STAT40740 Assignment Report

Xuefei Jiang (Student ID: 22200056)

April 2, 2023

How to reproduce results?

1. Extract the zip file.
2. Enter the top directory and set it as working directory.
3. Run the R script.

How to read report and R script?

The necessary R variables are highlighted in blue so that the grader may easily find them in the R script. There are two kinds of variables in the R script:

- global variables: the variables shared by all questions.
- local variables: the variables only used in the corresponding and related questions. The question number is prefixed on local variables, e.g., q2.spectra.

Actually, there is only one global variable in the R script: global_data, and all other variables are local.

1 Question 01: Initialize workspace [0 marks]

This section covers the initialization of the workspace and includes the following operations:

- Loading the milk spectra data set into the workspace. (global_data)
- Setting the random seed to my student number 22200056 to ensure reproducibility.
- Randomly deleting one observation from the dataset, which reduces the number of observations from 431 to 430. This change will be reserved in the subsequent questions. (global_data has been changed)

2 Question 02: Data preprocessing and visualization [10 marks]

2.1 Handle NA values in β Lactoglobulin B

Rows with NA β Lactoglobulin B values are removed from the dataset, which reduces the number of observations from 430 to 305. This change will be reserved in the subsequent questions. (global_data has been changed)

2.2 Data visualization

2.2.1 spectra

The whole dataframe has 582 columns. We only plot the last 531 columns because the first 51 columns are not spectra data. The line graph is created to show the absorbances of the wavelengths in each observation. Each line corresponds to a specific observation. The plot indicates that, to some extent, the absorbance distributions of the observations are similar. However, at some specific wavelengths, e.g., around 1050nm, 1500nm, 1600nm and 3300nm, the observations have different absorbances.

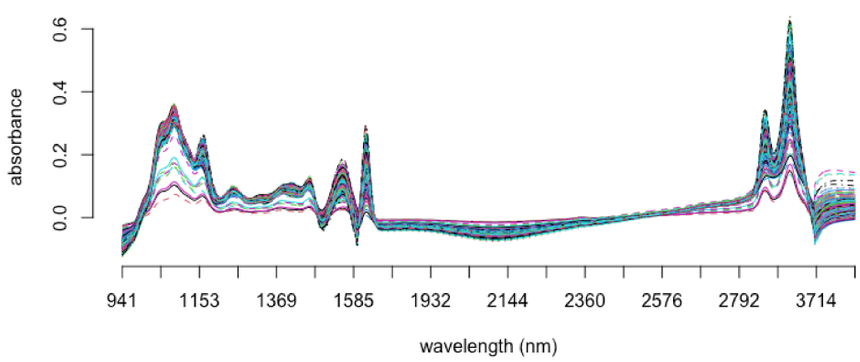

Figure 1: line graph for spectra

2.2.2 β Lactoglobulin B

A histogram is created to show the distributions of β Lactoglobulin B among all observations. The plot indicates that 1) the distribution is a little skewed 2) outliers can be found 3) there are many zero values.

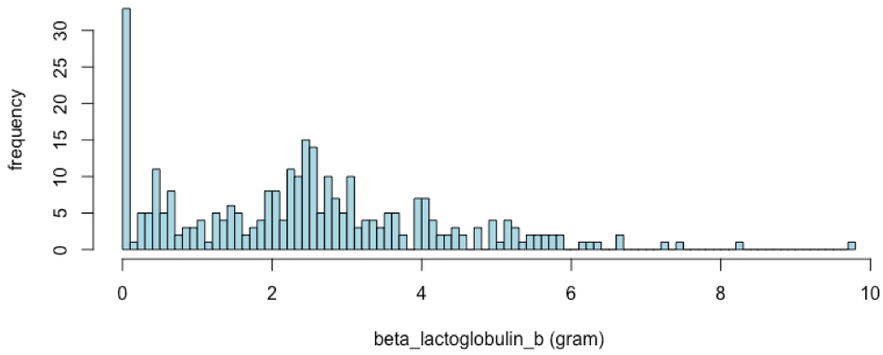

Figure 2: historgram for β Lactoglobulin B

We are interested in how many zeros are in β Lactoglobulin B and a pie chart is created to illustrate this. We can tell from Figure 3 that 10.5% values of β Lactoglobulin B are zero.

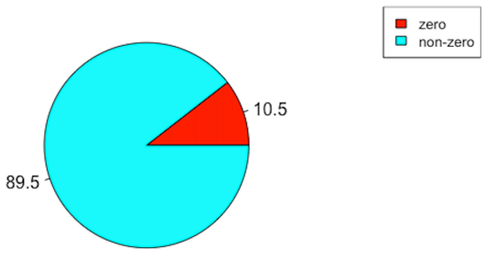

Figure 3: piechart for zero, non-zero values in β Lactoglobulin B

2.3 Handle outliers for β Lactoglobulin B

Observations with β Lactoglobulin B values outside of 3 standard deviations from the mean are removed from the data set, which reduces the number of observations from 305 to 303. This change will be reserved in the subsequent questions. (global_data has been changed)

3 Question 03: Clustering [25 marks]

3.1 Hierarchical clustering

Hierarchical clustering is conducted on spectra data. Due to the continuous nature of the spectra values, Euclidean distance is employed. As the spectra data are measured on the same scale, no standardization is done before clustering. Single, complete, average linkages are tried and the dendrograms are illustrated in Figure 4, Figure 5 and Figure 6, respectively.

