Practical Data Science With R
Practical 6B HW
Felix Carter

Introduction

By studying data from coffee samples obtained using liquid chromatrography – mass spectrometry (LC-MS), we can investigate how coffee from different countries differs in taste, and how this relates to the chemical properties observed of the coffee. In this dataset, coffee from Brazil, Columbia, Ethiopia, Java and Kenya was sampled three times for each country of origin, and three repeats of the analysis of each sample were taken. The taste scores measured were citrus, berry, floral, nuts, chocolate and spice, each given a rating by an 'expert coffee taster' on a scale of 1 to 5.

Results

After performing PCA on the data, we get the following scores plot:

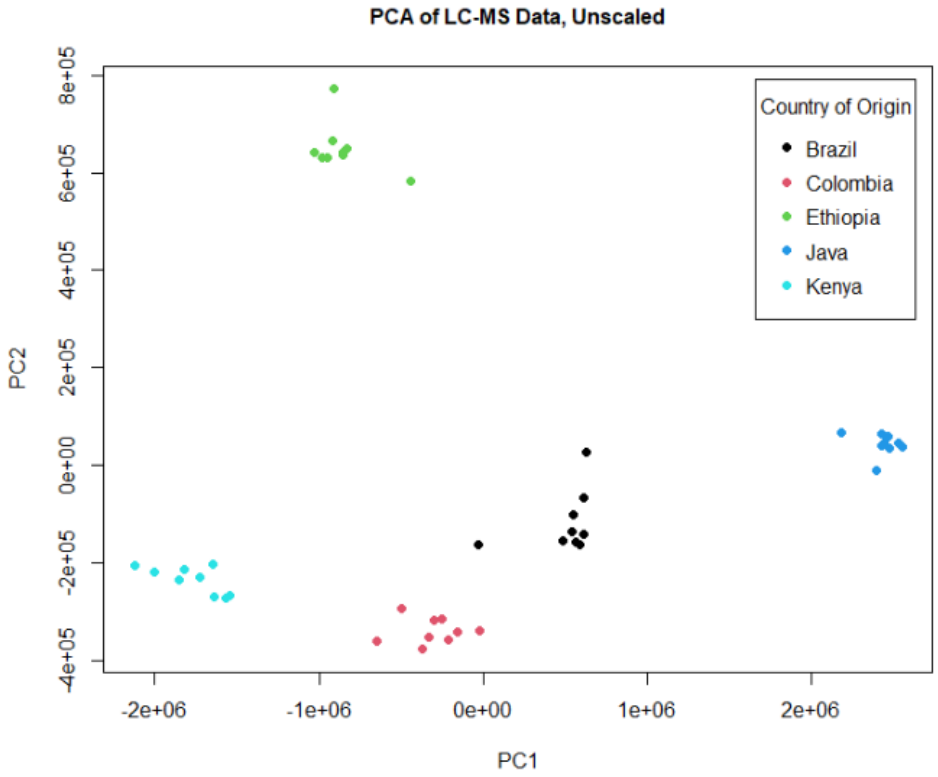

Figure 1: A scores plot of PC1 against PC2 for the coffee dataset, coloured by country of origin.

Looking at this scores plot, we can see clear separation between each country of origin. Ethiopian coffee is best separated from the other countries by PC2, whereas PC1 shows best separation between the other countries, however there is separation between coffee from all countries along both PC1 and PC2.

After combining the data matrix with a response matrix we perform PLSR to get our model. This uses the default ('kernelpls') method with scaled data.

By plotting the RMSEPs, we get the following:

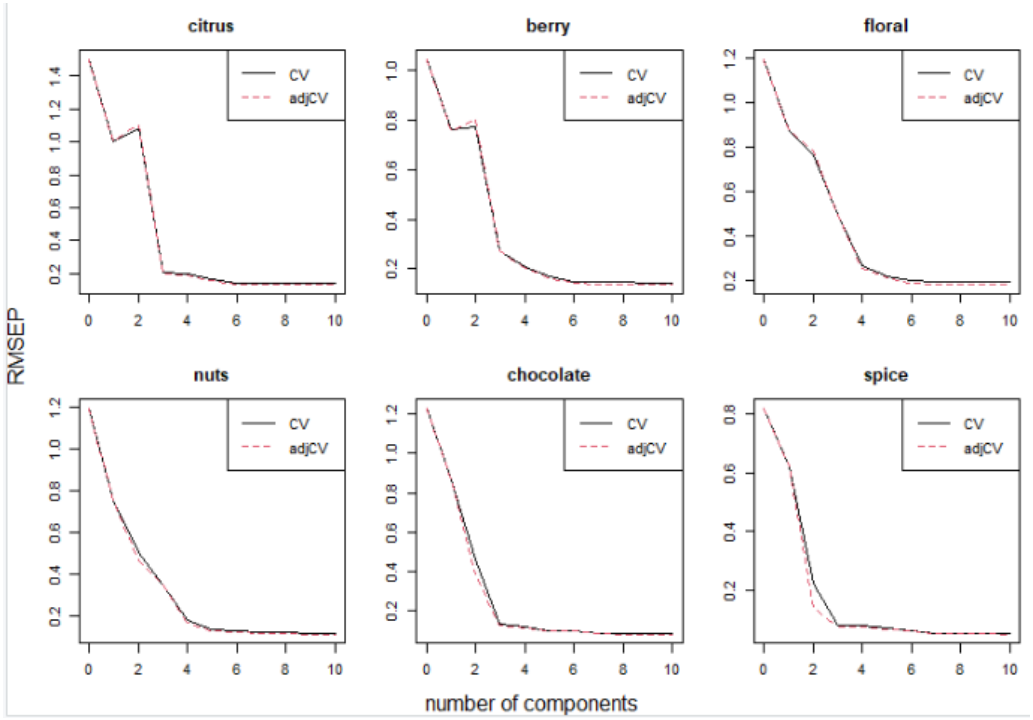

Figure 2: A plot of the RMSEPs for each of the six different taste levels.

