Homework05

Alexis Bryant

2025-09-25

```
knitr::opts_chunk$set(echo = TRUE)
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.2.3

## Warning: package 'tibble' was built under R version 4.2.3

## Warning: package 'tidyr' was built under R version 4.2.3

## Warning: package 'readr' was built under R version 4.2.3

## Warning: package 'purrr' was built under R version 4.2.3

## Warning: package 'dplyr' was built under R version 4.2.3

## Warning: package 'stringr' was built under R version 4.2.3

## Warning: package 'forcats' was built under R version 4.2.3

## Warning: package 'lubridate' was built under R version 4.2.3

## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(Rtsne)
```

```
## Warning: package 'Rtsne' was built under R version 4.2.3
```

1.1 Question 1

a) Import your data

```r
wine <- read_csv("wine.csv")
```

```
## Rows: 178 Columns: 14
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (14): Alcohol, Malicacid, Ash, Alcalinity_of_ash, Magnesium, Total_pheno...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

b) Check out the columns present using one of R's data frame summary.

```r
library(dplyr)
glimpse(wine)
```

```
## Rows: 178
## Columns: 14
## $ Alcohol                        <dbl> 14.23, 13.20, 13.16, 14.37, 13.24, 14.2~
## $ Malicacid                      <dbl> 1.71, 1.78, 2.36, 1.95, 2.59, 1.76, 1.8~
## $ Ash                            <dbl> 2.43, 2.14, 2.67, 2.50, 2.87, 2.45, 2.4~
## $ Alcalinity_of_ash              <dbl> 15.6, 11.2, 18.6, 16.8, 21.0, 15.2, 14.~
## $ Magnesium                      <dbl> 127, 100, 101, 113, 118, 112, 96, 121, ~
## $ Total_phenols                  <dbl> 2.80, 2.65, 2.80, 3.85, 2.80, 3.27, 2.5~
## $ Flavanoids                     <dbl> 3.06, 2.76, 3.24, 3.49, 2.69, 3.39, 2.5~
## $ Nonflavanoid_phenols           <dbl> 0.28, 0.26, 0.30, 0.24, 0.39, 0.34, 0.3~
## $ Proanthocyanins                <dbl> 2.29, 1.28, 2.81, 2.18, 1.82, 1.97, 1.9~
## $ Color_intensity                <dbl> 5.64, 4.38, 5.68, 7.80, 4.32, 6.75, 5.2~
## $ Hue                            <dbl> 1.04, 1.05, 1.03, 0.86, 1.04, 1.05, 1.0~
## $ `0D280_0D315_of_diluted_wines` <dbl> 3.92, 3.40, 3.17, 3.45, 2.93, 2.85, 3.5~
## $ Proline                        <dbl> 1065, 1050, 1185, 1480, 735, 1450, 1290~
## $ class                          <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
```

c) Get summary statistics on the numeric variables.

```r
wine %>% select(-class) %>% summary()
```

```
##     Alcohol        Malicacid          Ash        Alcalinity_of_ash
##  Min.   :11.03   Min.   :0.740   Min.   :1.360   Min.   :10.60
##  1st Qu.:12.36   1st Qu.:1.603   1st Qu.:2.210   1st Qu.:17.20
##  Median :13.05   Median :1.865   Median :2.360   Median :19.50
##  Mean   :13.00   Mean   :2.336   Mean   :2.367   Mean   :19.49
##  3rd Qu.:13.68   3rd Qu.:3.083   3rd Qu.:2.558   3rd Qu.:21.50
##  Max.   :14.83   Max.   :5.800   Max.   :3.230   Max.   :30.00
##    Magnesium      Total_phenols     Flavanoids    Nonflavanoid_phenols
##  Min.   : 70.00   Min.   :0.980   Min.   :0.340   Min.   :0.1300
##  1st Qu.: 88.00   1st Qu.:1.742   1st Qu.:1.205   1st Qu.:0.2700
##  Median : 98.00   Median :2.355   Median :2.135   Median :0.3400
##  Mean   : 99.74   Mean   :2.295   Mean   :2.029   Mean   :0.3619
##  3rd Qu.:107.00   3rd Qu.:2.800   3rd Qu.:2.875   3rd Qu.:0.4375
##  Max.   :162.00   Max.   :3.880   Max.   :5.080   Max.   :0.6600
##  Proanthocyanins Color_intensity       Hue         0D280_0D315_of_diluted_wines
```