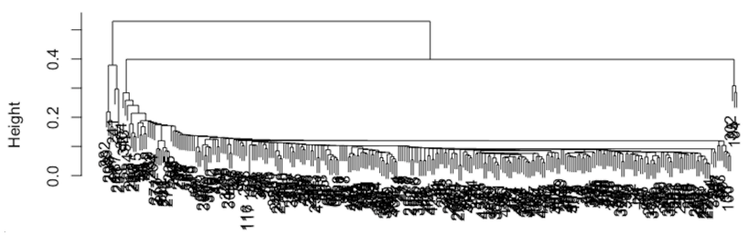

Figure 4: hierarchical clustering dendrogram: single linkage

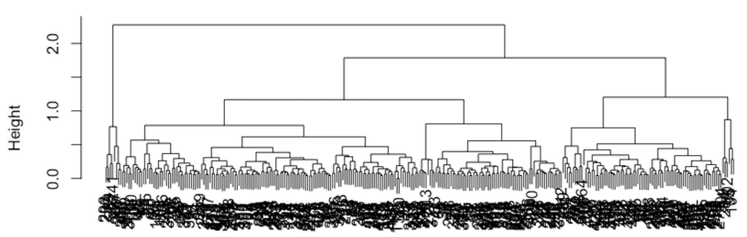

Figure 5: hierarchical clustering dendrogram: complete linkage

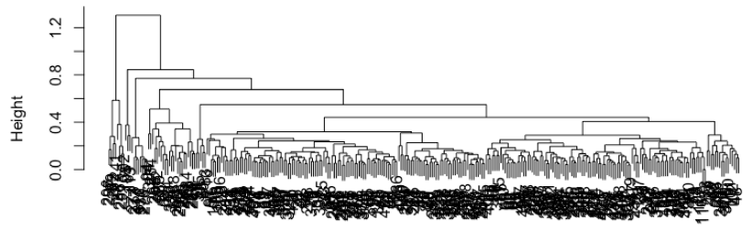

Figure 6: hierarchical clustering dendrogram: average linkage

We can tell from the results that:

- single linkage is suffered from the chaining issue
- average linkage doesn't generate clear clusters
- complete linkage is the best of the three

As a consequence, complete linkage is selected for further processing.

Then *cutree* is used to partition the data into disjoint clusters. By looking at Figure 5, the number of groups can be set to 3 or 5 and the results are illustrated in Figure 7.

```
> print(table(q3.hier_complete_3))
q3.hier_complete_3
  1   2   3
212  83   8
```

```
> print(table(q3.hier_complete_5))
q3.hier_complete_5
  1   2   3   4   5
143  69  77   8   6
```

Figure 7: results of hierarchical clustering with complete linkage

We can tell from the results that:

1. If we set the number of groups to 3, we get one large cluster, one medium cluster and one small cluster.
2. If we set the number of groups to 5, we get one large cluster, two medium clusters and two small clusters.

3.2 k-means clustering

k-means clustering with different number of clusters is conducted. We plot the within cluster sum of squares (SS) against the number of clusters k. The formula to calculate SS is illustrated in Equation 1.

$$SS = \sum_{k=1}^{K} \sum_{i \in S_k} \|\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k\|_2^2 \tag{1}$$

When $k = 1$:

$$SS = \sum_{i=1}^{n} \|\boldsymbol{x}_i - \bar{\boldsymbol{x}}\|_2^2 \tag{2}$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{p} (x_{ij} - \bar{x}_j)^2 \tag{3}$$

$$= \sum_{j=1}^{p} \sum_{i=1}^{n} (x_{ij} - \bar{x}_j)^2 \tag{4}$$

$$= \sum_{j=1}^{p} (n-1) \left(\frac{1}{n-1} \sum_{i=1}^{n} (x_{ij} - \bar{x}_j)^2 \right) \tag{5}$$

$$= (n-1) \sum_{j=1}^{p} \left(\frac{1}{n-1} \sum_{i=1}^{n} (x_{ij} - \bar{x}_j)^2 \right) \tag{6}$$

where n is the number of observations, p is the number of variables of $\boldsymbol{x}$. $\frac{1}{n-1} \sum_{i=1}^{n} (x_{ij} - \bar{x}_j)^2$ is the sample variance of jth variable, which can be calculated conveniently in R by using *apply* function. Equation 6 corresponds to the code for calculating SS when $k = 1$ in the R script.

When $k > 1$: SS can be easily calculated by the results of *kmeans* command.

The SS plot is illustrated in Figure 8.