Looking at these six plots, we can see that using six components gives a low RMSEP for each of the different taste levels, so well will re-do our model with six components.

The following biplot shows the scores and loadings for our new model with six components:

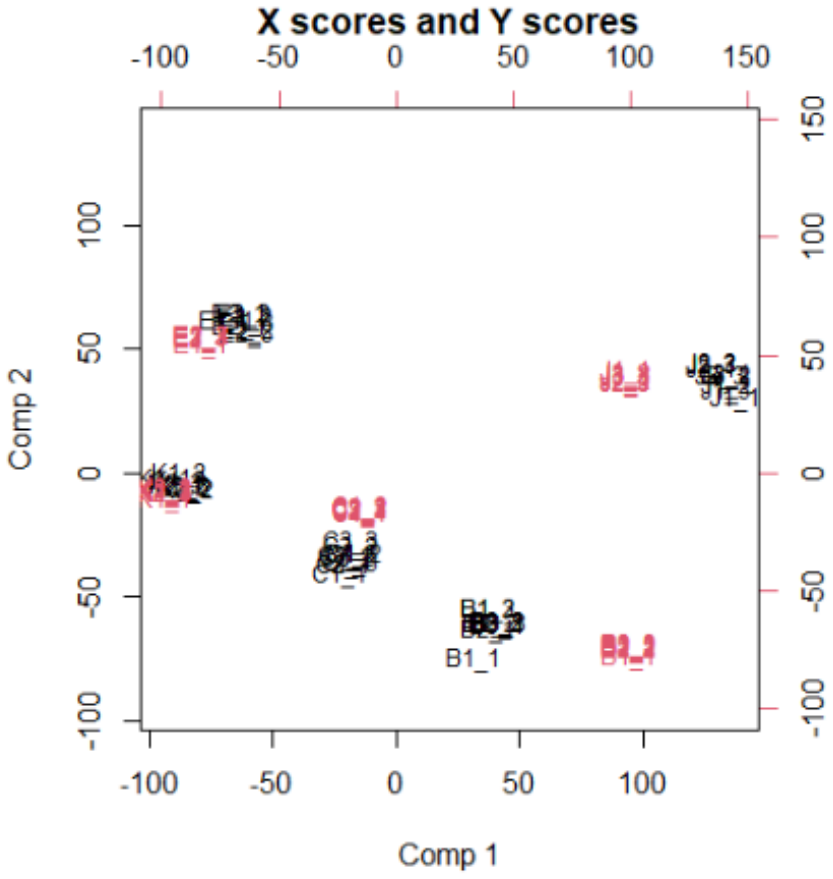

Figure 3: A biplot of the X and Y scores for the model, with the letter corresponding to the country of origin (e.g 'B' for Brazil), and the number of the observation, followed by the number of the analysis of that observation.

There seems to be some correlation between the X and Y scores as their points are near each other on the plot, however for each of the countries there is some separation between the X and Y scores. Some appear to be more closely correlated than others: the Ethiopian and Kenyan X and Y scores seem to have the closest correlation, followed by the Colombian scores. The Java and Brazil scores show clear separation between X and Y so are probably not closely correlated.

We can then produce a biplot of the Y scores and Y loadings:

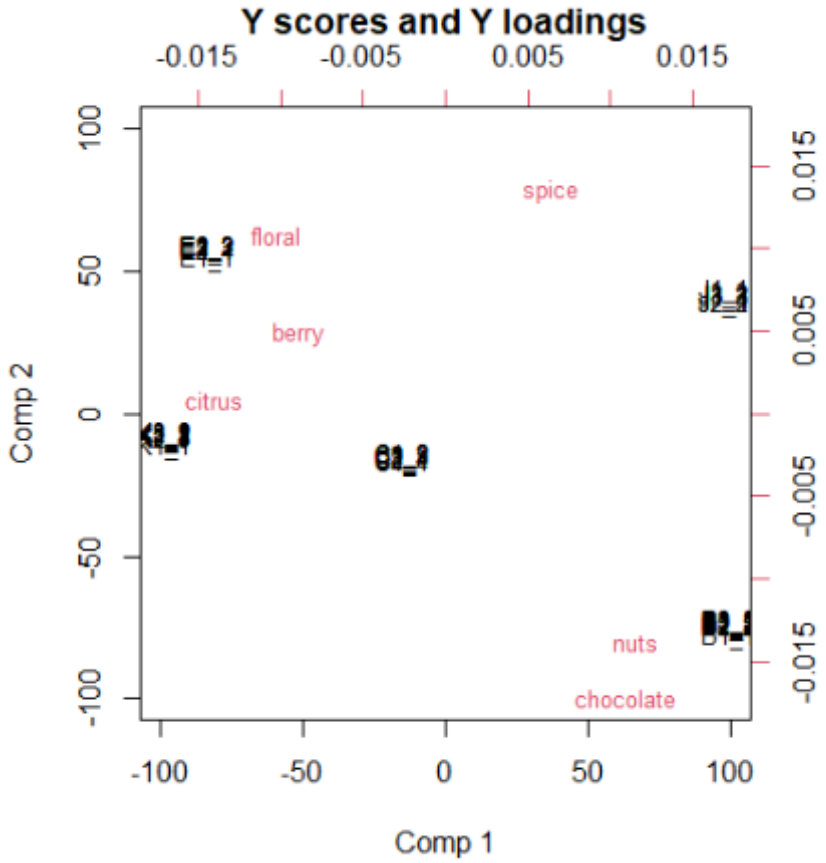

Figure 4: A biplot of the Y scores and Y loadings for the model.

