Stat 787 - Assignment 2

Zach Gagnon

2023-02-12

Exercise 1

The following R chunk loads the data for the closing prices of nine major tech companies over the last ten years, and plots them as time series.

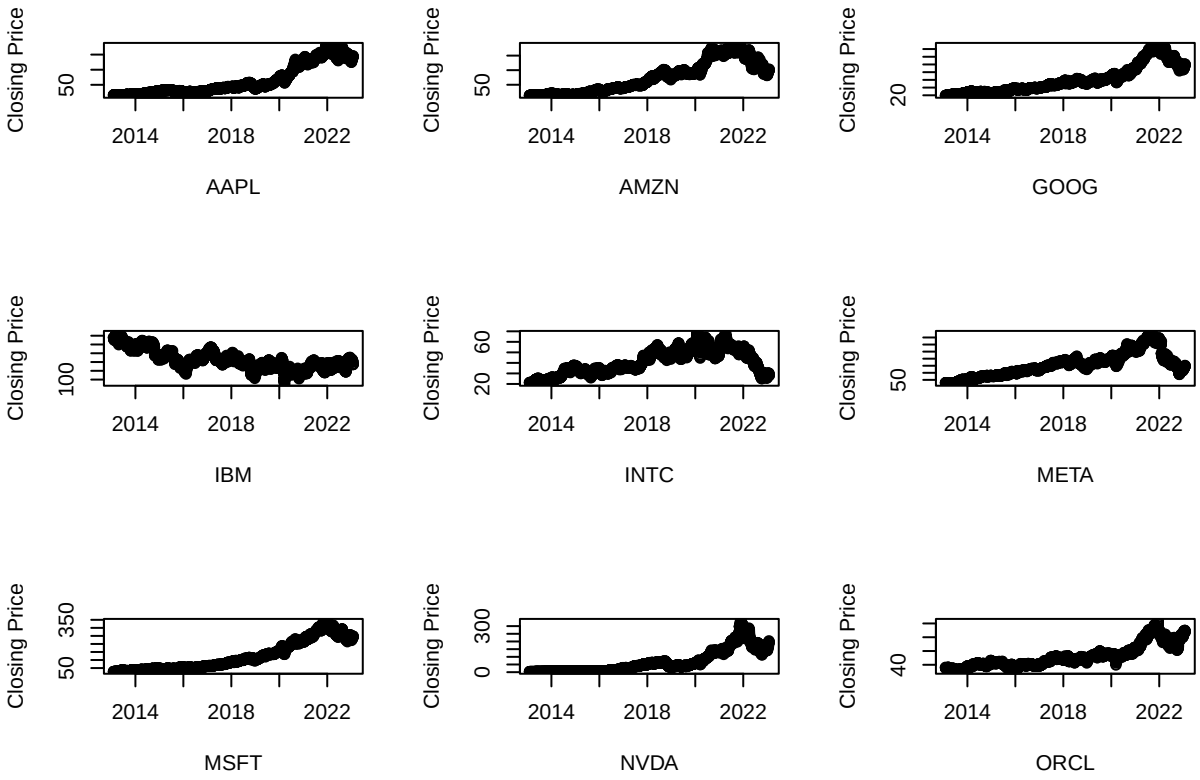
```
setwd("C:/Users/16317/Documents/Spring 2023/STAT 787/Assignment 2")

close <- read.csv("ClosingPrices.csv")
head(close)
```

```
##       Date     AAPL    AMZN     GOOG      IBM  INTC  META  MSFT   NVDA  ORCL
## 1 2/1/2013 16.20071 13.2500 19.31759 196.1568 21.36 29.73 27.93 3.0925 36.21
## 2 2/4/2013 15.79714 12.9990 18.90464 194.8279 21.16 28.11 27.44 3.0400 35.13
## 3 2/5/2013 16.35143 13.3445 19.07201 193.8719 21.18 28.64 27.50 3.1100 35.48
## 4 2/6/2013 16.33393 13.1110 19.18235 192.1797 20.99 29.05 27.34 3.0850 35.10
## 5 2/7/2013 16.72214 13.0115 19.27650 190.9560 20.81 28.65 27.28 3.0725 34.56
## 6 2/8/2013 16.96357 13.0975 19.56093 192.8107 21.00 28.55 27.55 3.0925 34.90
```

```
close$Date <-  as.Date(close$Date, "%m/%d/%Y")

par(mfrow=c(3,3))
plot(close$AAPL ~ close$Date, xlab="AAPL", ylab="Closing Price")
plot(close$AMZN ~ close$Date, xlab="AMZN", ylab="Closing Price")
plot(close$GOOG ~ close$Date, xlab="GOOG", ylab="Closing Price")
plot(close$IBM ~ close$Date, xlab="IBM", ylab="Closing Price")
plot(close$INTC ~ close$Date, xlab="INTC", ylab="Closing Price")
plot(close$META ~ close$Date, xlab="META", ylab="Closing Price")
plot(close$MSFT ~ close$Date, xlab="MSFT", ylab="Closing Price")
plot(close$NVDA ~ close$Date, xlab="NVDA", ylab="Closing Price")
plot(close$ORCL ~ close$Date, xlab="ORCL", ylab="Closing Price")
```