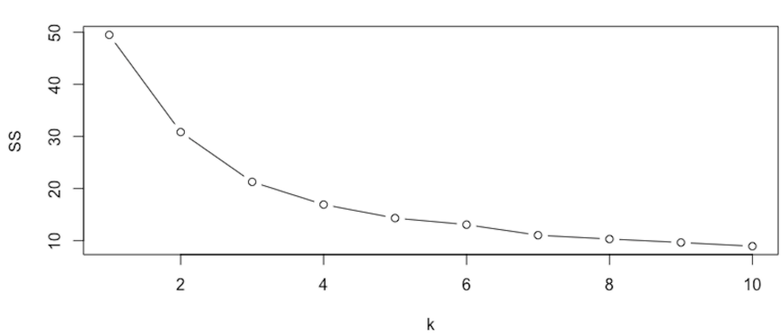

Figure 8: k-means clustering: SS for $k = 1$ to 10

The plot shows that $k = 3$ is the elbow of the plot and 3 is thus selected as the number of clusters. However, to compare the results with hierarchical clustering, we also show the result of $k = 5$.

```
> print(table(q3.kmeans_3$cluster))

  1   2   3
152 143   8
```

```
> print(table(q3.kmeans_5$cluster))

  1   2   3   4   5
 80  35   8  78 102
```

Figure 9: results of kmeans clustering

We can tell from the results that:

1. If we set the number of groups to 3, we get two large clusters and one small cluster.
2. If we set the number of groups to 5, we get one large cluster, two medium clusters and two small clusters.

3.3 Compare hierarchical clustering with k-means clustering

Cross-tabulation and adjusted rand index are employed to measure the similarity between two clustering solutions.

When $k = 3$:

```
> print(table(q3.hier_complete_3, q3.kmeans_3$cluster))

q3.hier_complete_3   1   2   3
                 1 152  60   0
                 2   0  83   0
                 3   0   0   8
> print(classAgreement(table(q3.hier_complete_3, q3.kmeans_3$cluster))$crand)
[1] 0.3878132
```

Figure 10: hierarchical clustering v.s. k-means clustering with $k = 3$

We can tell from Figure 10 the two clustering solutions agree on cluster 3 but differ on clusters 1 and 2.

When $k = 5$:

```
> print(table(q3.hier_complete_5, q3.kmeans_5$cluster))

q3.hier_complete_5  1  2  3  4  5
                 1 18 29  0  4 92
                 2 62  0  0  0  7
                 3  0  3  0 71  3
                 4  0  0  8  0  0
                 5  0  3  0  3  0
> print(classAgreement(table(q3.hier_complete_5, q3.kmeans_5$cluster))$crand)
[1] 0.5329933
```

Figure 11: hierarchical clustering v.s. k-means clustering with $k = 5$

We can tell from Figure 11 the two clustering solutions agree on cluster 3 but differ on other clusters.

In conclusion, when $k = 5$, a higher adjusted rand index is achieved, which means more agreement between hierarchical clustering and k-means clustering. The cluster of 8 observations is always, at least in the experiments, well separated from other observations.

3.4 Compare hierarchical clustering solution ($k = 3$) with built-in categories

In this section, clustering solutions are compared with the built-in categories. The solution of hierarchical clustering with $k = 3$ is selected to do this as an example. However, the method can be extended to any clustering solution. Due to space constraint, I only select the traits that can readily classify observations into a reasonable number of groups, while I believe further research should be conducted to determine the relationship between clustering solutions and the original data.

3.4.1 Breed

The clustering solution is compared with Breed.

```
> print(table(q3.hier_complete_3, global_data$Breed))

q3.hier_complete_3 FRX- Hol Fri HOX-  JE JEX-  MO  NR
                 1   11     148    1  31   13   1   7
                 2    5      38    2  26   10   0   2
                 3    0       6    0   1    1   0   0
> print(classAgreement(table(q3.hier_complete_3, global_data$Breed))$crand)
[1] 0.09325714
```

It seems that the observations in cluster 3 are more likely to be picked up from Fri.

3.4.2 Parity

The clustering solution is compared with Parity.

```
> print(table(q3.hier_complete_3, global_data$Parity))

q3.hier_complete_3  1  2  3  4  5  6  7  8  9 11
                 1 79 47 28 26 17  6  4  3  1  1
                 2 22 25 16  8  7  2  1  1  1  0
                 3  0  4  1  1  0  2  0  0  0  0
> print(classAgreement(table(q3.hier_complete_3, global_data$Parity))$crand)
[1] 0.01584808
```

There is nothing valuable to be found here.

3.4.3 Milking_Time

The clustering solution is compared with Milking_Time.

```
> print(table(q3.hier_complete_3, global_data$Milking_Time))

q3.hier_complete_3   1   2
                 1 154  58
                 2  30  53
                 3   4   4
> print(classAgreement(table(q3.hier_complete_3, global_data$Milking_Time))$crand)
[1] 0.1454515
```

It seems that observations in cluster 1 are more likely to be picked up from Parity 1 and the adjusted rand index is the highest among the four.

3.4.4 DaysInMilk

The clustering solution is compared with DaysInMilk. The values in DaysInMilk are classified into 3 categories:

- category 1: values lower than 1/3 quantile;
- category 2: values higher than or equal to 1/3 quantile and lower than 2/3 quantile
- category 3: values higher than or equal to 2/3 quantile;

```
> print(table(q3.hier_complete_3, q3.days_in_milk))
                   q3.days_in_milk
q3.hier_complete_3  1  2  3
                 1 61 72 79
                 2 32 30 21
                 3  7  0  1
> print(classAgreement(table(q3.hier_complete_3, q3.days_in_milk))$crand)
[1] 0.007239998
```

It seems that the observations in cluster 3 are more likely to be picked up from category 1, the category with lower values of DaysInMilk.