From this, we can see that observations from Brazil are closely related to the nuts and chocolate taste levels, Kenyan observations are closely related to citrus tastes, and Ethiopian observations are related to floral tastes most closely. Java observations may have some relation to spice taste levels, and Columbian observations don't seem to relate closely to any particular taste levels.

The table 'Output 1' in the appendix shows the cumulative variances accounted for, for the data. From this, we find that component 1 accounts for 36.37% of the variance in the model, component 2 accounts for 11.62% (47.99%-36.37% = 11.62), and so on. This can be summarised in the following table:

Component	1	2	3	4	5	6
% of Variance	36.37	11.62	8.30	4.81	6.45	2.90

Table 1: A table showing the percentage of variance accounted for in X by each component, for the model.

The R output in 'Output 1' shows the percentage of variance explained broken down into each of the six taste levels, as well as just for X. In PCA, this wouldn't be included, but for PLSR it is. This is because in PLSR the derived directions are highly correlated with class[1].

Using the values from Table 1, we can find the 40 variables with the highest combined loadings. These are stored in the dataframe 'datasmall'. These 40 variables are given by the entries in 'Output 2' in the appendix.

Making a linear regression model for the response variable chocolate based on the 'datasmall' dataset, we get the following plot:

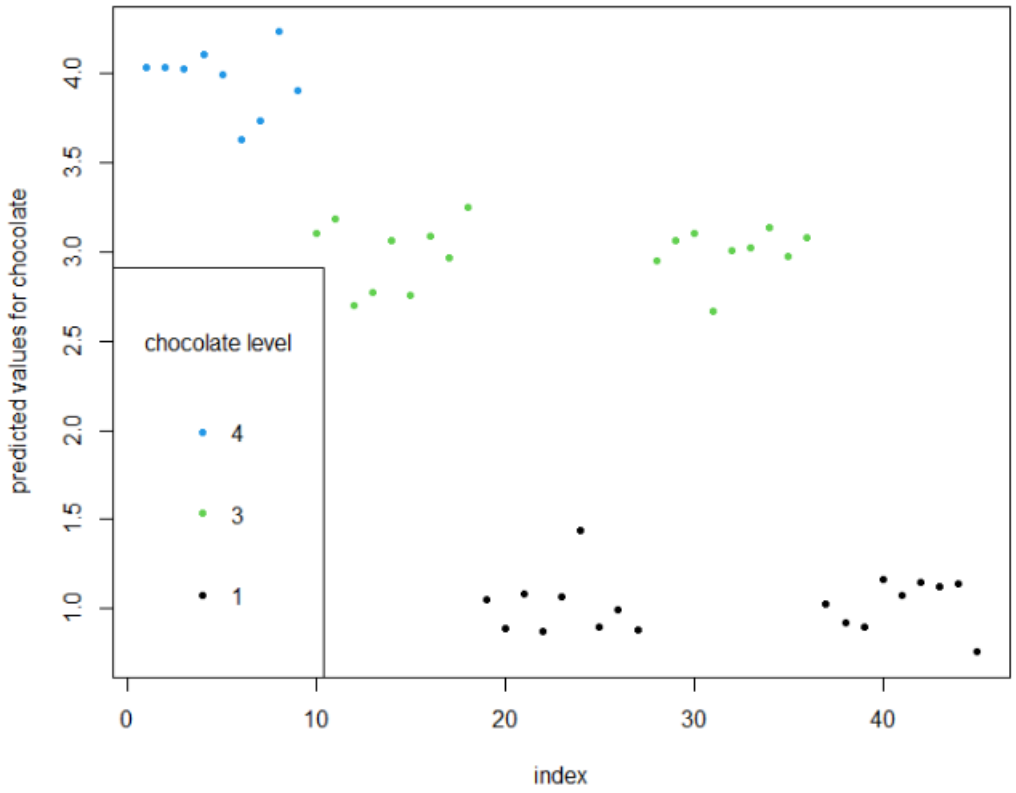

Figure 5: A plot of the predicted values for chocolate levels, coloured by their actual level.

This model seems to fit the data well, as each predicted value is within 0.5 of the actual chocolate level so would predict the correct value if rounded to the nearest integer. Therefore this is a good model.

After using the step function to reduce the number of variables, we get a new model:

