MTH 3270 Notes 10

7.1 Unsupervised Learning (Cont'd) (9)

- Recall that ***unsupervised learning*** is when there are **explanatory** variables (X's) but ***no*** **response** variable (Y).

7.1.3 Dimension Reduction: SVD and PCA

- ***Dimension reduction*** refers to reducing the number of variables (columns) of a data set to a smaller number of ones that carry the most information.

  This sometimes facilitates prediction or cluster identification.

- For example, a variable can be discarded if it doesn't distinguish between individuals (e.g. age if everyone is the same age), and redundant ones can also be discarded (e.g. weights of people in kilograms if their weights in pounds are also recorded).

  Consider, as an example, this (hypothetical) data set on eight people:

```
my.data <- data.frame(Age = c(25, 25, 25, 25, 25, 25, 25, 25),
                      WtLb  = c(160, 155, 165, 170, 180, 155, 165, 175),
                      WtKg = c(72.6, 70.3, 74.8, 77.1, 81.6, 70.3, 74.8, 79.4))
my.data
```

```
##   Age WtLb WtKg
## 1  25  160 72.6
## 2  25  155 70.3
## 3  25  165 74.8
## 4  25  170 77.1
## 5  25  180 81.6
## 6  25  155 70.3
## 7  25  165 74.8
## 8  25  175 79.4
```

  We can think of *eliminating* the `Age` and `WtKg` columns as **"deriving"** a new variable `V` via a ***linear combination*** of the columns in `my.data`, with ***weights*** **0**, **1**, and **0**:

$$\texttt{V} \;=\; 0 \times \texttt{Age} \;+\; 1 \times \texttt{WtLb} \;+\; 0 \times \texttt{WtKg},$$

  i.e.

```
my.new.data <- my.data %>%
                 mutate(V = 0*Age + 1*WtLb + 0*WtKg) %>%
                 select(V)
my.new.data
```

```
##     V
## 1 160
## 2 155
## 3 165
## 4 170
## 5 180
## 6 155
## 7 165
## 8 175
```

- In practice, it's rare for variables to be *entirely* uninformative (as `Age` and `WtKg` are above), but usually some are *less* informative than others. We can still **"derive"** new variables via **linear combinations** in this case. The less informative variables in the linear combination get *smaller weights* (but not zero weights).
- ***Singular value decomposition*** (or ***SVD***), a procedure from matrix algebra, is used to **"derive"** new variables $\boldsymbol{V_1, V_2, \ldots, V_p}$, each of which is a **linear combination** of explanatory variables $\boldsymbol{X_1, X_2, \ldots, X_p}$, such that $\boldsymbol{V_1}$ is the most informative, $\boldsymbol{V_2}$ the second most informative, ..., $\boldsymbol{V_p}$ the least informative, and there **aren't redundancies** among the $\boldsymbol{V}$'s (i.e. they carry completely independent information).*

  Dimension reduction is then done by *discarding* all but the first few (most informative) $\boldsymbol{V}$'s. The *retained* $\boldsymbol{V}$'s can then be used as explanatory variables in random forests, in cluster analyses, etc.

  *"Most informative" means its values *vary* the most across individuals (subject to a certain constraint on how large the weights can be – in each V, they're constrained so that the sum of their squares equals one). "Second most informative" means its values *vary* second most, etc.

- ***Principal components analysis*** (or ***PCA***) is **SVD** performed after ***centering*** each of the $\boldsymbol{X}$ variables (by subtracting their means). **PCA** gives **more meaningful** results than SVD.** An example of PCA is given below.

  **Without centering, V_1 would weight heavily X's whose values vary from *zero*. With centering, it weights heavily X's whose values vary away from *their mean*.