4 Question 04: PCA [10 marks]

PCA is applied to the spectra data. In the spectra case, standardization is not conducted because all spectra values are measured on the same scale and in the same units (nm). PCA is carried out with *prcomp* command, and the following values are manually calculated:

1. variance of each PC (q4.var)
2. proportion of variance explained (q4.prop)
3. cumulative proportion of variance explained (q4.cum_prop)

In order to determine how many PCs are sufficient, the cumulative proportions of variance explained for the first 10 PCs are plotted in Figure 12. We can tell from the results that:

- The first PCs account for 63.45% of the variation in the data.
- The first 2 PCs account for 82.36% of the variation in the data.
- The first 3 PCs account for 95.08% of the variation in the data.
- The first 4 PCs account for 99.20% of the variation in the data.

I believe that, for most cases, 3 PCs are sufficient to capture the variation of the data. However, if you want to capture more variation, you may choose 4.

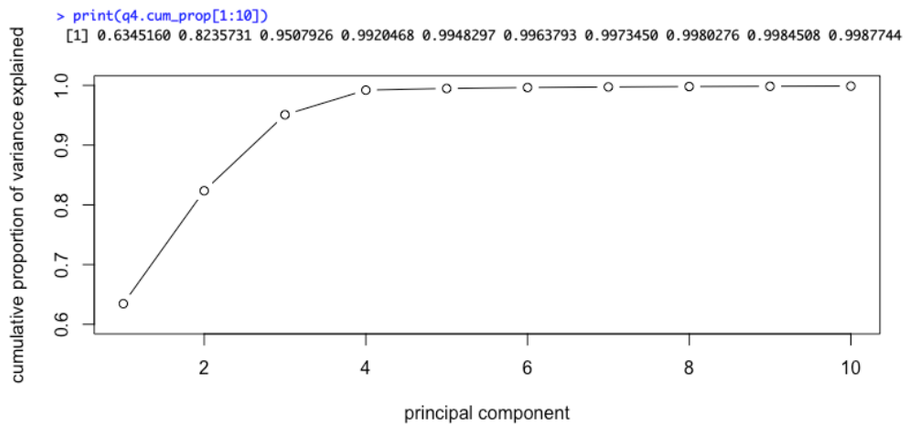

Figure 12: cumulative proportion of variance explained

5 Question 05: PCA [15 marks]

In PCA, given an observation $\boldsymbol{x}$, we can calculate its score on ith PC by

$$y_i = \boldsymbol{a}_i^T \boldsymbol{x}. \tag{7}$$

Given n observations, the vectorized version can be used to calculate scores on the first q PCs efficiently.

$$\underbrace{\begin{bmatrix} - & \boldsymbol{y}^{(1)} & - \\ & \vdots & \\ - & \boldsymbol{y}^{(n)} & - \end{bmatrix}}_{n \times q} = \underbrace{\begin{bmatrix} - & \boldsymbol{x}^{(1)} & - \\ & \vdots & \\ - & \boldsymbol{x}^{(n)} & - \end{bmatrix}}_{n \times p} \underbrace{\begin{bmatrix} | & & | \\ \boldsymbol{a}^{(1)} & \dots & \boldsymbol{a}^{(q)} \\ | & & | \end{bmatrix}}_{p \times q} \tag{8}$$

First, scores on the first 3 PCs for the spectra data are calculated in the following steps:

1. Center the spectra data because we mainly care about variations of data. (q5.centered_spectra)
2. Loadings for the first 3 PCs (q5.loadings_3) are extracted from the results of *prcomp* command (q5.pca).
3. Multiply q5.centered_spectra by q5.loadings_3 to get the scores according to Formula 8.

Then, we want to create scatter plots to illustrate the scores on one PC against the scores on another PC. *pairs* command are employed to plot all pairs of the first 3 PCs and the results are illustrated in Figure 13. We can tell from PC1-PC2 plot that there are 8 well separeted observations in the bottom left corner, which might corresponds to the well separated cluster generated by both hierarchical clustering and k-means clustering.

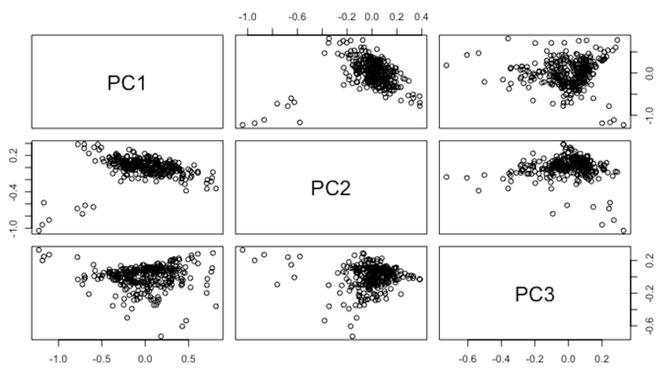

Figure 13: scores on the first 3 PCs

Then we differentiate the observations according to the clusters generated by the hierarchical clustering with $k = 3$, which is illustrated in Figure 14. The conjecture has been verified that the 8 well separated observations in PC1-PC2 plot belong to the same cluster from the hierarchical clustering.