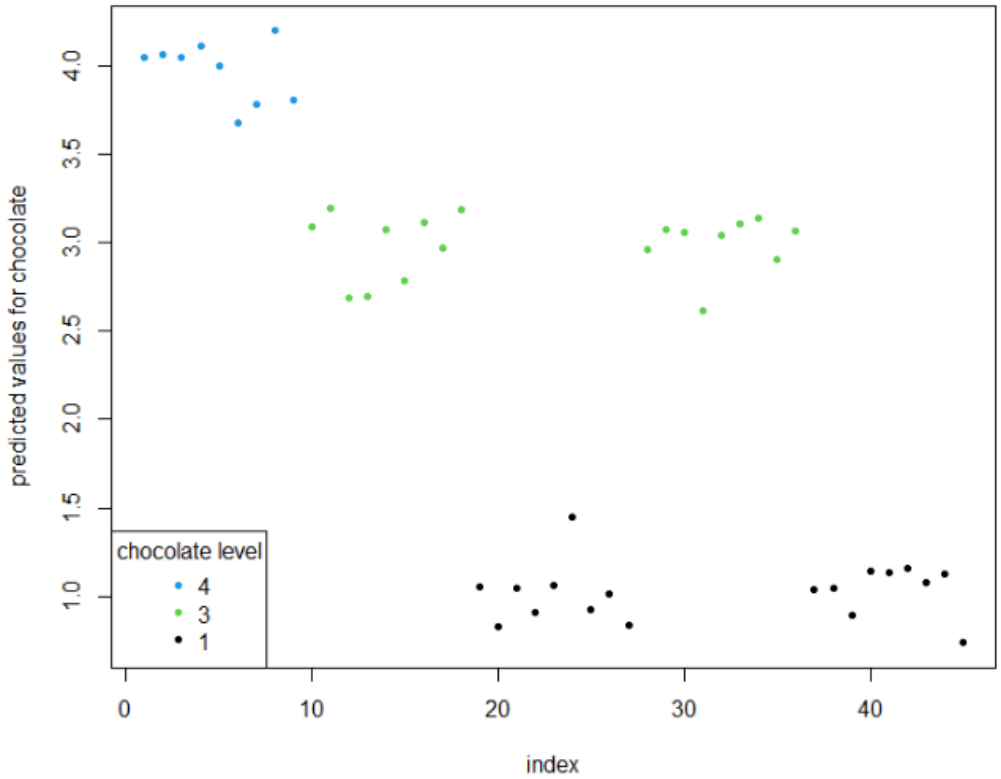

Figure 6: A plot of the predicted values for chocolate levels, coloured by their actual level. This is from the reduced model.

This model gives very similar predicted values to the previous one – it seems to fit the data well, as each predicted value is within 0.5 of the actual chocolate level, so would predict the correct value if rounded to the nearest integer. Therefore this is also a good model for the chocolate response variable.

The significance level of each variable in this model can be found in 'Output 3' in the appendix. We can see from this that eight of the variables are significant at the 0.001 level, a further eleven variables are significant at the 0.01 level, a further five variables are significant at the 0.05 level, and a further two variables are significant at the 0.1 level.

We can then make a linear regression model relating the 'datasmall' dataset to the nuts response variable. From this we obtain the following plot:

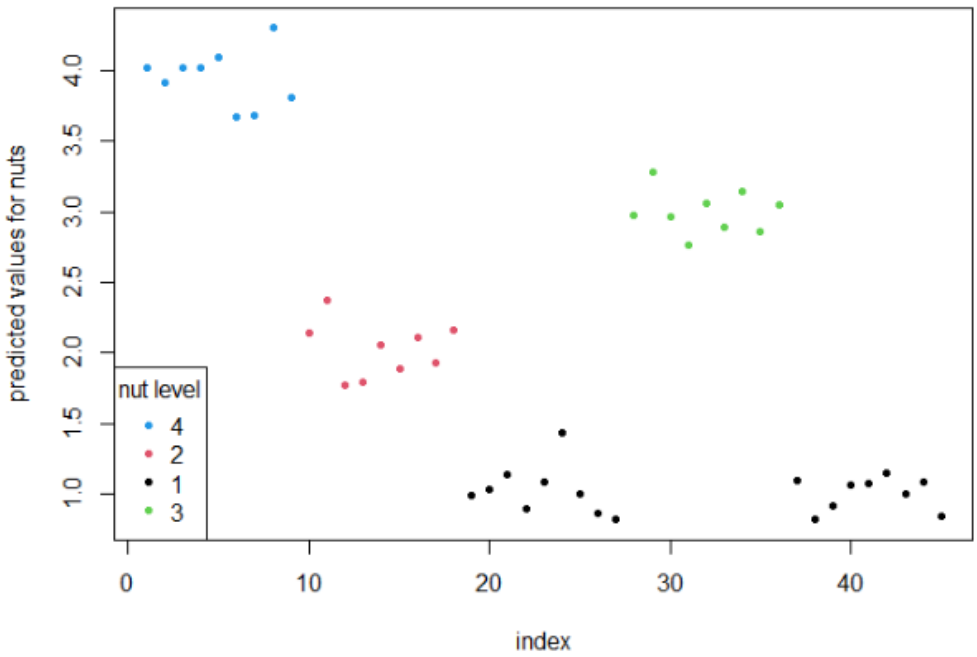

Figure 7: A plot of the predicted values for nut levels, coloured by their actual level.

Overall, this model is a good model as it predicts values for the nut level within 0.5 of the actual score given, so would predict the nut level if rounded to the nearest integer.

Making a model for the response variable citrus, we get the following plot:

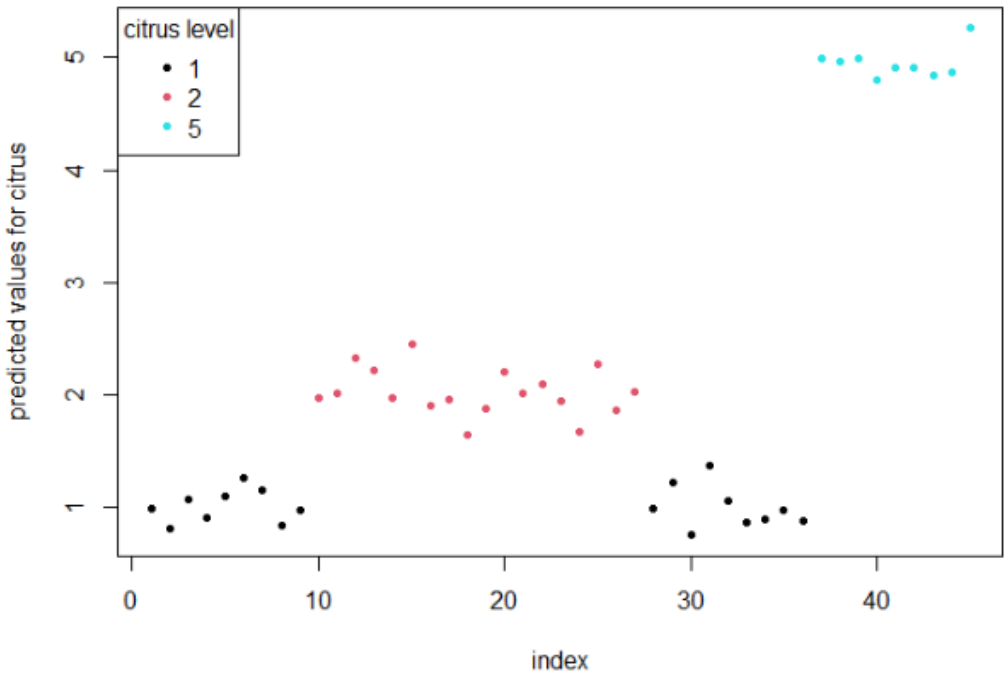

Figure 8: A plot of the predicted values for citrus levels, coloured by their actual level.

Once again, we can see that this is a model and predicts the data well, as each predicted value matches the actual citrus level to the nearest integer (it is within 0.5 of the actual citrus value).

Making a model for the response variable floral, we get the following plot:

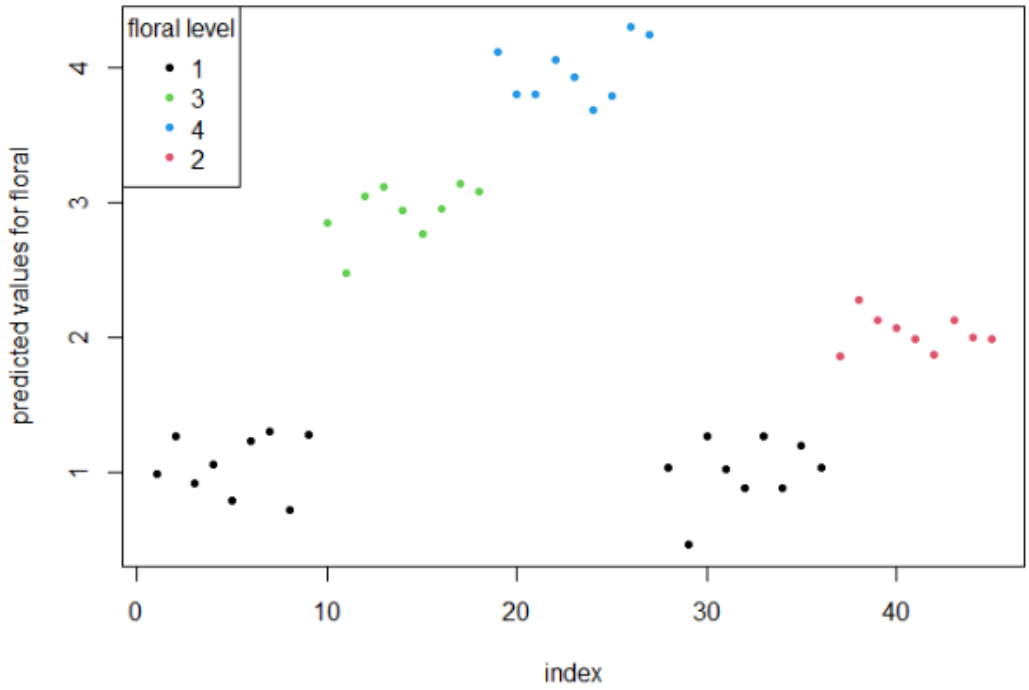

Figure 9: A plot of the predicted values for floral levels, coloured by their actual level.

We can see that this model predicts the data fairly well, as most of the predicted values match the actual floral level to the nearest integer. However, observation C1_2 (the 11th observation) has predicted value 2.478 when its actual floral value is 3, so would be classified incorrectly by this model if rounded to the nearest integer. Also, the observation J1_2 (the 29th observation) has the value 0.460 which would be rounded to 0 to the nearest integer, however the taste scale is from 1 to 5.

If we were to classify these predictions based on the predicted value to the nearest integer, this would give an accuracy of 95.6%, so this is still a good model with a pretty high accuracy rate.

Making a model for the response variable berry, we get the following plot:

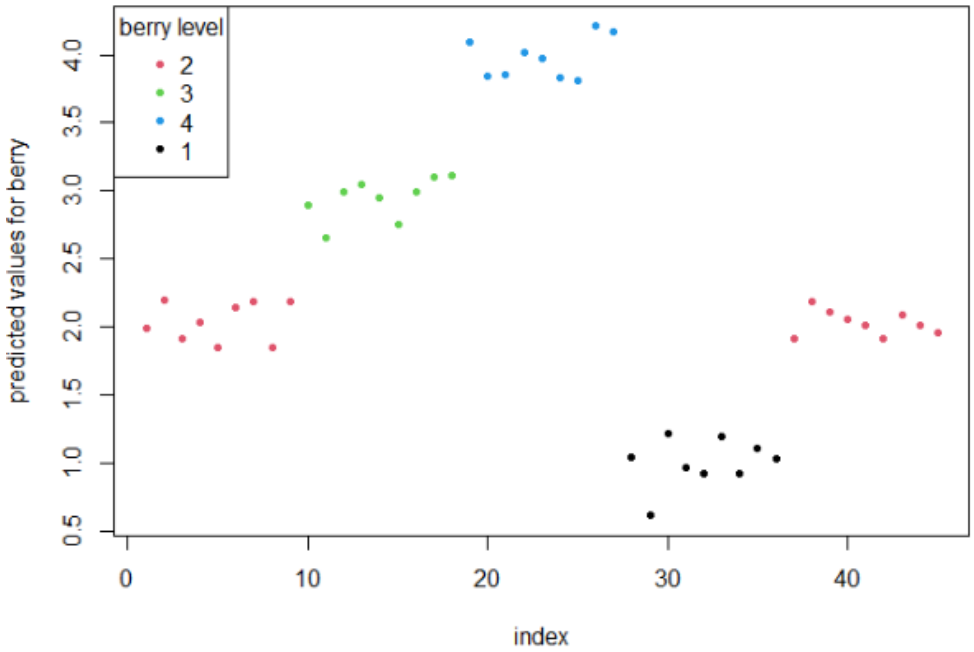

Figure 10: A plot of the predicted values for berry levels, coloured by their actual level.

This model predicts the response variable berry well, as each predicted value matches the actual berry score to the nearest integer. Therefore this is a good model.

Making a model for the response variable spice, we get the following plot:

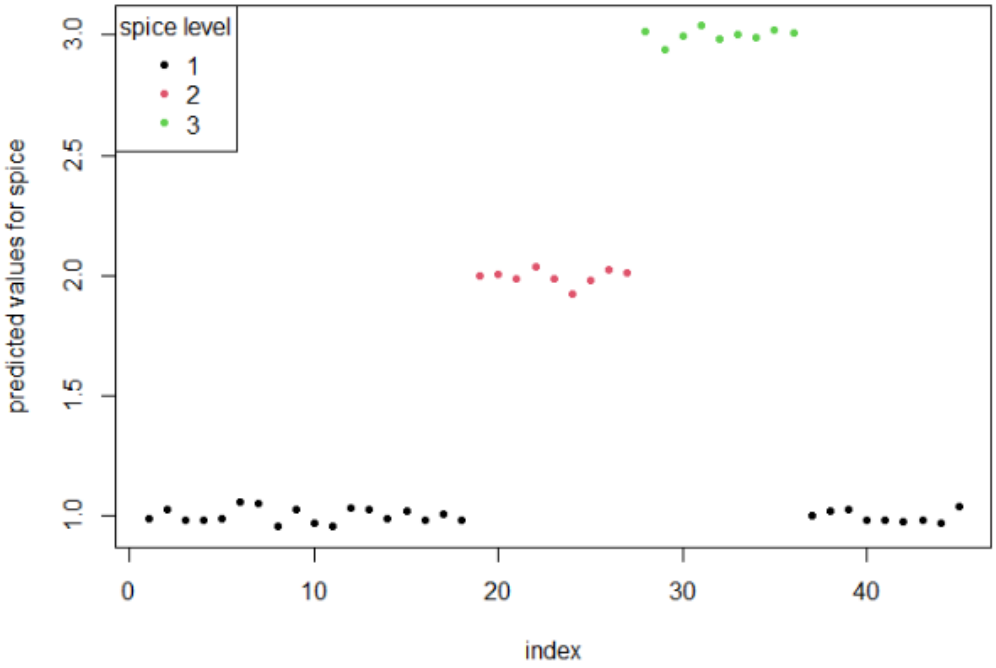

Figure 11: A plot of the predicted values for spice levels, coloured by their actual level.

This model is a good model and predicts the spice levels well, as each predicted value is within 0.5 of the actual spice level given by the coffee taster.

Having used the 40 variables chosen by PLSR to create a model for each of the six response variables, we have been able to get a good model for each. Of each of the 6 models, only the floral model had any incorrect predicted values if we were to round to the nearest integer for taste level. We could try to use the step function to improve this model. (For comparison, the original floral plot can be found in Figure 9).

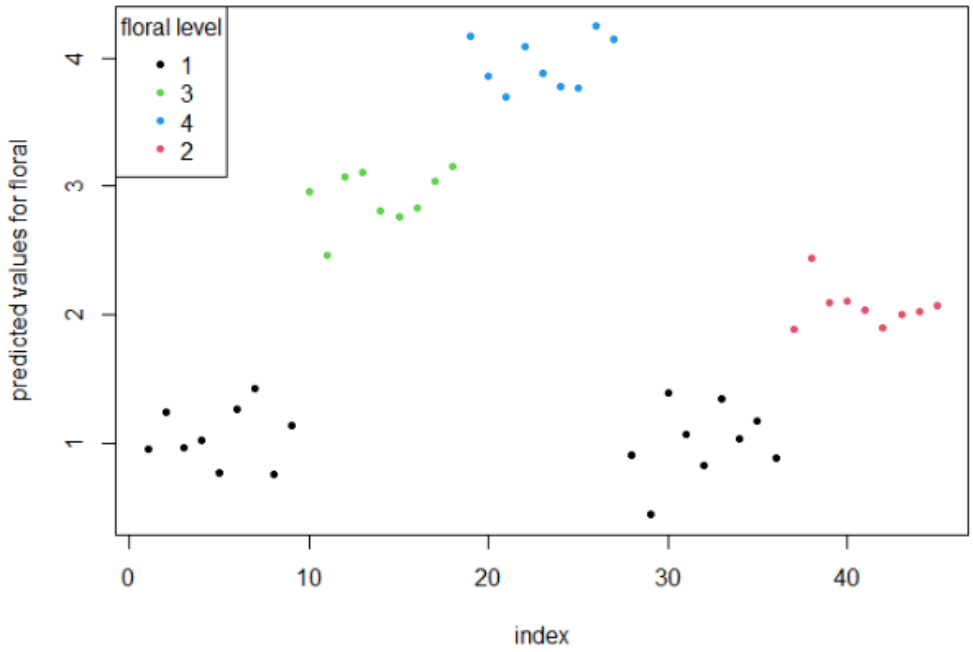

Figure 12: A plot of the predicted values for floral levels, coloured by their actual level. This is from the reduced floral model.

