# Hw_4

Keshav Ramesh

```r
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.1
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 becom
```

```r
library(httr)
library(jsonlite)
```

```
Attaching package: 'jsonlite'

The following object is masked from 'package:purrr':

    flatten
```

## Task 1 Conceptual Questions

### Question 1

lapplly is a functino that applies a function to each element of a list and also returns a list. The purrr equivilent function is map(), which also applies a function to each element of a list and returns a list.

## Question 2

```r
lapply(my_list, cor, method = "kendall")
```

## Question 3

There is more stability from purrr as it enforeces certain types to return which gives greater consistance between functions. And there is also no consistent way to pass arguments onto the mapper function with the apply() family of functions.

## Question 4

A side effect function performs and action rather than returning a value.

## Question 5

R uses lexical scoping, the sd variable will only be valid within the context of the function, as long as the sd() global function is not called within the function there should be no conflicts.

## Task 2: Writing R functions

### Question 1

```r
getRMSE = function(responses, predicted, ...) {
  sqrt(mean((responses - predicted)^2, ...))
}
```

### Question 2

```r
set.seed(10)
n <- 100
x <- runif(n)
resp <- 3 + 10*x + rnorm(n)
pred <- predict(lm(resp ~ x), data.frame(x))

print(resp)
```

```
  [1]  7.674144  5.733128  8.637031 12.068788  4.357179  6.040709  4.843093
  [8]  6.255948  8.512399  7.587703  8.278962  8.221201  3.304767  9.299369
 [15]  7.646876  8.504220  4.254724  5.160568  7.550652 10.115022 12.028134
 [22]  7.723097  9.702653  6.337183  5.568563 11.239175  9.903050  4.965503
 [29]  9.656077  8.081564  8.948798  3.708220  5.410925 12.714925  7.666618
 [36] 10.636295 11.886290 14.767056  8.670500  7.931076  5.338484  5.097557
 [43]  3.213884 11.444994  6.093762  3.192188  1.563749  8.753929  4.177170
 [50] 12.242498  5.781476 12.783701  4.418721  8.442989  4.282396  9.395394
 [57]  8.255719  6.016290  8.026494  9.180810  2.038727  5.273544  7.225220
 [64]  6.654107 12.260485 10.688362  9.773488  8.216967  5.093565  6.142304
 [71]  3.274337  8.547150  9.381826  7.061813  4.016495  7.543794  6.976389
 [78] 11.550401  5.209433  3.872522 13.043037  8.277356  3.231859  8.553664
 [85]  4.576422  2.213665 11.475262  6.469006  5.333390  5.656304  6.209727
 [92]  8.908905  6.956097  9.642321  7.188749 12.413663  6.020730  8.507994
 [99] 11.776177  3.387353
```

```r
getRMSE(resp, pred)
```

```
[1] 0.9581677
```

```r
resp_missing = resp
resp_missing[c(2, 3)] = NA_real_

resp_missing
```

```
  [1]  7.674144        NA        NA 12.068788  4.357179  6.040709  4.843093
  [8]  6.255948  8.512399  7.587703  8.278962  8.221201  3.304767  9.299369
 [15]  7.646876  8.504220  4.254724  5.160568  7.550652 10.115022 12.028134
 [22]  7.723097  9.702653  6.337183  5.568563 11.239175  9.903050  4.965503
 [29]  9.656077  8.081564  8.948798  3.708220  5.410925 12.714925  7.666618
 [36] 10.636295 11.886290 14.767056  8.670500  7.931076  5.338484  5.097557
 [43]  3.213884 11.444994  6.093762  3.192188  1.563749  8.753929  4.177170
 [50] 12.242498  5.781476 12.783701  4.418721  8.442989  4.282396  9.395394
 [57]  8.255719  6.016290  8.026494  9.180810  2.038727  5.273544  7.225220
 [64]  6.654107 12.260485 10.688362  9.773488  8.216967  5.093565  6.142304
 [71]  3.274337  8.547150  9.381826  7.061813  4.016495  7.543794  6.976389
 [78] 11.550401  5.209433  3.872522 13.043037  8.277356  3.231859  8.553664
 [85]  4.576422  2.213665 11.475262  6.469006  5.333390  5.656304  6.209727
 [92]  8.908905  6.956097  9.642321  7.188749 12.413663  6.020730  8.507994
 [99] 11.776177  3.387353
```

```r
#test with missing 2 values
getRMSE(resp_missing, pred)
```

```
[1] NA
```

```r
# test with na.rm
getRMSE(resp_missing, pred, na.rm = TRUE)
```

```
[1] 0.9579819
```