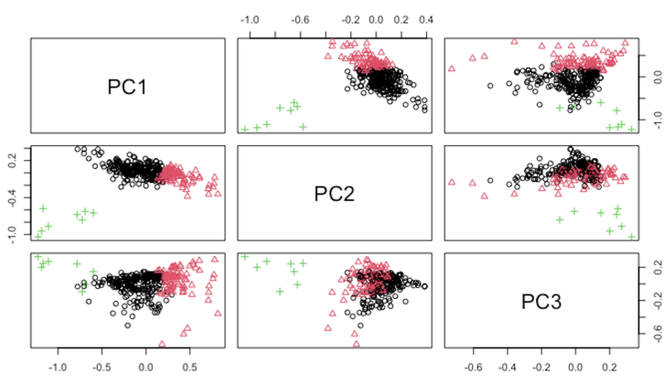

Figure 14: differentiate observations according to hierarchical clustering

To better explain, we denote the green cluster as cluster 1, the black cluster as cluster 2 and the red cluster as cluster 3. We can tell from the results that:

- The observations in cluster 1 have much lower scores on PC1 and PC2 than other observations, which can explain why they are well separated in both hierarchical clustering and k-means clustering. And, though not as obvious as PC1 and PC2, the observations in cluster 1 tend to have higher scores on PC3.
- The differences between cluster 2 and cluster 3 can also be well explained here. Observations in cluster 2 tend to have lower scores on PC1 and higher scores on PC2 than observations in cluster 3, which may be the reason why they are classified into different clusters in the hierarchical clustering.

Finally, we differentiate the observations according to the built-in categories. According to the space constraint, I only put the results of Breed and DaysInMilk here, which are illustrated in Figure 15. However, you can find results of other built-in categories in the R script. What's more, as we did in Question 3, values in DaysInMilk are classified into 3 categories.

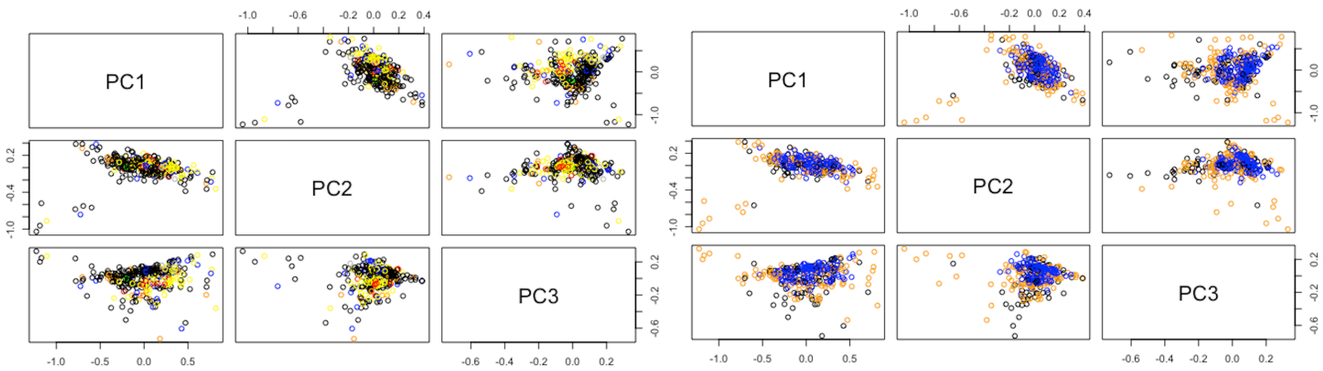

Figure 15: differentiate observations according to Breed (LHS) and DaysInMilk (RHS)

However, when applying the same strategy to the built-in categories, there is nothing as valuable as we found in explaining the results of the hierarchical clustering. More research needs to be done on this topic.

6 Question 06: Reivew on PCR [30 marks]

Principal Components Regression (PCR) originates from the vanilla linear regression (LR) model

$$y = \boldsymbol{\beta}^T \boldsymbol{x} + \epsilon \tag{9}$$

which describes the relationship between r.v. $y \in \mathbb{R}$ and r.v. $\boldsymbol{x} \in \mathbb{R}^p$, and the least squares method is used to estimate the parameters of the model. There are some extensions to the vanilla LR model which may improve the prediction accuracy and model interpretability. The common methods are subset selection, srhinkage methods and dimension reduction methods, and PCR is a widely used dimension reduction technique.

1. **explain the method's purpose**
   In the fields of statistics and machine learning, we typically favor lower-dimensional models over higher-dimensional ones without reducing performance (curse of dimensionality). The rationale for this is that since variables may be correlated, a model can be established with fewer variables. According to this logic, PCR's goal is to transform higher-dimensional space into lower-dimensional space, and then apply the LR model in the latter.

2. **provide a general description of how the method works**
   In PCR, Principal Component Analysis (PCA) is conducted on $\boldsymbol{x}$. It is assumed that only a small number of PCs are sufficient to model the variation of $\boldsymbol{x}$ and the relationship between y and $\boldsymbol{x}$. The loadings that correspond to those PCs are selected as the axes of the new lower-dimensional space. $\boldsymbol{x} \in \mathbb{R}^p$ is then transformed into $\boldsymbol{z} \in \mathbb{R}^q$ ($q << p$) using Equation 8. Now the relationship between y and $\boldsymbol{x}$ can be modeled by another linear regression model

   $$y = \boldsymbol{\beta'}^T \boldsymbol{z} + \epsilon'. \tag{10}$$

   Finally, we can employ any techniques that are applicable to the vanilla LR model in the transformed LR model, such as estimating the parameters using the least squares method.