The same plotting will be done again, but using ggplot.

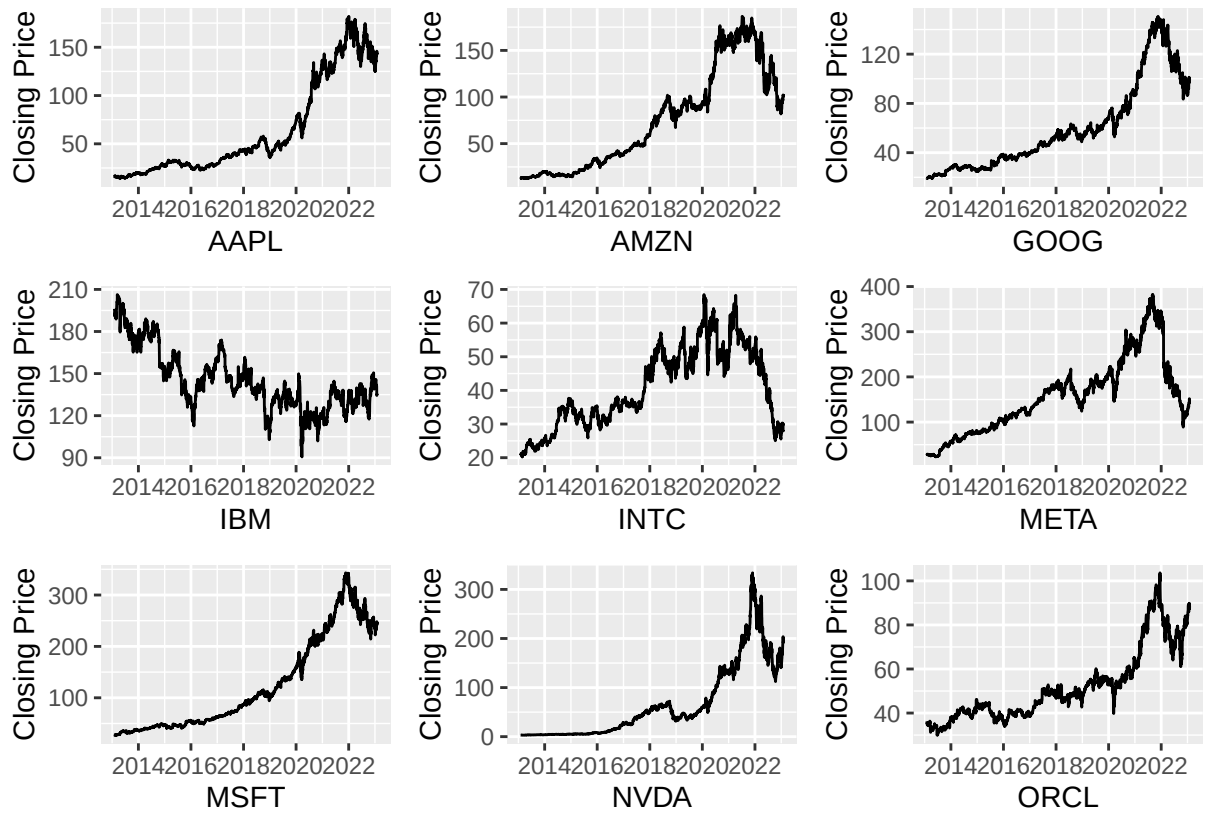
```
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

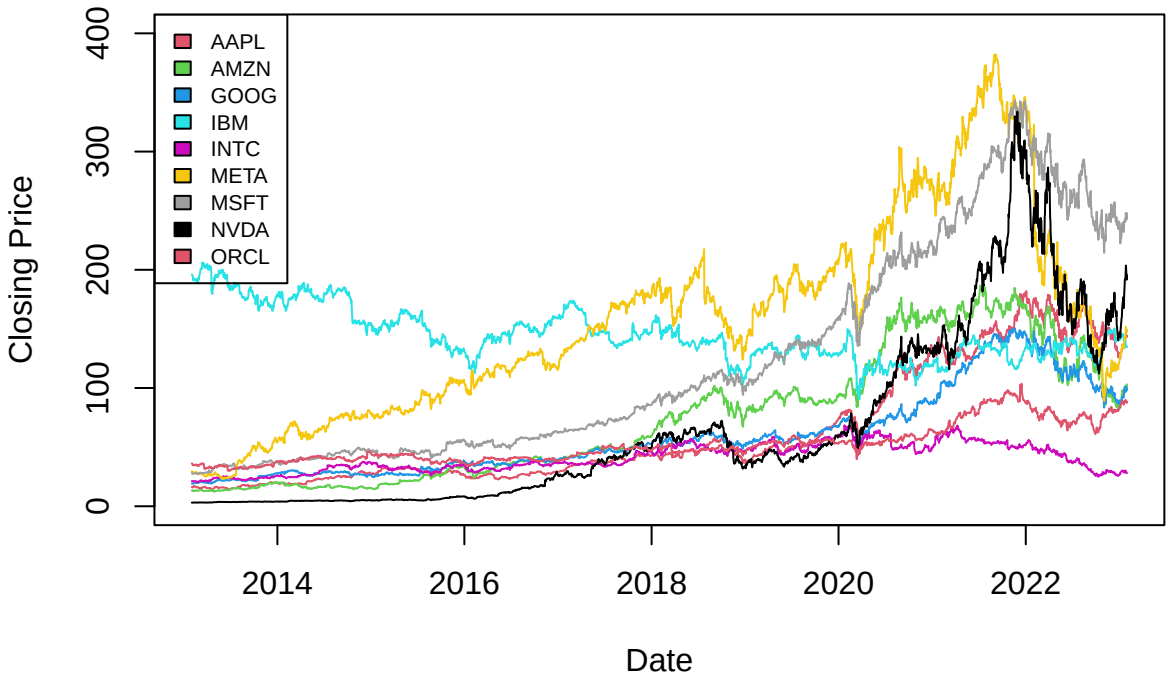
```
p1 <- ggplot(close, aes(x=Date, y=AAPL)) + geom_line() + xlab("AAPL") + ylab("Closing Price")
p2 <- ggplot(close, aes(x=Date, y=AMZN)) + geom_line() + xlab("AMZN") + ylab("Closing Price")
p3 <- ggplot(close, aes(x=Date, y=GOOG)) + geom_line() + xlab("GOOG") + ylab("Closing Price")
p4 <- ggplot(close, aes(x=Date, y=IBM)) + geom_line() + xlab("IBM") + ylab("Closing Price")