- We'll use the function:

```
svd()       # Perform a singular value decomposition (or a principal
            # components analysis) of a set of explanatory variables
            # (columns) from a data frame (stored as a matrix).
```

- As an example of **PCA** (i.e. **SVD** on the **centered variables**), we'll use `my.data` (from above). The means of `Age`, `WtLb`, and `WtKg` are:

```
colMeans(my.data)

##      Age     WtLb     WtKg
##  25.0000 165.6250  75.1125
```

We can **center** the variables (subtract their means) using `scale()`:

```
# Center the variables in my.data:
cntrd.data <- scale(my.data, scale = FALSE) %>% as.data.frame()
cntrd.data

##   Age    WtLb    WtKg
## 1   0  -5.625 -2.5125
## 2   0 -10.625 -4.8125
## 3   0  -0.625 -0.3125
## 4   0   4.375  1.9875
## 5   0  14.375  6.4875
## 6   0 -10.625 -4.8125
## 7   0  -0.625 -0.3125
## 8   0   9.375  4.2875
```

Now we perform **PCA**:

```
# Perform PCA (SVD on the centered variables):
my.svd <- svd(as.matrix(cntrd.data))
```

The object `my.svd` is a *list* containing three things:

```
names(my.svd)

## [1] "d" "u" "v"
```

The **weights** in the optimal **linear combinations** of `Age`, `WtLb`, and `WtKg` that generate $\boldsymbol{V_1}$, $\boldsymbol{V_2}$, and $\boldsymbol{V_3}$ are the columns of `my.svd$v`:

```
my.svd$v

##            [,1]       [,2] [,3]
## [1,] 0.0000000  0.0000000    1
## [2,] 0.9109678  0.4124774    0
## [3,] 0.4124774 -0.9109678    0
```

Thus

$$V_1 \;=\; 0.00 \times \texttt{Age} \;+\; 0.91 \times \texttt{WtLb} \;+\; 0.41 \times \texttt{WtKg}\,, \tag{1}$$

$$V_2 \;=\; 0.00 \times \texttt{Age} \;+\; 0.41 \times \texttt{WtLb} \;-\; 0.91 \times \texttt{WtKg}\,, \tag{2}$$

and

$$V_3 \;=\; 1 \times \texttt{Age} \;+\; 0 \times \texttt{WtLb} \;+\; 0 \times \texttt{WtKg} \tag{3}$$

(where `Age`, `WtLb`, and `WtKg` are their *centered* versions). Note that V_1 is like an "average" of `WtLb` and `WtKg`.

The values the vector `my.svd$d`:

```
my.svd$d

## [1] 26.25108372  0.06598016  0.00000000
```

can be used as measures of the relative amounts of information in each of V_1, V_2, and V_3. Larger values indicate more information. Thus almost all of the information in `my.data` is contained in V_1, and none of it is contained in V_3.

We can obtain *re-scaled versions* of the **"derived"** variables $\boldsymbol{V_1}$, $\boldsymbol{V_2}$, and $\boldsymbol{V_3}$ via `my.svd$u`:***

```
# Re-scaled version of "derived" variables:
as.data.frame(my.svd$u)

##            V1          V2            V3
## 1 -0.23467768 -0.47558005  8.477912e-01
## 2 -0.44432759  0.02212828 -1.105815e-01
## 3 -0.02659906  0.40738085  2.211629e-01
## 4  0.18305085 -0.09032748  3.382017e-14
## 5  0.60077938  0.29492509  3.317444e-01
## 6 -0.44432759  0.02212828 -1.105815e-01
## 7 -0.02659906  0.40738085  2.211629e-01
## 8  0.39270075 -0.58803582 -2.211629e-01
```

The *actual* variables $\boldsymbol{V_1}$, $\boldsymbol{V_2}$, and $\boldsymbol{V_3}$ given by (1)-(3) are obtained by multiplying each column of `my.svd$u` by its `my.svd$d` value. We can do this with `sweep()`:

```
# Multiply each variable (column) of u by its d value:
V <- sweep(x = my.svd$u,
           MARGIN = 2,
           STATS = my.svd$d,
           FUN = "*") %>%
     as.data.frame()
V

##            V1           V2 V3
## 1  -6.1605435 -0.031378850  0
## 2 -11.6640807  0.001460028  0
## 3  -0.6982541  0.026879055  0
```

```
## 4   4.8052831 -0.005959822  0
## 5  15.7711097  0.019459205  0
## 6 -11.6640807  0.001460028  0
## 7  -0.6982541  0.026879055  0
## 8  10.3088203 -0.038798699  0
```

Ordered left to right, `V1` is the most informative, `V2` the second most, and `V3` the least informative.