3. **detail any choices that need to be made when using the method**
   First, we should decide whether the data should be standarized because the PCA is sensitive to data scaling. The general rule is that, if you believe that the variances variables are of the same importance, the standarization should be avoided to capture the differences between variables. Otherwise, standardization should be carried out.

   Second, a suitable number of PCs should also be decided to achieve the best of performance in PCR. We may first plot the cumulative proportion of variance explained to see how many PCs are sufficient to capture the most variation of the original data. Then the cross-validation is carried out formally to determine the suitable number.

   Third, we must design the evaluation process as well, e.g., which metric to use, how to partition the dataset for training and testing, which baseline should our model is compared with, etc.

4. **outline the advantages and disadvantages of the method**
   The fact that PCR transforms higher-dimensional data into lower-dimensional data leads to the following benefits:

   - less parameters of the new model and less data needed for training;
   - hopefully, the strategy can mitigate overfitting;

   PCR also assumes that the transformed lower-dimensional data can automatically capture the relationship between the response and the predictors. However, this is not always the case. The new lower-dimensional predictors do capture the most variations of the higher-dimensional ones, but this doesn't mean they can also capture the relationship between the response and the predictors very well. We may need to find a way to take such relationship into account in PCR ($\Rightarrow$ Partial Least Square).

7 Question 07: PCR [25 marks]

PCR is applied to model the relationship between β Lactoglobulin B and the spectra data. Using a ratio of 2:1, the dataset is divided into a train set and a test set. Some decisions are made when fitting the PCR model:

- Standardization is not applied to the spectra data because all spectra values are measured on the same scale and in the same units (nm).
- However, centering is applied to the spectra data because I don't want the predictors to shift too far from the origin.
- Root Mean Squared Error (RMSE) is selected as the performance metric because we are dealing with a regression task.

PCR is carried out with *pcr* command in *pls* package. I tried 100 principal components and repeated the experiment three times using different random seeds. The results are illustrated in Figure 16, 17 and 18. The LHS of each figure is the RMSE using different number of PCs in the training stage, and the RHS is the same thing in the testing stage.

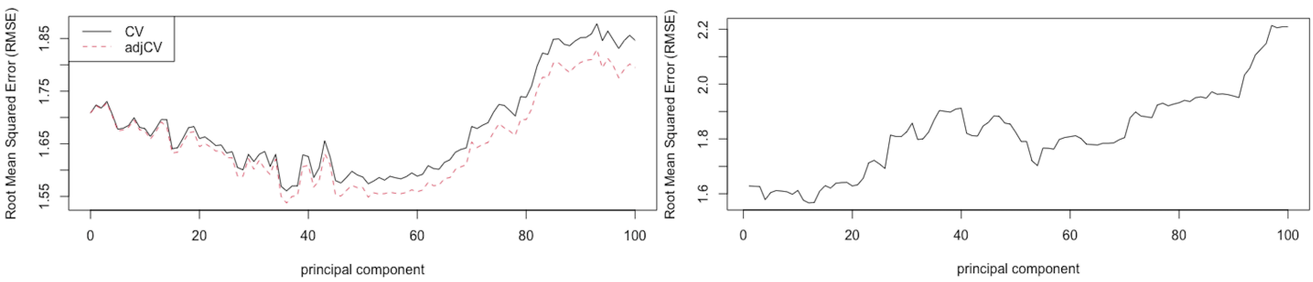

Figure 16: PCR experiment 1: RMSE against number of PCs, LHS is for training and RHS is for testing

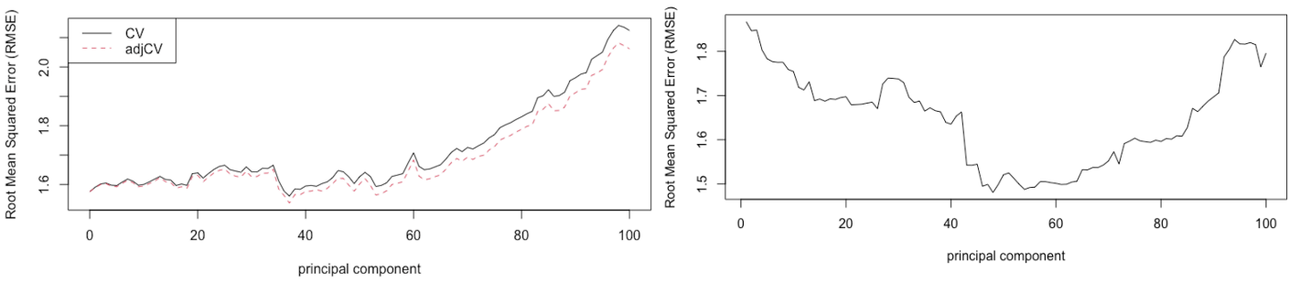

Figure 17: PCR experiment 2: RMSE against number of PCs, LHS is for training and RHS is for testing