p5 <- ggplot(close, aes(x=Date, y=INTC)) + geom_line() + xlab("INTC") + ylab("Closing Price")
p6 <- ggplot(close, aes(x=Date, y=META)) + geom_line() + xlab("META") + ylab("Closing Price")
p7 <- ggplot(close, aes(x=Date, y=MSFT)) + geom_line() + xlab("MSFT") + ylab("Closing Price")
p8 <- ggplot(close, aes(x=Date, y=NVDA)) + geom_line() + xlab("NVDA") + ylab("Closing Price")
p9 <- ggplot(close, aes(x=Date, y=ORCL)) + geom_line() + xlab("ORCL") + ylab("Closing Price")
library(patchwork)
p1 + p2 + p3 + p4 + p5 + p6 + p7 + p8 + p9
```



We can see from both plots above that Apple, Amazon, Google, Meta, Microsoft, Nividia, and Oracle all have similar trends. From 2013 until 2022 we see an exponential increase in closing price, but then a decrease in 2023. Note that we see a big jump after 2020 with this companies, probably due to the Covid-19 pandemic. As for IBM, it appears their stock has decreased somewhat steadily over the last decade. Intel's stock increases in what appears to be a linear trend over from 2013 to 2021, and then took a major dip over the last two years.

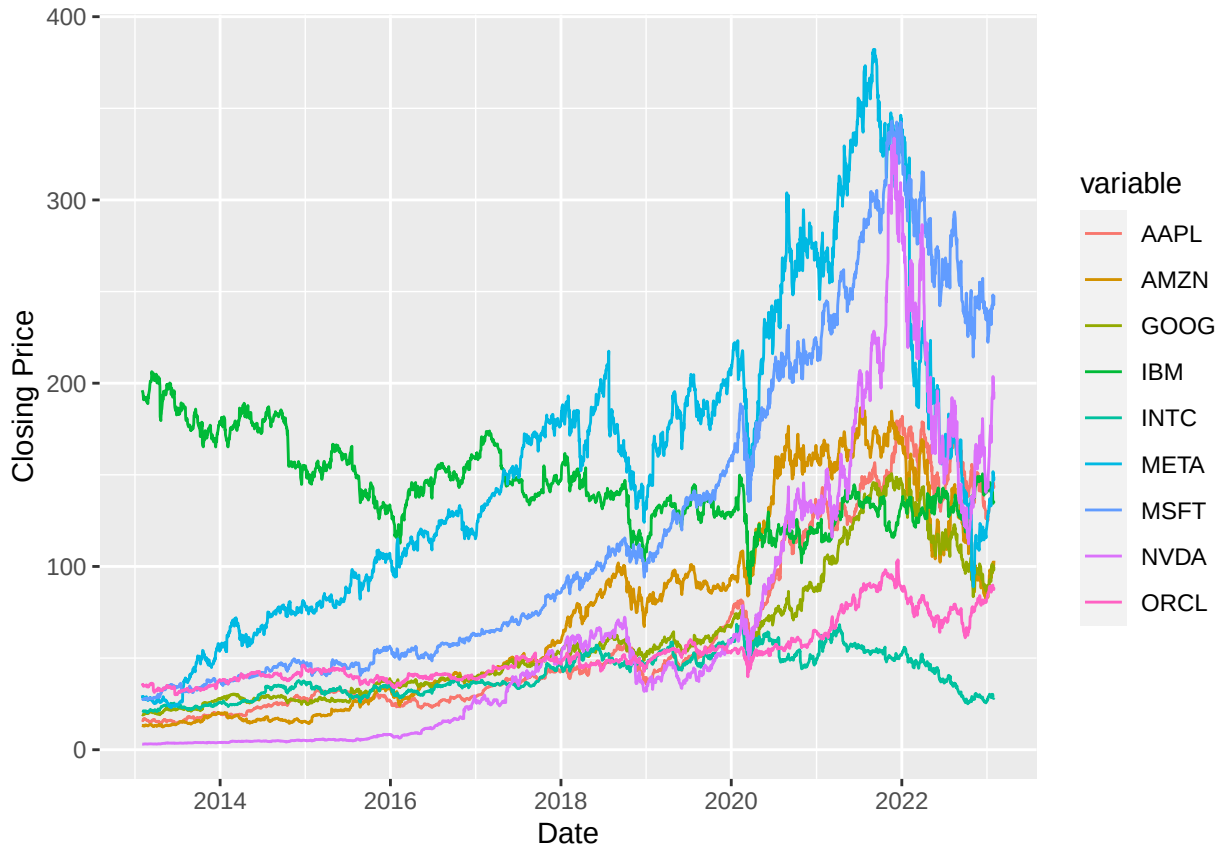
The following R chunks plot the time series seen above on one plot. The first chunk does so with traditional plotting and the second with ggplot.