### Question 3

```r
getMAE <- function(actual, predicted, ...) {
  mean(abs(actual - predicted), ...)
}
```

**Question 4**

```
set.seed(10)
n <- 100
x <- runif(n)
resp <- 3 + 10 * x + rnorm(n)
pred <- predict(lm(resp ~ x), data.frame(x = x))
```

```
# test MAE function
getMAE(resp, pred)
```

```
[1] 0.8155776
```

```
## add 2 missing values
resp_missing = resp
resp_missing[c(3, 4)] = NA_real_

resp_missing
```

```
  [1]  7.674144  5.733128        NA        NA  4.357179  6.040709  4.843093
  [8]  6.255948  8.512399  7.587703  8.278962  8.221201  3.304767  9.299369
 [15]  7.646876  8.504220  4.254724  5.160568  7.550652 10.115022 12.028134
 [22]  7.723097  9.702653  6.337183  5.568563 11.239175  9.903050  4.965503
 [29]  9.656077  8.081564  8.948798  3.708220  5.410925 12.714925  7.666618
 [36] 10.636295 11.886290 14.767056  8.670500  7.931076  5.338484  5.097557
 [43]  3.213884 11.444994  6.093762  3.192188  1.563749  8.753929  4.177170
 [50] 12.242498  5.781476 12.783701  4.418721  8.442989  4.282396  9.395394
 [57]  8.255719  6.016290  8.026494  9.180810  2.038727  5.273544  7.225220
 [64]  6.654107 12.260485 10.688362  9.773488  8.216967  5.093565  6.142304
 [71]  3.274337  8.547150  9.381826  7.061813  4.016495  7.543794  6.976389
 [78] 11.550401  5.209433  3.872522 13.043037  8.277356  3.231859  8.553664
 [85]  4.576422  2.213665 11.475262  6.469006  5.333390  5.656304  6.209727
 [92]  8.908905  6.956097  9.642321  7.188749 12.413663  6.020730  8.507994
 [99] 11.776177  3.387353
```

```
## test with 2 missing values
getMAE(resp_missing, pred)
```

```
[1] NA
```

```r
## test with na,rm
getRMSE(resp_missing, pred, na.rm = TRUE)
```

```
[1] 0.9373585
```

### Question 5

```r
getMetrics = function(response, predicted, metric = c("RMSE", "MAE"), ...) {
  ## Check inputs
  if(is.atomic(response) && is.atomic(predicted) &&
     is.numeric(response) && is.numeric(predicted) &&
     is.vector(response) && is.numeric(predicted)) {

  } else {
    return("Both inputs must be numeric (atomic) vectors")
  }

  result = list()

  if("RMSE" %in% metric) {
    result$RMSE = getRMSE(response, predicted, ...)
  }

  if("MAE" %in% metric) {
    result$MAE = getMAE(response, predicted, ...)
  }


  return(result)

}
```