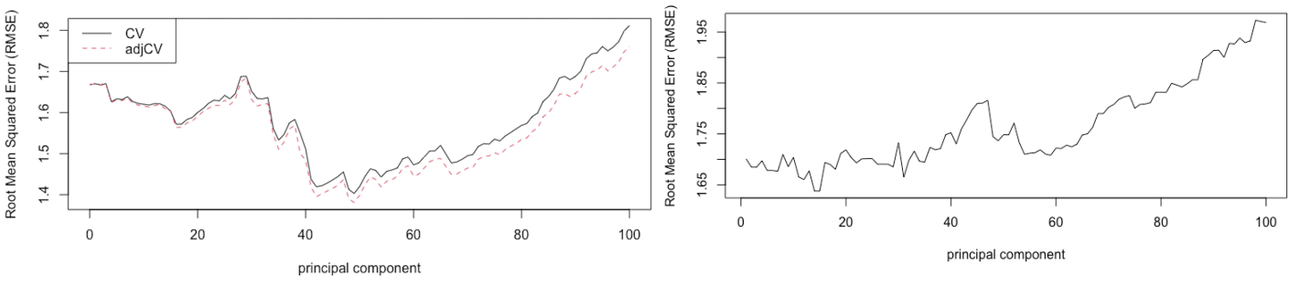

Figure 18: PCR experiment 3: RMSE against number of PCs, LHS is for training and RHS is for testing

The random seed, which causes varied partitioning of the data set, is the only difference across the three experiments. We can tell from those figures that the result of training stage may not be consistent with the testing stage. For example, in experiment 1, the training stage indicates that 37 is the best choice for the number of PCs, however, the testing stage doesn't achieve the lowest RMSE at this number. This means that, in such experiments, the model doesn't learn a generalized relationship between the β Lactoglobulin B value and the spectra data. This may be due to the following reason:

- The model cannot capture such relationship in nature, e.g., the relationship is not linear at all.
- As discussed in Question 6, the first several PCs capture the most variation of data, however, in this case, this does not lead to a good relationship. ($\Rightarrow$ Try Partial Least Square method.)
- There are too many zero values in β Lactoglobulin B, those "outliers" make the model hard to learn a generalized relationship. (I will try to solve this problem in Question 8 and 9)

The Partial Least Square model has been tried here and the results are illustrated in Figure 19.

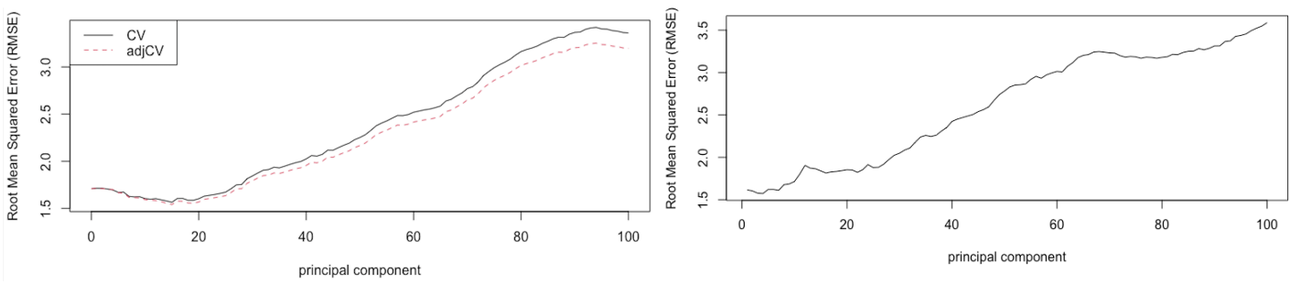

Figure 19: PLSR: RMSE against number of PCs, LHS is for training and RHS is for testing

We can tell from the results that, in this application, PLSR seems to learn a better relationship than PCR even with less number of PCs.

8 Question 08: Matrix imputation [30 marks]

Three imputation strategies are implemented in the impute function.

1. mean imputation: fill missing values with the corresponding column mean;
2. pca imputation: fill missing values with the corresponding values in the reconstructed matrix by PCA;
3. svd imputation: fill missing values with the corresponding values in the reconstructed matrix by SVD;

Mean imputation and pca imputation will be used in Question 9 but svd imputation is not.

It should be noted that the impute function is a general matrix imputation method and will fill all missing values. The question only asks to fill β Lactoglobulin B. However, I've checked that the missing values only appear in β Lactoglobulin B, so the results are equivalent. What's more, the impute function only treats NA values as missing values. However, in this question, zero values are identified as missing values. The simple solution is to transform zero values into NA values before feeding the matrix into the impute function.

First, the impute function is tested on *lena128* dataset from *filling* package. Due to the space reason, I only put the results of mean imputation and pca imputation in the report, which are illustrated in Figure 20. However, you can find the results of svd imputation in the R script.

Figure 20: test impute function, LHS for mean imputation and RHS for pca imputation

We can tell from the results that:

- pca imputation achieves better performance than mean imputation in the image restoration task.
- pca imputation achieves similar performance to svd imputation in this task. Actually, in theory, if the data is centered, both strategies are totally the equivalent.
- In pca imputation, setting the number of PCs to 10 and 25 achieves better performance than 50. More PCs don't guarantee a better imputation.

Then, the impute function is applied in protein data, which is extracted from global_data as a matrix (q8.proteins). Zero values in the matrix are transformed into NA values. Finally, the matrix is passed into the impute function to be filled. In this case, I found that:

- In PCA imputation, as k, the number of PCs increases, the method need more iterations to converge.
- In PCA imputation, as k, the number of PCs increases, the final RMSE decreases.

