MATH38161 Multivariate Statistics and Machine Learning

Task A: Cluster analysis of the Penguin data set using K-means

Nurfahimah binti Mohd Ghazali, 10499719

1 Dataset

This report aims to analyse an Antarctic penguin `X.penguin` data set containing 4 measured variables; `bill_length_mm`; `bill_depth_mm`; `flipper_length_mm`; and `body_mass_g` on $n = 333$ penguins using K-means clustering. In the `penguins.rda` file [1] loaded in R, we are provided with three factors, the sex of the penguins: `L.sex` and two known clusters: `L.species`; `L.islands`. We start off by using descriptive statistics to analyse each variable in the data using the following R code.

```
> summary(X.penguins)
bill_length_mm  bill_depth_mm   flipper_length_mm  body_mass_g
Min.   :32.10   Min.   :13.10   Min.   :172       Min.   :2700
1st Qu.:39.50   1st Qu.:15.60   1st Qu.:190       1st Qu.:3550
Median :44.50   Median :17.30   Median :197       Median :4050
Mean   :43.99   Mean   :17.16   Mean   :201       Mean   :4207
3rd Qu.:48.60   3rd Qu.:18.70   3rd Qu.:213       3rd Qu.:4775
Max.   :59.60   Max.   :21.50   Max.   :231       Max.   :6300
```

These values are even more useful when visualised as boxplots in Figure 1. Looking at the median mark within the box, we see that the `bill_length_mm` as well as `bill_depth_mm` seem to be dispersed fairly symmetrically while the other two are slightly positively skewed. We can also note that these observations are confirmed when we compare the median and mean of each variable. From the first two columns in the summary, the medians are larger than the means, with the difference being very small; 0.51 for the first column; and 0.14 for the second. In contrast, for `flipper_length_mm` and `body_mass_g` the means are larger than medians by 4mm and 156g respectively. These differences are also considered to be small when compared to the range of the data points of each variable: `bill_length_mm` ranging from 32.10mm to 59.60mm; `bill_depth_mm` ranging from 13.10mm to 21.50mm; `flipper_length_mm` ranging from 172mm to 231mm; `body_mass_g` ranging from 2700g to 6300g.

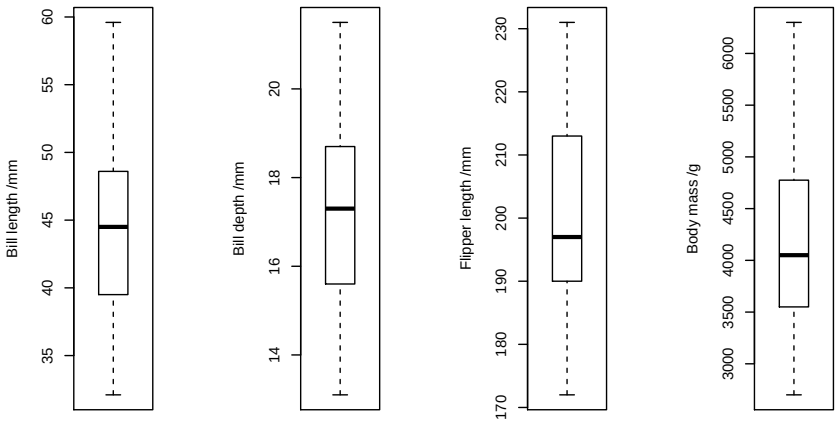

Figure 1: Boxplots of each variable in dataset

```
par(mfrow=c(1,4))
boxplot(X.penguins[,1], ylab = "Bill length /mm")
boxplot(X.penguins[,2], ylab = "Bill depth /mm")
boxplot(X.penguins[,3], ylab = "Flipper length /mm")
boxplot(X.penguins[,4], ylab = "Body mass /g")
```

Using the `cov()` commmand in R, we get the symmetric covariance matrix, $\mathbf{\Sigma}$ in (1).The signs on the covariance values tell us how the relationship between the variables are like. However, the values themselves depend on the data points and do not tell us much on how strong the relationship between the variables.

$$\mathbf{\Sigma} = \begin{pmatrix} 29.91 & -2.46 & 50.06 & 2595.62 \\ -2.46 & 3.88 & -15.95 & -748.46 \\ 50.06 & -15.95 & 196.44 & 9852.19 \\ 2595.62 & -748.46 & 9852.19 & 648372.49 \end{pmatrix} \tag{1}$$