```
par(mfrow=c(1,1))
plot(close$AAPL ~ close$Date, type="l", col=2, xlab="Date", ylab="Closing Price",
     ylim=c(0,400))
lines(close$AMZN ~ close$Date, type="l", col=3)
lines(close$GOOG ~ close$Date, type="l", col=4)
lines(close$IBM ~ close$Date, type="l", col=5)
lines(close$INTC ~ close$Date, type="l", col=6)
lines(close$META ~ close$Date, type="l", col=7)
lines(close$MSFT ~ close$Date, type="l", col=8)
lines(close$NVDA ~ close$Date, type="l", col=9)
lines(close$ORCL ~ close$Date, type="l", col=10)
legend("topleft", c("AAPL", "AMZN", "GOOG", "IBM", "INTC", "META", "MSFT", "NVDA", "ORCL"),
       cex=0.70, fill=2:10)
```




```
library(reshape2)
meltdf <- melt(close, id="Date")

ggplot(meltdf, aes(x=Date,  y=value, colour=variable, group=variable)) +
  geom_line() + ylab("Closing Price")
```



Exercise 2

Using the R data set mtcars, ten linear learners were built using bootstrap samples of the data.

```
data(mtcars)
mtcars
```

```
##                    mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360        14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D         24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230          22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280          19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## Merc 280C         17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## Merc 450SE        16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL        17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## Merc 450SLC       15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128          32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic       30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
```

```
## Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

```
L <- 10
n <- nrow(mtcars)
models <- list()

for(i in 1:L)
{
  boot <- mtcars[sample(n, replace=T),]
  learner <- lm(mpg ~ ., data=boot)
  models[[i]] <- learner
}
models
```

```
## [[1]]
##
## Call:
## lm(formula = mpg ~ ., data = boot)
##
## Coefficients:
## (Intercept)          cyl         disp           hp         drat           wt
##    28.36778     -0.11827      0.02308     -0.09836     -0.11799     -5.45729
##        qsec           vs           am         gear         carb
##     1.01524     -5.74371      5.39885     -0.50020      1.94158
##
##
## [[2]]
##
## Call:
## lm(formula = mpg ~ ., data = boot)
##
## Coefficients:
## (Intercept)          cyl         disp           hp         drat           wt
##    -5.48398      0.02467      0.02146     -0.01882      0.25119     -5.80589
##        qsec           vs           am         gear         carb
##     1.98149     -1.25406      1.43569      1.44264      0.29871
##
##
## [[3]]
##
## Call:
## lm(formula = mpg ~ ., data = boot)
##
```

```
## Coefficients:
## (Intercept)          cyl         disp           hp         drat           wt
##    0.699918     0.373926    -0.018502    -0.009424     3.488044     1.371262
##        qsec           vs           am         gear         carb
##   -0.057662    -2.593914     2.517439     4.410051    -3.043890
##
##
## [[4]]
##
## Call:
## lm(formula = mpg ~ ., data = boot)
##
## Coefficients:
## (Intercept)          cyl         disp           hp         drat           wt
##   -12.98905      1.02307     -0.02097      0.06594      2.77867     -0.12766
##        qsec           vs           am         gear         carb
##     0.64367      4.25841      8.32932      1.08308     -2.59642
##
##
## [[5]]
##
## Call:
## lm(formula = mpg ~ ., data = boot)
##
## Coefficients:
## (Intercept)          cyl         disp           hp         drat           wt
##  -24.236069    -1.102674     0.024870     0.007605     2.159339    -6.337379
##        qsec           vs           am         gear         carb
##    3.018243    -4.338715    -0.039774     1.128686     0.352654
##
##
## [[6]]
##
## Call:
## lm(formula = mpg ~ ., data = boot)
##
## Coefficients:
## (Intercept)          cyl         disp           hp         drat           wt
##   -71.57629     -7.17685      0.03272      0.07591      6.39133     -7.90751
##        qsec           vs           am         gear         carb
##     6.13040    -22.80519    -18.97073      9.16280     -1.09351
##
##
## [[7]]
##
## Call:
## lm(formula = mpg ~ ., data = boot)
##
## Coefficients:
## (Intercept)          cyl         disp           hp         drat           wt
##   -39.19941      0.11564      0.05322     -0.02111      0.60596     -8.94001
##        qsec           vs           am         gear         carb
##     3.66038     -2.77458      3.25002      2.10027      0.89610
##
```

```
##
## [[8]]
##
## Call:
## lm(formula = mpg ~ ., data = boot)
##
## Coefficients:
## (Intercept)          cyl         disp           hp         drat           wt
##     2.28531     -0.38112      0.06792     -0.05329     -0.42498     -9.16091
##        qsec           vs           am         gear         carb
##     1.46698      0.89313      1.11110      3.58466      0.85864
##
##
## [[9]]
##
## Call:
## lm(formula = mpg ~ ., data = boot)
##
## Coefficients:
## (Intercept)          cyl         disp           hp         drat           wt
##     3.69196     -0.19277      0.01695     -0.01848     -1.34984     -4.98286
##        qsec           vs           am         gear         carb
##     1.58979     -2.09404      0.56309      2.74323     -0.24973
##
##
## [[10]]
##
## Call:
## lm(formula = mpg ~ ., data = boot)
##
## Coefficients:
## (Intercept)          cyl         disp           hp         drat           wt
##  10.3733487   -0.7080426    0.0290937   -0.0435802   -0.9002313   -5.6076349
##        qsec           vs           am         gear         carb
##   1.7485188   -2.9302333    4.2258274    0.0004737    1.2306109
```

