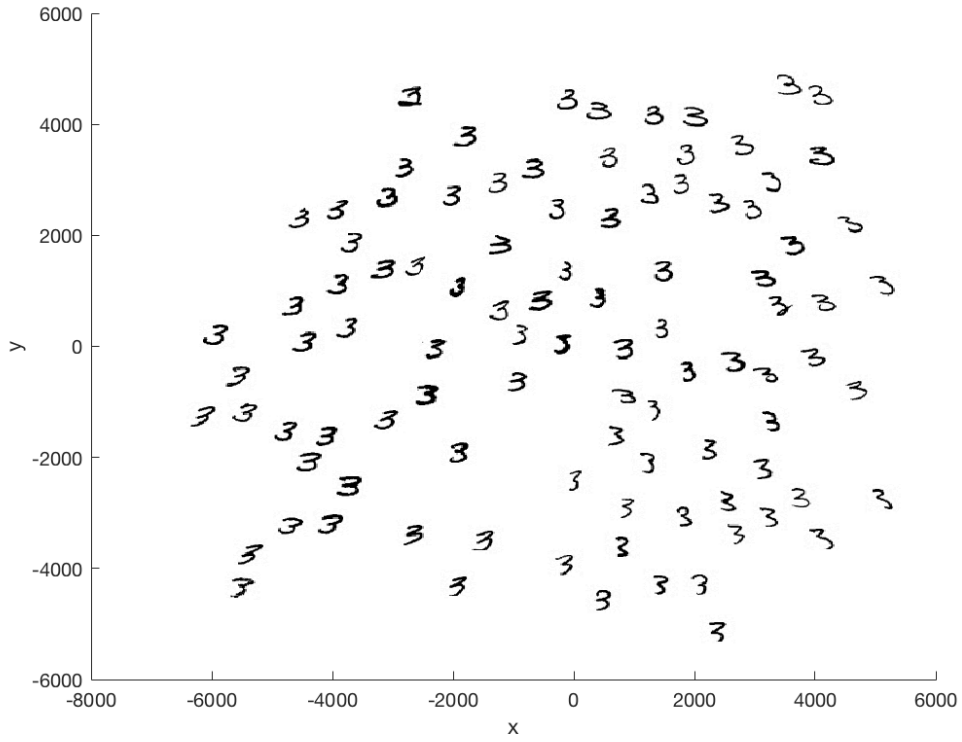

Figure 8: The first vs. second components of ISOMAP

the difference between the classes of data. Thats why it has a relatively positive influence on classification.

Finally, we see that LLE does a slightly better job than ISOMAP in terms of classification. As nonlinear dimensionality reduction methods, LLE and ISOMAP use some similar steps such as KNN to find neighbors to reduce high dimensional manifold to lower dimension. LLE is a global method while Isomap is a local method. So here we change the number of neighbors to see the effects of both methods. When the neighbor number reduce to 4, accuracy of LLE is 92.47% and ISOMAP is 96.06%. While if the neighbor number increase, accuracy of LLE improves and ISOMAP decreases a little bit.

Table 1: Classification accuracies of PAC, LDA, LLE and ISOMAP

Method	Average-accuracy(%)
PCA	94.27
LDA	99.81
LLE	95.82
ISOMAP	95.01

4 Feature Selection

The code is in questionFour.m and questionFour_new.m

4.1 Sequential Forward Selection with Squared Euclidean Distances

We Use Sequential Forward Selection (SFS) strategy and the sum of squared Euclidean distances as an objective function. Each time, we select the feature which can maximize the squared Euclidean distance between different classes. We select total 8 features. The selected features are [21,13,10,12,8,5,7,4].

4.2 Sequential Forward Selection with Bayes Classifier

We use the Naive Bayes classifier as the objective function, realize a wrapper based feature selection with SFS search strategy (to select 8 features).The selected features are not stable but have considerable overlap, depending on which part of data being chosen as training and testing set. Some sample results are: [7,10,8,18,2,5,14,6], [7,10,17,2,19,4,6,3], [7,10,8,17,14,6,16,2], [7,10,8,17,2,5,6,16], [7,10,8,6,18,2,5,16]

4.3 Sequential Backward Selection with Bayes Classifier

We use the Naive Bayes classifier as the objective function, realize a wrapper based feature selection with Sequential Backward Selection (SBS) search strategy (to select 8 features).The selected features are not stable but have considerable overlap, depending on which part of data being chosen as training and testing set. Some sample results are: [1,2,6,7,8,10,12,17], [1,2,7,8,10,12,14,17], [1,2,5,7,8,10,14,17], [1,2,7,8,10,12,17,21]

4.4 Average Accuracy and Run Time

Table 2: Accuracy and run-time for each feature selection method with munual implemention.

Algorithm	Average Accuracy (%)	Feature Selection Runtime(s)	Classification Runtime(s)
Filter SFS	80.14	3.7391	0.0143
Wrapper SFS	83.76	2.6393	0.0116
Wrapper SBS	84.05	3.2568	0.0069
All 21 Features	78.24	-	0.0088

Table 3: Accuracy and run-time for each feature selection method using fucntion sequentialfs.

Algorithm	Average Accuracy (%)	Feature Selection Runtime(s)	Classification Runtime(s)
Filter SFS	80.29	36.6397	0.0931
Wrapper SFS	83.76	1.4972	0.0274
Wrapper SBS	83.57	3.1823	0.0334
All 21 Features	78.29	-	0.0429

We use two approaches to realize the three feature selection methods, implementing manually and using function sequentialfs.

4.4.1 Average accuracy

We use Naive Bayes classifier with K-fold and classperf to compute average accuracy in Bayes_fun.m. Classperf provides an interface to keep track of the performance during the validation of classifiers. Classperf creates and, optionally, updates a classifier performance object, CP, which accumulates the results of the classifier. Both of the two approaches don't show high accuracies. We guess it is because this accuracy computation method is precise and more strict than just compute $error = sum(preidicted(Xtrain, Ytrain, Xtest) \neq Ytest)$;

As the table shows, all three algorithms Filter SFS, Wrapper SFS and Wrapper SBS have better average accuracy than using all 21 features. The reason may be that 21 features are too many to present the data, resulting in over-fitting. Also, wrappers approaches always have better accuracies than filters as the latter is tied to the classifier and here is no significant difference between Wrapper SFS and Wrapper SBS.

4.4.2 Running time

Table 2 presents that Filter SFS is a bit slower than wrapper approaches and Wrapper SFS is the fastest one. However it is much slower shown in the table 3.

As the lecture slide says, Filter Approach has low complexity and therefore good for large number of features. We guess this weired running time is because the object function we write to calculate the sum of squared Euclidean distances is not efficient, especially not fitting

function sequentialfs. Also, the number of features and samples may be not large enough to show the advantage of Filter SFS.

Also, Wrapper SFS is faster than Wrapper SBS. We think the reason is that in Wrapper SFS, the iteration number is 8, compared with in SBS the iteration number being 13. Therefore, the Wrapper SFS is the most efficient one. To wrap up, Wrapper SFS is the best choice for this question.

References

[1] Wikipedia: Feature scaling,
`https://en.wikipedia.org/wiki/Feature_scaling`

[2] Wikipedia: Standard score,
`https://en.wikipedia.org/wiki/Standard_score`

[3] lect3:slide22.

[4] Wikipedia: Linear discriminant analysis,
`https://en.wikipedia.org/wiki/Linear_discriminant_analysis`