From $\mathbf{\Sigma}$, it is possible to extract the standard deviation of each variable by calculating the square roots of its diagonal entries:

```
> sqrt(diag(cov(X.penguins)))
   bill_length_mm     bill_depth_mm flipper_length_mm       body_mass_g
         5.468668          1.969235         14.015765        805.215802
```

These values tell us how the data is dispersed, with a smaller standard error value being more favourable. The deviations are largely caused by the range, which explains why σ is very large for `body_mass_g`, considering its large range. We check that the matrix is positive definite using the command below through the positive-valued eigenvalues.

```
eigen(cov(X.penguins))$values
```

$$\lambda^T = \begin{pmatrix} 6.485e+05 & 5.074e+01 & 1.615e+01 & 2.361e+00 \end{pmatrix} \tag{2}$$

The `eigen()` function allows us to obtain the eigenvectors, which tell us the direction of the spread between each variable against each other while the eigenvalues give us the magnitude. These information is specifically vital for whitening processes.

$$V = \begin{pmatrix} 0.004 & -0.319 & 0.941 & -0.110 \\ -0.001 & 0.0868 & 0.144 & 0.986 \\ 0.015 & -0.944 & -0.305 & 0.128 \\ 1.000 & 0.016 & 0.001 & -0.0004 \end{pmatrix} \tag{3}$$

The scatterplot in Figure 2 is obtained using the R codes below. By the patterns in the plot, we see that it is possible to simplify the dataset by grouping the data in clusters. The group structure seems to be split to two or three different groupings. In the first row and column of the plots, there seems to be three distinct clusters, while the rest seem to split into two bigger groups. As mentioned previously, the data can be split into their sexes as well as the two known clusters in the data file. The dataset will be looked at as a whole since the physical difference between the sexes are small [2]. We will also be assuming that the class labels in the clusters are unknown.

```
pairs(X.penguins, labels= c("Bill Length/mm", "Bill depth/mm",
"Flipper Length/mm", "Body Mass/g"))
```

2 Methods

Before applying the algorithm to any dataset, each of its variables must be scaled. K-means clustering is an algorithmic method of clustering since the method itself is not explicitly based on a probabilistic model. In general, the method aims to minimise the within-group(unexplained) variation as well as maximise the between-group (explained) variation. Each data point x_i is grouped into exactly one of the pre-specified number of clusters K. The algorithm starts off by selecting initial K random means, $\hat{\mu}_k$ and assigning observation x_i into each K based on the Euclidean distance,

$$d_{x_i,\hat{\mu}_k} = \sqrt{\sum_{i=1}^{n}(x_i - \hat{\mu}_k)^2}$$

Then new means are calculated for each K cluster, and the data points, x_i are assigned to the new ones. The whole process is then repeated until convergence is reached. In R, the maximum times that

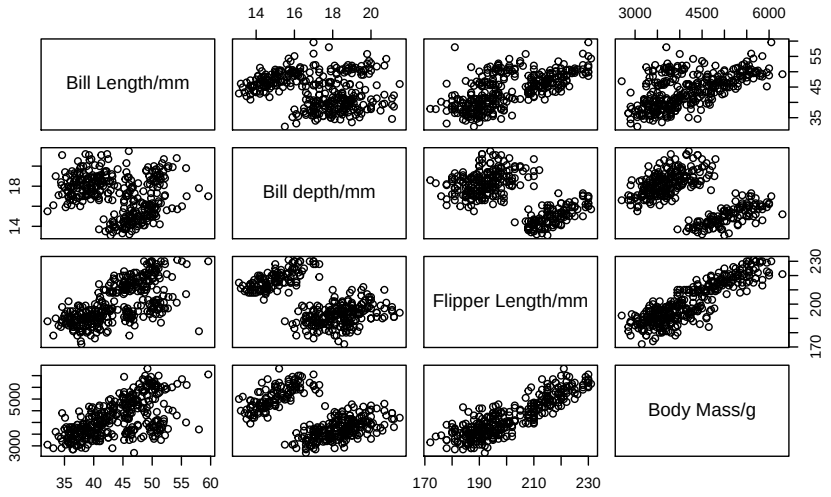

Figure 2: Scatterplot of each variable in dataset against each other

the algorithm is repeated by default is 10 times.