Then, a matrix is created to store the coefficients and MSE of each model. From this matrix, the "average" model is calculated along with its MSE.

```
models.mat <- matrix(NA, nrow=L, ncol=12)
for(i in 1:L)
{
  models.mat[i,] <- c(models[[i]]$coefficients, mean((models[[i]]$residuals)^2))
}

fhat.avg <- c()
for(i in 1:12)
{
  avg <- mean(models.mat[,i])
  fhat.avg <- c(fhat.avg, avg)
}

singular <- lm(mpg ~ ., data=mtcars)
compare <- matrix(NA, nrow=2, ncol=12)
compare[1,] <- fhat.avg
```

```
compare[2,] <- c(singular$coefficients, mean((singular$residuals)^2))
compare
```

```
##           [,1]       [,2]       [,3]        [,4]     [,5]      [,6]      [,7]
## [1,] -10.80665 -0.8142412 0.02298455 -0.01136004 1.288149 -5.295588 2.1197059
## [2,]  12.30337 -0.1114405 0.01333524 -0.02148212 0.787111 -3.715304 0.8210407
##            [,8]      [,9]    [,10]      [,11]    [,12]
## [1,] -3.9382897 0.7820825 2.515569 -0.1405254 3.098620
## [2,]  0.3177628 2.5202269 0.655413 -0.1994193 4.609201
```

The first row of the matrix compare seen above contains the coefficients and MSE of the average model, while the second row contains that of the singular linear model. Comparing the two, we can see that most of the coefficient are quite similar, but the MSE of the average model shows that it performs better than the singular model.