Sadly, this model still predicts the same observations incorrectly (C1_2 and J1_2, when rounded to the nearest integer) as before, even after having used the step function, so we have not been able to improve this model. However, this model still correctly predicts the other 43 observations to the nearest integer (giving an accuracy rate of 95.6% in this way) so is still a pretty good model overall.

Overall, this analysis may be flawed because only three samples were taken of each single-origin coffee, and each of these was analysed using LC-MS three times. The three samples per coffee may not be representative of the chemical properties of the coffee type overall. Additionally, if analysing the same sample three times you are likely to get similar results for each, meaning that these observations are likely to have correlated values.

References

1) Data Mining: Practical Machine Learning Tools and Techniques, by Ian H. Witten, Eibe Frank, Mark A. Hall, and Christopher J. Pal (p.307-308)

Appendix: R outputs

```
TRAINING: % variance explained
           1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
X            36.37    47.99    56.29    61.10    67.55    70.45
citrus       56.58    56.66    99.13    99.18    99.66    99.94
berry        47.33    51.97    95.41    98.50    99.18    99.91
floral       46.88    64.62    86.26    99.05    99.12    99.81
nuts         61.13    89.67    93.20    99.42    99.70    99.77
chocolate    51.63    93.07    99.36    99.66    99.76    99.79
spice        41.06    99.45    99.51    99.52    99.72    99.82
```

Output 1: Table 1: Cumulative 'variance explained in X' values for the model.

```
> topvarnums
 [1] 12318 11291  6694  5759  2373 12345 12244
 [8] 12246  5653   824 12273  6283  6287  6332
[15] 12250 12284 12242 12249   144 12253 12243
[22]  6284  6293  9963 12285 12257 12265 12254
[29] 14563 12317 13248 12256  5529  9958 12255
[36]  6286  6292  5624  5332  5356
```

Output 2: A table showing the column numbers of the top 40 variables with the highest combined loadings.

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.8583417  4.2867004   0.434 0.673844
V12318      -0.0032903  0.0009903  -3.323 0.007714 **
V6694       -0.0088131  0.0015181  -5.805 0.000172 ***
V5759       -0.0005206  0.0002953  -1.763 0.108409
V2373        0.0033727  0.0008777   3.843 0.003250 **
V12345       0.0017291  0.0010065   1.718 0.116558
V12244      -0.0113851  0.0018684  -6.093 0.000117 ***
V12246      -0.0146590  0.0038487  -3.809 0.003436 **
V824        -0.0012038  0.0003659  -3.290 0.008149 **
V12273       0.0093595  0.0045656   2.050 0.067506 .
V6283       -0.0053062  0.0032582  -1.629 0.134466
V6287       -0.0283892  0.0079860  -3.555 0.005226 **
V6332       -0.0029538  0.0009278  -3.184 0.009757 **
V12250       0.0007700  0.0006464   1.191 0.261113
V12284       0.0724273  0.0090420   8.010 1.16e-05 ***
V12242      -0.0086219  0.0023193  -3.717 0.003992 **
V12249       0.0023736  0.0009749   2.435 0.035163 *
V144        -0.0110014  0.0010708 -10.274 1.24e-06 ***
V12253      -0.0013092  0.0015192  -0.862 0.408980
V6284       -0.0104502  0.0077190  -1.354 0.205598
V9963       -0.0021371  0.0008307  -2.573 0.027767 *
V12257       0.0094334  0.0032536   2.899 0.015852 *
V12265      -0.0228930  0.0038528  -5.942 0.000143 ***
V14563      -0.0076542  0.0012985  -5.895 0.000152 ***
V12317      -0.0082334  0.0021615  -3.809 0.003434 **
V13248       0.0122898  0.0046921   2.619 0.025629 *
V12256      -0.0015698  0.0021209  -0.740 0.476206
V5529       -0.0007044  0.0001881  -3.744 0.003818 **
V9958        0.0021206  0.0004380   4.841 0.000680 ***
V12255       0.0059869  0.0022182   2.699 0.022352 *
V6286        0.0063171  0.0019603   3.222 0.009137 **
V6292        0.0326605  0.0063537   5.140 0.000438 ***
V5624        0.0035354  0.0019151   1.846 0.094650 .
V5332       -0.0001977  0.0001488  -1.329 0.213420
V5356        0.0008198  0.0002239   3.661 0.004380 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3534 on 10 degrees of freedom
```

Output 3: An R output for the linear regression model relating datasmall to the response variable ‘chocolate’, showing the significance level for each of the variables.

Appendix: R code Used

```
setwd("~/R/PracDataSci")
#coffee chromotography dataset
data = read.table("coffee-origin.txt") #LC-MS data
info = read.table("origin-info.txt", header =TRUE) #info