The command in R that carries out the K-means algorithm is `kmeans()`, however, the algorithm requires a user-input K clusters. In this report we choose $K = 3$ based on the apparent grouping in the `pairs` plot in the first section. The clusters that the algorithm produces can be compared to the known clusters in the dataset: `L.species`; `L.islands` visually through a scatterplot, and analytically through counts in a table. This comparison gives us how realistic it is to be using K-means algorithm instead of a probabilistic model-based algorithm or another algorithmic clustering method in unsupervised learning.

In order to recognise the optimal number of clusters, we compare the explained variation also known as the between group variation in for each k from 1 up to 10. Then the optimal K is the smallest k such that there is no substantial increase in explained variation as k increases. This is so that the model is not unnecessarily complex. The same can be done on the unexplained variation, choosing the smallest k where there is no significant decrease in the within-group variation as the number of clusters are increased, also known as the elbow method.

3 Results and Discussion

```
> set.seed(543261)
> Xpen = scale(X.penguins[,1:4], scale = T)
> kmc = kmeans(Xpen, 3)
```

We store the K-means algorithm including the variations within the group and between them in `kmc`. Then we can compare the clusters in `kmc` to the known clusters. Since the label assignment is done arbitrarily, if the clusters match up perfectly, we expect that there would be only one entry in each row. We first compare it with `L.species` as shown in the following R output.

```
> addmargins(table(L.species, kmc$cluster))

L.species     1   2   3 Sum
  Adelie      0   7 139 146
  Chinstrap   0  63   5  68
  Gentoo    119   0   0 119
  Sum       119  70 144 333
```

Based on the counts in the clusters, we see that the there are little samples that are classified differently than the grouping based on the species. The column margin added to the table is the classification based

on the species. We note that the counts of the first two species are distributed to two of the clusters in `kmc`. However, the misclassified values are small compared to the total counts in the cluster itself. The total counts in each class are similar to the total counts of the data when split into species. Based off of that and the counts distribution in the clusters, the classes can be roughly matched up: cluster 3 with Adelie; cluster 2 with Chinstrap; and cluster 1 with Gentoo.

```
> addmargins(table(L.islands, kmc$cluster))

L.islands      1   2   3 Sum
  Biscoe     119   2  42 163
  Dream        0  64  59 123
  Torgersen    0   4  43  47
  Sum        119  70 144 333
```

Similar to the species clustering, the column margin is the counts of the data split into the islands that the penguins live in. The counts in the K-means clusters are significantly different from the counts based on the islands. The counts of penguins on the island with the most penguin habitation- Biscoe, is assigned to all three clusters but most of the counts lying in cluster 1 and 3. The other two counts are dispersed to cluster 2 and 3. Compared to the species classification, the grouping by the island has more samples that are misplaced in the clusters as it is harder to recognise which island matches the cluster obtained from K-means.

To visualise the clustering, it is sufficient to look at the plot of `bill_depth_mm` against `bill_length_mm`, where the colours indicate the clustering done using K-means and the point characters indicate the known clusters in the data file. The point characters in the left plot are based on `L.species` and in the right plot are based on `L.islands`. Looking at the left plot in Figure 3, we see from the point characters that the species' classes are split to three distinct sections. In contrast, in the right plot, there are sections where the point characters are mixed up, mainly in the upper left corner.