Exercise 3

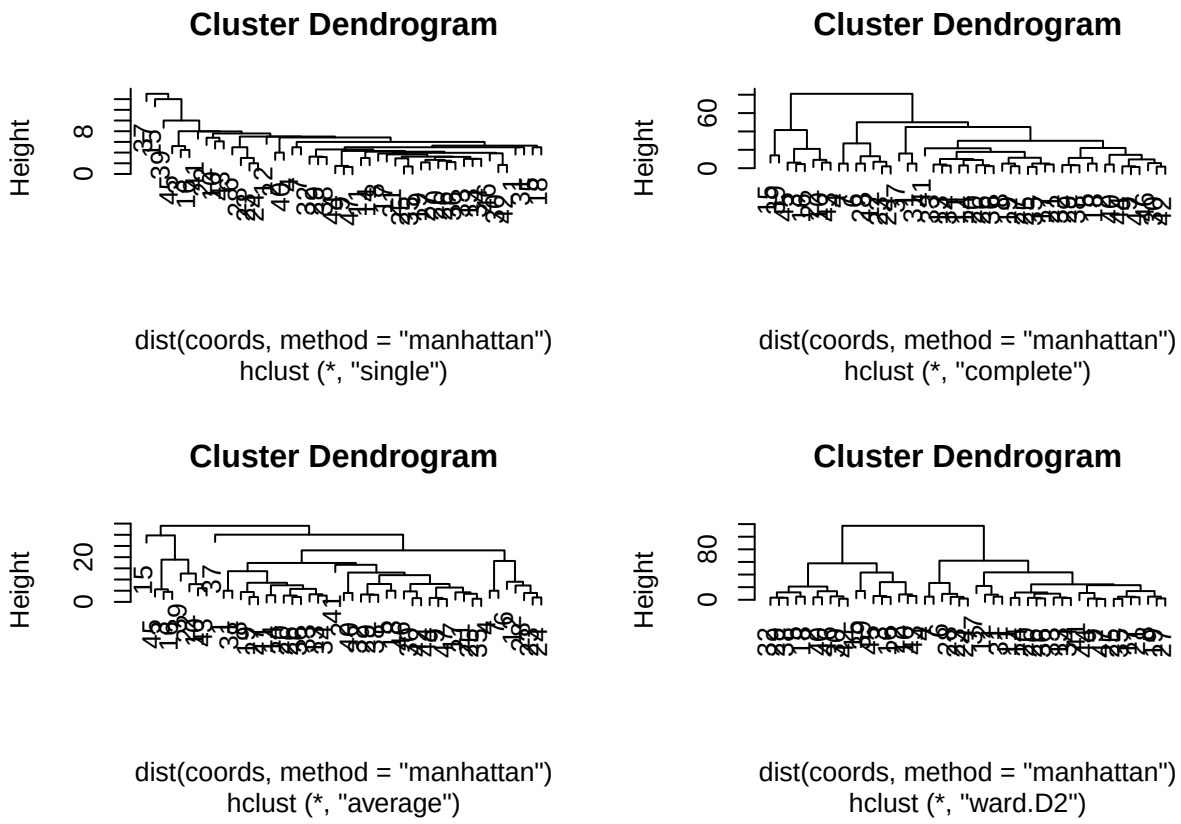
The following data set contains the coordinates of the countries of Europe. From this data, dendrograms are plotted of four different hierarchical clustering of the manhattan distances between countries. Single, double, average, and Ward D2 clustering were used.

```
europe <- read.csv("europe.csv")
head(europe)
```

```
##   zoom    name abbreviation ISO.alpha.2 ISO.alpha.3 ISO.numeric land.area.km
## 1    3 Ukraine         Ukr.          UA         UKR         804       603700
## 2    3  France          Fr.          FR         FRA         250       547030
## 3    3   Spain        Spain          ES         ESP         724       504782
## 4    3  Sweden         Swe.          SE         SWE         752       449964
## 5    3 Germany         Ger.          DE         DEU         276       357021
## 6    3 Finland         Fin.          FI         FIN         246       337030
##   population latitude longitude continent
## 1   45415596     49.0      32.0        eu
## 2   64768389     46.0       2.0        eu
## 3   46505963     40.0      -4.0        eu
## 4    9045000     62.0      15.0        eu
## 5   82369000     51.5      10.5        eu
## 6    5244000     64.0      26.0        eu
```

```
coords <- cbind(europe$latitude, europe$longitude)
countries <- europe$name
hc.single <- hclust(dist(coords, method="manhattan"), method="single")
hc.double <- hclust(dist(coords, method="manhattan"), method="complete")
hc.avg <- hclust(dist(coords, method="manhattan"), method="average")
hc.WD2 <- hclust(dist(coords, method="manhattan"), method="ward.D2")

par(mfrow=c(2,2))
plot(hc.single)
plot(hc.double)
plot(hc.avg)
plot(hc.WD2)
```



The dendrograms were recreated using ggplot.

```
library(dendextend)
```

```
##
## ---------------------
## Welcome to dendextend version 1.16.0
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
##
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issues
## You may ask questions at stackoverflow, use the r and dendextend tags:
##   https://stackoverflow.com/questions/tagged/dendextend
##
##  To suppress this message use:  suppressPackageStartupMessages(library(dendextend))
## ---------------------
```

```
##
## Attaching package: 'dendextend'
```