#Q1 PCA : scores for PC1, PC2, coloured by origin

cofpc = prcomp(data)

plot(cofpc$x[ ,1], cofpc$x[ ,2], main = "PCA of LC-MS Data, Unscaled", cex.main =1, xlab= "PC1", ylab= "PC2",
col =as.factor(info$origin),pch =19)
#text(cofpc$x[,1], cofpc$x[,2]+2,labels=info$origin, cex=0.5) #adds label above each point, red is W
#used to check colours match up etc
legend("topright",inset = 0.02,title = "Country of Origin", legend = unique(info$origin), col =c(1,2,3,4,5), pch
=19) #fill are the colours used
#scores for first 2 PCs, coloured by origin

#Q2
```

```
row.names(data) = info$name
coffee.df = data.frame(X=I(as.matrix(data)),Y=I(as.matrix(info[,5:10])))
library("pls")
#create train/test data
library("caret")

set.seed(123)

model <-plsr(Y ~., ncomp = 10, validation = "CV", scale = TRUE, data= coffee.df)#default method, CV, 10
comps, scaled
#predict Y?

summary(model)

#Q3 plot rmsep

plot(RMSEP(model),legendpos="topright")
#decide how many components would be best for classification
selectNcomp(model, plot=T) #doesnt work as not univariate
#see from plots that ncomp=6 gives low errors
model <-plsr(Y ~., ncomp = 6, validation = "CV", scale = TRUE, data= coffee.df)#default method, CV, 10
comps, scaled

#Q4
biplot(model, which = "scores", cex =0.8) #comps = 1:2 etc to specify comps
#which can be one of x, y, scores, loadings

#Q5
biplot(model,which="y", cex =0.8)

#Q6
summary(model)

#Q7
#weight loadings for each component
top = 36.37*model$loadings[,1]*model$loadings[,1] + 11.62*model$loadings[,2]*model$loadings[,2]
+8.30*model$loadings[,3]*model$loadings[,3] + 4.81*model$loadings[,4]*model$loadings[,4]+
6.45*model$loadings[,5]*model$loadings[,5]+ 2.90*model$loadings[,6]*model$loadings[,6]

#find the 40 variables with highest combined loadings
topvarnums = order(top)[18306:18345]
topvarnums
datasmall = data[,topvarnums] #reduce the data

#Q8
model <-lm(info$chocolate~ ., data = datasmall)

index = c(1:45) #because 45 obs
plot(index, model$fitted.values, ylab= "predicted values for chocolate", pch= 20, col = info$chocolate)
legend("bottomleft", pch= 20, legend = unique(info$chocolate), title = "chocolate level", col =
unique(info$chocolate))

table(true = info$chocolate, predicted = round(model$fitted.values)) #rounds to nearest 0.5
```

```
#can see how many obs are predicted correctly if rounded to the nearest integer

summary(model)


newmodel = step(model)
summary(newmodel)

plot(index, newmodel$fitted.values, ylab= "predicted values for chocolate", pch= 20, col = info$chocolate)
legend("bottomleft", pch= 20, legend = unique(info$chocolate), title = "chocolate level", col =
unique(info$chocolate))
table(true = info$chocolate, predicted = round(newmodel$fitted.values)) #rounds to nearest 0.5


#Q9
model <-lm(info$nuts~ ., data = datasmall)
plot(index, model$fitted.values, ylab= "predicted values for nuts", pch= 20, col = info$nuts)
legend("bottomleft", pch= 20, legend = unique(info$nuts), title = "nut level", col = unique(info$nuts))
table(true = info$nuts, predicted = round(model$fitted.values)) #rounds to nearest 0.5

model <-lm(info$citrus~ ., data = datasmall)
plot(index, model$fitted.values, ylab= "predicted values for citrus", pch= 20, col = info$citrus)
legend("topleft", pch= 20, legend = unique(info$citrus), title = "citrus level", col = unique(info$citrus))
table(true = info$citrus, predicted = round(model$fitted.values)) #rounds to nearest 0.5

model <-lm(info$floral~ ., data = datasmall)
plot(index, model$fitted.values, ylab= "predicted values for floral", pch= 20, col = info$floral)
legend("topleft", pch= 20, legend = unique(info$floral), title = "floral level", col = unique(info$floral))
table(true = info$floral, predicted = round(model$fitted.values)) #rounds to nearest 0.5

model$fitted.values[11] #would be predicted incorrectly
model$fitted.values[29]

model <-lm(info$berry~ ., data = datasmall)
plot(index, model$fitted.values, ylab= "predicted values for berry", pch= 20, col = info$berry)
legend("topleft", pch= 20, legend = unique(info$berry), title = "berry level", col = unique(info$berry))
table(true = info$berry, predicted = round(model$fitted.values)) #rounds to nearest 0.5

model <-lm(info$spice~ ., data = datasmall)
plot(index, model$fitted.values, ylab= "predicted values for spice", pch= 20, col = info$spice)
legend("topleft", pch= 20, legend = unique(info$spice), title = "spice level", col = unique(info$spice))
table(true = info$spice, predicted = round(model$fitted.values)) #rounds to nearest 0.5

#now we need to redo the floral model to try and improve it
model <-lm(info$floral~ ., data = datasmall)
newmodel = step(model)
summary(newmodel)

plot(index, newmodel$fitted.values, ylab= "predicted values for floral", pch= 20, col = info$floral)
legend("topleft", pch= 20, legend = unique(info$floral), title = "floral level", col = unique(info$floral))
table(true = info$floral, predicted = round(newmodel$fitted.values)) #check if it has correct predictions
when rounded
```