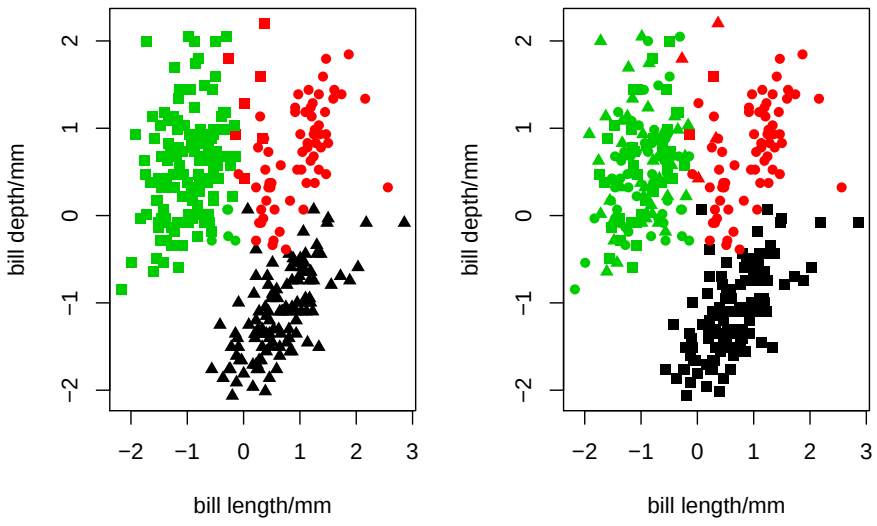

Figure 3: Scatterplot the first two variables in the dataset

```
> par(mfrow=c(1,2), mai = c(0.8, 0.8, 0.5, 0.3))
> plot(Xpen, col = kmc$cluster, pch = as.integer(L.species)+14, main="",
    xlab = "bill length/mm", ylab = "bill depth/mm")
> plot(Xpen, col = kmc$cluster, pch = as.integer(L.islands)+14, main="",
    xlab = "bill length/mm", ylab = "bill depth/mm")
```

The wrong classification of the counts can be seen clearly in the colours of the point characters. Comparing the comments made from the counts dispersion to its visual representation, the K-means

algorithm seem to be more similar to classification by species compared to by islands. This is clear when looking at the mismatch of colours and shapes in the left graph, confirmed with the table of counts analysed previously. On the surface, it seems that this may be due to how the variables are related to the clustering.

We finish up this report by finding the optimal number of clusters to be specified for the algorithm. Figure 4 is produced by running the R codes listed underneath the plot. The number of clusters that are optimal for the algorithm is recognised from the variation for that cluster, where both of the graphs has a turning point, and starts to converge as k increases. From the graph, we clearly see that this number is $K = 3$. Therefore we can confirm that the visual observations made from the scatterplot in Figure 1 matches up with the analysis on the K-means algorithm.

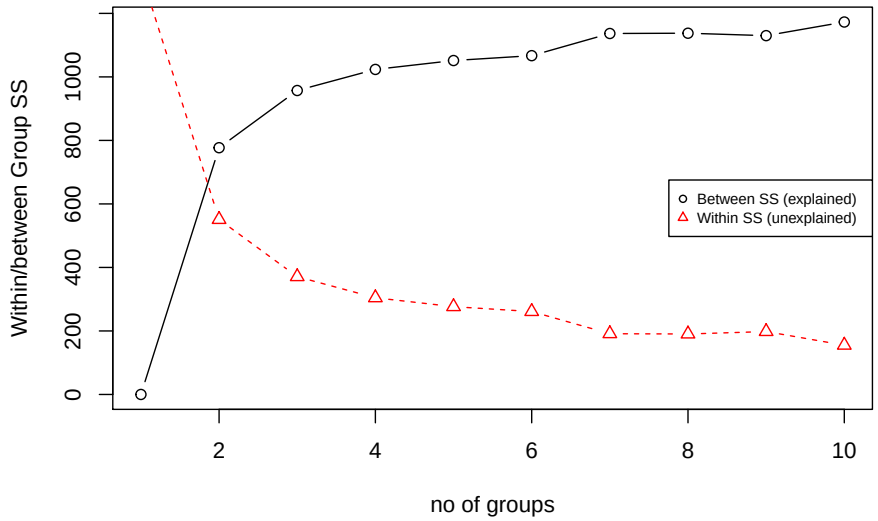

Figure 4: Plot of within group and between group variation against number of clusters

```
> maxk = 10
> bssvec = numeric(maxk)
> wssvec = numeric(maxk)
> for (k in 1:maxk) {
+   kmcn = kmeans(Xpen, k)
+   bssvec[k] = kmcn$betweenss
+   wssvec[k] = kmcn$tot.withinss
+ }
> par(mfrow=c(1,1))
> plot(1:maxk, bssvec, type="b", xlab="no of groups", ylab= "Within/between
    Group SS", main="")
> points(1:maxk, wssvec, type="b", col=2, lty="dashed", pch=2)
> legend("right", c("Between SS (explained)", "Within SS (unexplained)"),
    col=c(1,2), pch=c(1,2), cex = 0.7)
```

This section concludes this report on application of K-means algorithm on the `X.penguin` data set. All in all, K-means method of clustering is relatively simple to understand and implement using R as demonstrated in this report. However, the need of a user-input number of clusters is not favourable.

References

[1] K. G. Allison Horst, Alison Hill, "Palmer penguins," 2020. Last accessed 4 December 2021.

[2] D. A. Horvath, "How do penguins find their mate in a sea of tuxedos?," 2015. Last accessed 4 December 2021.