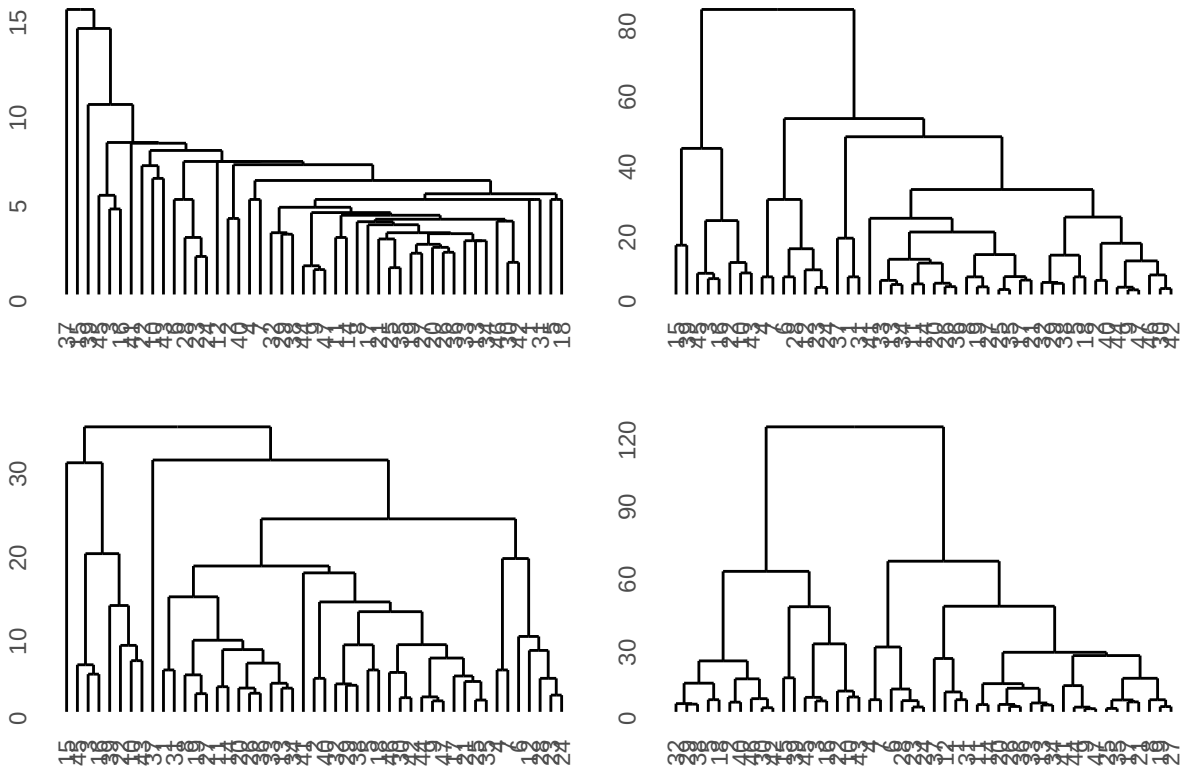
```
## The following object is masked from 'package:stats':
##
##     cutree
```

```
#install.packages("ggdendro")
library(ggdendro)
```

```
##
## Attaching package: 'ggdendro'
```

```
## The following object is masked from 'package:dendextend':
##
##     theme_dendro
```

```
dend.single <- as.dendrogram(hc.single)
d1 <- ggdendrogram(dend.single)
dend.double <- as.dendrogram(hc.double)
d2 <- ggdendrogram(dend.double)
dend.avg <- as.dendrogram(hc.avg)
d3 <- ggdendrogram(dend.avg)
dend.WD2 <- as.dendrogram(hc.WD2)
d4 <- ggdendrogram(dend.WD2)
d1 + d2 + d3 + d4
```



I was able to find the packages igraph and graph, but I was not able to figure out how to create a "graphplot" with clustering of group size 3. I did a lot of searching and used chat gpt, but unfortunately everything I tried and chat gpt coded returned an error. The libraries are commented in the R chunk below.

```
#install.packages("igraph")
#library(igraph)
#if (!require("BiocManager", quietly = TRUE))
#  install.packages("BiocManager")
#BiocManager::install("graph")
#library(graph)
```