```
##  Min.   :0.410   Min.   : 1.280   Min.   :0.4800   Min.   :1.270
##  1st Qu.:1.250   1st Qu.: 3.220   1st Qu.:0.7825   1st Qu.:1.938
##  Median :1.555   Median : 4.690   Median :0.9650   Median :2.780
##  Mean   :1.591   Mean   : 5.058   Mean   :0.9574   Mean   :2.612
##  3rd Qu.:1.950   3rd Qu.: 6.200   3rd Qu.:1.1200   3rd Qu.:3.170
##  Max.   :3.580   Max.   :13.000   Max.   :1.7100   Max.   :4.000
##     Proline
##  Min.   : 278.0
##  1st Qu.: 500.5
##  Median : 673.5
##  Mean   : 746.9
##  3rd Qu.: 985.0
##  Max.   :1680.0
```

1.2 Question 2

a) Scale and Center your data *Hint:* Use mutate() statement across all columns **except class** with function(x) as.numeric(scale(x)).

```
wine_scaled <- wine %>%
  mutate(across(-class, ~as.numeric(scale(.))))
```

b) Based on what you saw in the summary statistic table from the imported data, why would scaling and centering this data be helpful before we perform PCA.

Scaling ensures all variables contribute equally. Without scaling, variables with larger ranges like proline could effect the PCA outcome and hide smaller variables like ash that have a smaller range.

1.3 Question 3

a) Perform PCA

```
wine_pca <- prcomp(
  wine_scaled %>%
    select(-class), center = TRUE, scale. = TRUE
)
```

b) How much of the total variance is explained by PC1? PC2? What function do we use to see that information?

```
summary(wine_pca)$importance[2, 1:2]
```

```
##     PC1     PC2
## 0.36199 0.19207
```

36% of the variance is explained by PC1 and 19% of the variance is explained by PC2. We use the summary function to see this information.

c) Why are we doing PCA first?

PCA is done first because it compresses the data into a smaller uncorrelated data frame and it removes the redundancy between correlated variables and reduces noise.

d) What is the rotation matrix? Print it explicitly. *Hint:* Check the notes for a simple way to do this!

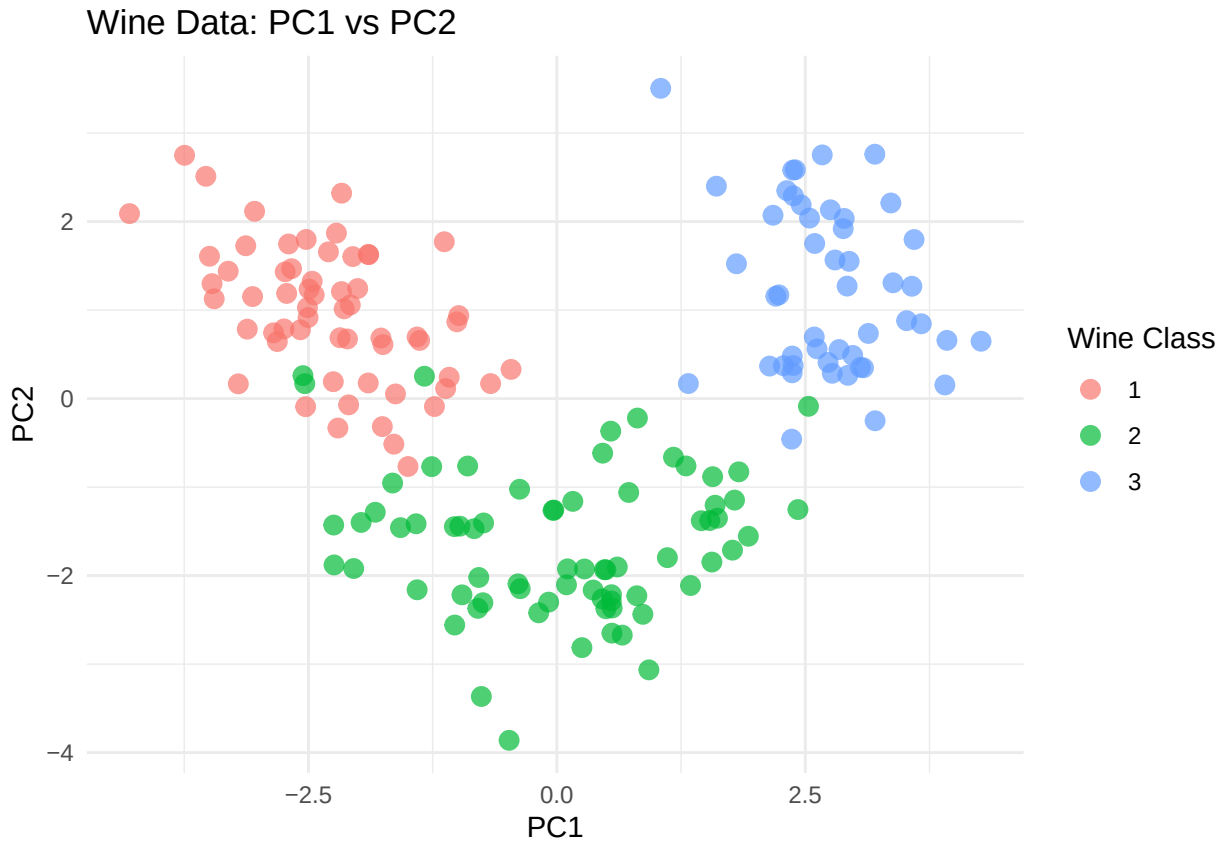
```
print(wine_pca$rotation, digits = 3)
```