9 Question 09: Handle missing values in PCR task [30 marks]

Three strategies are employed to handle observations with missing values:

1. omit observations with missing values; the results are illustrated in Figure 21
2. mean imputation (zero values are transformed into NA values before imputation because the impute function only deals with NA values); the results are illustrated in Figure 22
3. pca imputation (zero values are transformed into NA values before imputation because the impute function only deals with NA values); the results are illustrated in Figure 23

Most settings are the same as Question 7 and 8. However, it should be noted that there are no observations with missing values in the test set because those observations do not provide the ground truth.

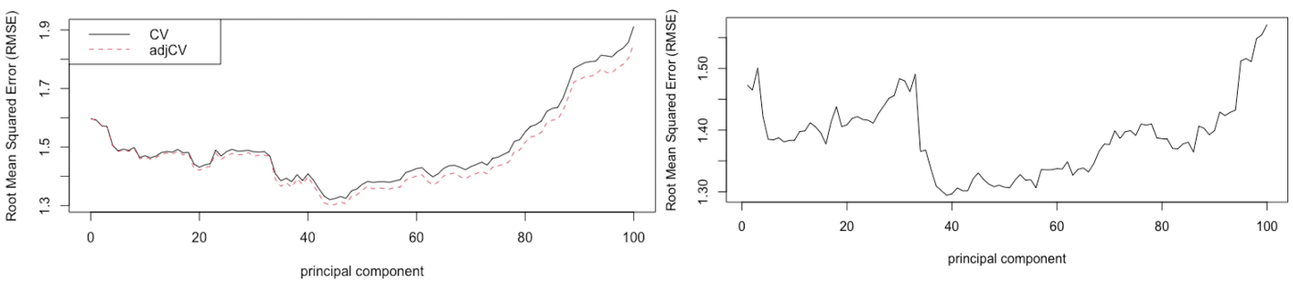

Figure 21: omit observations with zero values

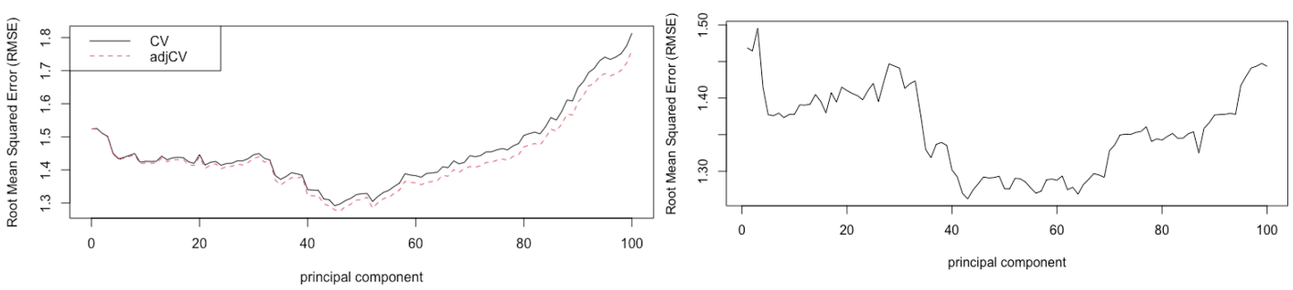

Figure 22: mean impute

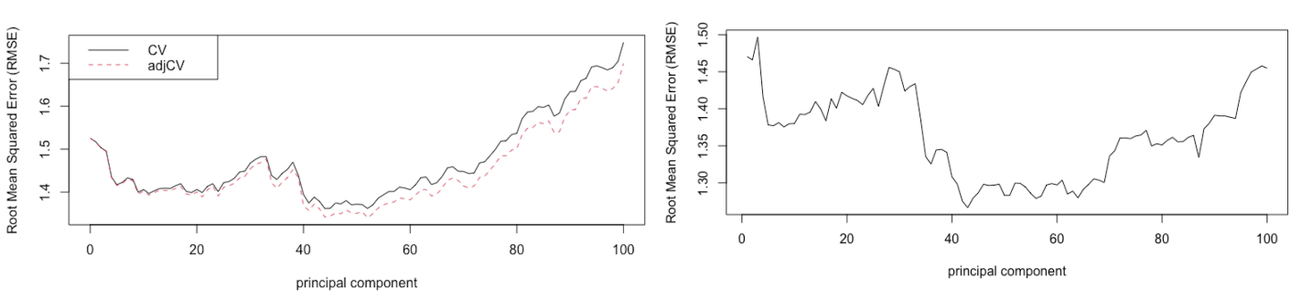

Figure 23: pca impute

In this experiment, the three strategies achieve better performance when the number of PCs is set to around 45. So, 45 is selected to compare different strategies on test set and the results are illustrated in Table 1.

stragety	RMSE on test set (k = 45)
omit observations with zero values	1.330604
mean imputation	1.282873
pca imputation	1.28659

Table 1: compare three strategies on test set

We can tell from Table 1 that:

- Both mean imputation and pca imputation achieve lower RMSE than simply omitting the observations with missing values. The reason is that if any observations are eliminated, information is lost.
- Though pca imputation performs better in image restoration task in Question 8, they both perform fairly similarly in this PCR task. We need to do more research to verify what scenarios they are each more applicable to.

Due to the variable results, this experiment has been repeated multiple times using different dataset splits (by setting different random seed at the beginning of the R script), and the conclusions are similar.