In this example, we'd *discard* `V2` and `V3` because essentially all of the information in the data is contained in `V1` (based on the `my.svd$d` values). We could then use this one variable, `V1`, for prediction, identifying clusters, etc.

***Sometimes these *re-scaled* V's are plotted and used for prediction, cluster identification, etc. instead of the *actual* V's. This is what the textbook does.

Section 7.1 Exercises

Exercise 1 Consider again the built-in `iris` data set. We'll use just the *Virginica* species:

```
library(dplyr)      # For filter(), select().

virginica <- iris %>%
               filter(Species == "virginica") %>%
               select(-Species)
```

The means of the four variables, `Sepal.Length`, `Sepal.Width`, `Petal.Length`, and `Petal.Width`, are:

```
colMeans(virginica)

## Sepal.Length  Sepal.Width Petal.Length  Petal.Width
##        6.588        2.974        5.552        2.026
```

Center the four variables (by subtracting their means) by typing:

```
cntrd.virginica <- scale(virginica, scale = FALSE) %>% as.data.frame()

head(cntrd.virginica, n = 3)

##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1       -0.288       0.326        0.448       0.474
## 2       -0.788      -0.274       -0.452      -0.126
## 3        0.512       0.026        0.348       0.074
```

Now perform **PCA** (i.e. **SVD** on the **centered** variables):

```
my.svd <- svd(as.matrix(cntrd.virginica))
```

The **weights** in the **"derived"** variables $\boldsymbol{V_1}, \boldsymbol{V_2}, \boldsymbol{V_3}$, and $\boldsymbol{V_4}$ are the columns of `my.svd$v`:

```
my.svd$v
```

```
##           [,1]       [,2]       [,3]       [,4]
## [1,] 0.7410168 -0.1652590  0.5344502  0.3714117
## [2,] 0.2032877  0.7486428  0.3253749 -0.5406841
## [3,] 0.6278918 -0.1694278 -0.6515236 -0.3905934
## [4,] 0.1237745  0.6192880 -0.4289653  0.6458723
```

Thus the **"derived"** variables are:

$$\begin{aligned} V_1 &= 0.74 \times \texttt{Sepal.Length} + 0.20 \times \texttt{Sepal.Width} + 0.63 \times \texttt{Petal.Length} \\ &\quad +0.12 \times \texttt{Petal.Width}, \\ V_2 &= -0.17 \times \texttt{Sepal.Length} + 0.75 \times \texttt{Sepal.Width} - 0.17 \times \texttt{Petal.Length} \\ &\quad +0.62 \times \texttt{Petal.Width}, \\ V_3 &= 0.53 \times \texttt{Sepal.Length} + 0.33 \times \texttt{Sepal.Width} - 0.65 \times \texttt{Petal.Length} \\ &\quad -0.43 \times \texttt{Petal.Width}, \end{aligned}$$

and

$$\begin{aligned} V_4 &= 0.37 \times \texttt{Sepal.Length} - 0.54 \times \texttt{Sepal.Width} - 0.39 \times \texttt{Petal.Length} \\ &\quad +0.65 \times \texttt{Petal.Width} \end{aligned}$$

(where `Sepal.Length`, `Sepal.Width`, `Petal.Length`, and `Petal.Width` are their *centered* versions).

Of these, $\boldsymbol{V_1}$ is the most informative, $\boldsymbol{V_2}$ the second most informative, etc. For **dimension reduction**, we'd probably *discard* $\boldsymbol{V_3}$ and $\boldsymbol{V_4}$.

The **weights** reflect a variable's *contribution* to the **linear combination** (larger weight = larger contribution).

Look at the **weights** in $\boldsymbol{V_1}$. Which of the four variables (`Sepal.Length`, `Sepal.Width`, `Petal.Length`, or `Petal.Width`) contributes the *most* to $\boldsymbol{V_1}$? Which contributes the *least*?