```
##                                 PC1      PC2     PC3     PC4     PC5     PC6
## Alcohol                      -0.14433  0.48365 -0.2074  0.0179 -0.2657  0.2135
## Malicacid                     0.24519  0.22493  0.0890 -0.5369  0.0352  0.5368
## Ash                           0.00205  0.31607  0.6262  0.2142 -0.1430  0.1545
## Alcalinity_of_ash             0.23932 -0.01059  0.6121 -0.0609  0.0661 -0.1008
## Magnesium                    -0.14199  0.29963  0.1308  0.3518  0.7270  0.0381
## Total_phenols                -0.39466  0.06504  0.1462 -0.1981 -0.1493 -0.0841
## Flavanoids                   -0.42293 -0.00336  0.1507 -0.1523 -0.1090 -0.0189
## Nonflavanoid_phenols          0.29853  0.02878  0.1704  0.2033 -0.5007 -0.2586
## Proanthocyanins              -0.31343  0.03930  0.1495 -0.3991  0.1369 -0.5338
## Color_intensity               0.08862  0.53000 -0.1373 -0.0659 -0.0764 -0.4186
## Hue                          -0.29671 -0.27924  0.0852  0.4278 -0.1736  0.1060
## OD280_OD315_of_diluted_wines -0.37617 -0.16450  0.1660 -0.1841 -0.1012  0.2659
## Proline                      -0.28675  0.36490 -0.1267  0.2321 -0.1579  0.1197
##                                 PC7     PC8     PC9    PC10    PC11    PC12
## Alcohol                      -0.0564  0.3961 -0.5086  0.2116  0.2259 -0.2663
## Malicacid                     0.4205  0.0658  0.0753 -0.3091 -0.0765  0.1217
## Ash                          -0.1492 -0.1703  0.3077 -0.0271  0.4987 -0.0496
## Alcalinity_of_ash            -0.2870  0.4280 -0.2004  0.0528 -0.4793 -0.0557
## Magnesium                     0.3229 -0.1564 -0.2714  0.0679 -0.0713  0.0622
## Total_phenols                -0.0279 -0.4059 -0.2860 -0.3201 -0.3043 -0.3039
## Flavanoids                   -0.0607 -0.1872 -0.0496 -0.1632  0.0257 -0.0429
## Nonflavanoid_phenols          0.5954 -0.2333 -0.1955  0.2155 -0.1169  0.0424
## Proanthocyanins               0.3721  0.3682  0.2091  0.1342  0.2374 -0.0956
## Color_intensity              -0.2277 -0.0338 -0.0562 -0.2908 -0.0318  0.6042
## Hue                           0.2321  0.4366 -0.0858 -0.5224  0.0482  0.2592
## OD280_OD315_of_diluted_wines -0.0448 -0.0781 -0.1372  0.5237 -0.0464  0.6010
## Proline                       0.0768  0.1200  0.5758  0.1621 -0.5393 -0.0794
##                                 PC13
## Alcohol                       0.0150
## Malicacid                     0.0260
## Ash                          -0.1412
## Alcalinity_of_ash             0.0917
## Magnesium                     0.0568
## Total_phenols                -0.4639
## Flavanoids                    0.8323
## Nonflavanoid_phenols          0.1140
## Proanthocyanins              -0.1169
## Color_intensity              -0.0120
## Hue                          -0.0899
## OD280_OD315_of_diluted_wines -0.1567
## Proline                       0.0144
```

e) Plot PC1 vs. PC2, using the wine class as labels for coloring. *Hint:* You'll first need a data set with only PC1 and PC2, then add back the class variable from your scaled data set with a mutate() statement. Then, you can use color = factor(class) in your ggplot statement.

```r
wine_scores <- as_tibble(wine_pca$x) %>%
  mutate(class = wine_scaled$class)

ggplot(wine_scores, aes(x = PC1, y = PC2, color = factor(class))) +
  geom_point(size = 3, alpha = 0.7) +
  labs(title = "Wine Data: PC1 vs PC2",
       x = "PC1",
       y = "PC2",
       color = "Wine Class") +
  theme_minimal()
```



f) What do you see after plotting PC1 vs PC2? What does this mean in context of wine classes?

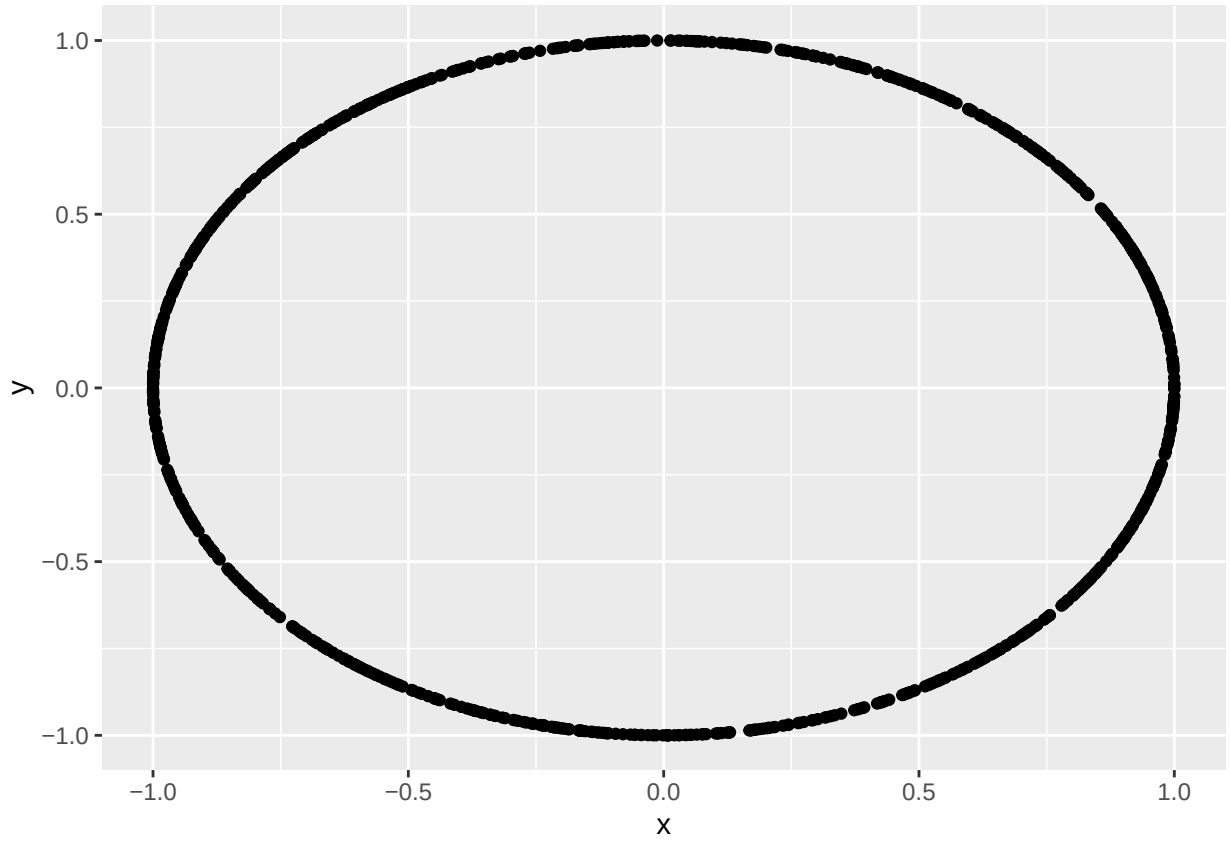
The wine classes form distinct clusters. Wine class 1 is more condensed than 2 and 3, with wine class 2 being the most spread out with a few points overlapping in wine class 1. This means that PC1 and PC@ explain enough of the variance to define class structure which could suggest that wines in different classes are chemically distinct.

g) Give an example of data where PCA would fail. You can describe the data or do a simulation. *Hint:* Our notes have a few examples!

PCA would fail if the data was arranged in a spiral and when the noise has a higher variance than the real pattern.In that example PCA would cause it to overlap, lose its actual structure, and align with the noise instead of the signal

Example:

```r
set.seed(1)
theta <- runif(1000, 0, 2*pi)
x<-cos(theta)
y<-sin(theta)
df<- tibble(x,y)
ggplot(df, aes(x,y)) + geom_point()
```



h) Explain the difference between vector space and manifold, and how these terms apply to what we did/will do with T-SNE.

A vector space is a mathematical structure where vectors are added and scaled by numbers from a field. Its restrictive because every operation has to make sense globally and linearly. PCA works in vector spaces and finds a rotation of the axes and aligns with the directions of maximum variance in the data. A Manifold is a space that can be curved globally but looks like a vector space locally. So, PCA can only work in vector spaces due the the assumption it holds that data can be globally represented as a vector space, which only works if the structure is linear.T-SNE is essentially a manifold method because it does not represent the entire data set with one linear rotation. It it keeps the local environment which unrolls the manifold the data lies on. This is why T-SNE can reveal non-linear structure such as clusters that PCA can't.

1.4 Question 4

a) Perform T-SNE set seed = 123. *Hint:* Subset your PCS results to PC1-PC10, add the class variable back in, remove duplicates, then perform T-SNE.

```r
set.seed(123)

pca_subset <- wine_pca$x[, 1:10] %>%
  as_tibble() %>%
  mutate(class = wine_scaled$class) %>%
  distinct()

tsne_results <- Rtsne(as.matrix(pca_subset %>% select(-class)), dims = 2)

tsne_df<- as_tibble(tsne_results$Y) %>%
  rename(Dim1 = V1, Dim2 = V2) %>%
  mutate(class = pca_subset$class)
```

```
## Warning: The `x` argument of `as_tibble.matrix()` must have unique column names if
## `.name_repair` is omitted as of tibble 2.0.0.
## i Using compatibility `.name_repair`.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

b) Plot the results in 2D. *Hint:* Convert your T-SNE results to a tibble and add back the class variable from your scaled data set using a mutate() statement. Then, you can use color = factor(class) in your ggplot statement.

```r
ggplot(tsne_df, aes(x=Dim1, y=Dim2, color = factor(class))) +
  geom_point(alpha = 0.7) +
  labs(title = "T-SNE Results on Wine Data",
       x = "Dim1",
       y = "Dim2",
       color = "Wine Class"
       )
```

c) Why didn't we stop at PCA?

PCA is a linear dimensionality reduction method that captures axes of maximum variance but assumes global linear structure, but not all data sets contain linear relationships. Meaning that PCA can spread the wine classes in rotated spaces but it can really separate them if the class boundaries are curved. But, T-SNE is non-linear and manifold based which allows the clusters to be created naturally, even if they are curved or non-linear. Its best to perform both to get the full picture.

d) What other types of data does this workflow make sense for?

This workflow is useful for high- dimensional data where local neighborhoods matter more than global variance. Examples of this would be biological data, and clinical data.