### Question 6

```r
set.seed(10)
n <- 100
x <- runif(n)
resp <- 3 + 10 * x + rnorm(n)
pred <- predict(lm(resp ~ x), data.frame(x = x))
```

```r
## One of each Metric
getMetrics(resp, pred, metric = "RMSE")
```

```
$RMSE
[1] 0.9581677
```

```r
getMetrics(resp, pred, metric = "MAE")
```

```
$MAE
[1] 0.8155776
```

```r
#Both metrics
getMetrics(resp, pred, metric = c("RMSE", "MAE"))
```

```
$RMSE
[1] 0.9581677

$MAE
[1] 0.8155776
```

```r
#Test with missing values
getMetrics(resp_missing, pred, metric = "RMSE")
```

```
$RMSE
[1] NA
```

```r
#Test with missing value accounted for
getMetrics(resp_missing, pred, metric = "RMSE", na.rm = T)
```

```
$RMSE
[1] 0.9373585
```

```r
getMetrics(as.data.frame(resp), pred)
```

```
[1] "Both inputs must be numeric (atomic) vectors"
```

```r
getMetrics(resp, as.character(pred))
```

```
[1] "Both inputs must be numeric (atomic) vectors"
```

## Task 3

### Question 1

```r
api_key = "a0e72eb552244c0989ffa388cc1285c3"

url = "https://newsapi.org/v2/everything"

news_stories = httr::GET(url, query = list(
  q = "Iran",
  from = "2025-06-15",
  sortBy = "popularity",
  apiKey = api_key
))
```

### Question 2

```r
news_stories_parsed = fromJSON(rawToChar(news_stories$content))

articles = as_tibble(news_stories_parsed$articles)

articles
```

```
# A tibble: 100 x 8
   source$id $name author title description url   urlToImage publishedAt content
   <chr>     <chr> <chr>  <chr> <chr>       <chr> <chr>      <chr>       <chr>
 1 wired     Wired Matt ~ Iran~ "Iran is l~ http~ "https://~ 2025-06-18~ "Alima~
 2 wired     Wired Andre~ Trut~ "The socia~ http~ "https://~ 2025-06-22~ "Truth~
 3 wired     Wired Lily ~ Isra~ "Plus: Ukr~ http~ ""         2025-06-21~ "Amid ~
 4 wired     Wired Molly~ Inte~ ""Violence~ http~ "https://~ 2025-06-23~ "The i~
 5 wired     Wired Steve~ What~ "Meta CTO ~ http~ "https://~ 2025-06-20~ "When ~
 6 the-verge The ~ Tina ~ Iran~ "In a purp~ http~ "https://~ 2025-06-17~ "The g~
 7 <NA>      Gizm~ Luc O~ Trum~ "The U.S. ~ http~ "https://~ 2025-06-22~ "It wa~
 8 <NA>      Gizm~ Matt ~ Sili~ "The tech ~ http~ "https://~ 2025-06-18~ "The U~
 9 <NA>      BBC ~ <NA>   Isra~ "Israel's ~ http~ "https://~ 2025-06-16~ "Jonat~
10 <NA>      BBC ~ <NA>   Wher~ "As the mi~ http~ "https://~ 2025-06-16~ "On Fr~
# i 90 more rows
```

**Question 3**

```
query_news = function(query, date, api_key) {

  news_stories = httr::GET(url, query = list(
  q = query,
  from = date,
  sortBy = "popularity",
  apiKey = api_key
  ))

  news_stories_parsed = fromJSON(rawToChar(news_stories$content))


  if(!is.null(news_stories_parsed$articles)){
    return(as_tibble(news_stories_parsed$articles))
  } else{
   return("No Articles Found")
  }

}
```

```
gamestop_news = query_news("gamestop", "2025-06-01", api_key)

gamestop_news
```

```
# A tibble: 100 x 8
   source$id $name author title description url   urlToImage publishedAt content
   <chr>     <chr> <chr>  <chr> <chr>       <chr> <chr>      <chr>       <chr>
 1 the-verge The ~ David~ A ni~ "I'm stand~ http~ https://p~ 2025-06-05~ "Body ~
 2 the-verge The ~ Brand~ The ~ "Amazon’s ~ http~ https://p~ 2025-06-20~ "Amazo~
 3 <NA>      Gizm~ Kyle ~ Targ~ "Check to ~ http~ https://g~ 2025-06-03~ "The S~
 4 <NA>      Gizm~ James~ Did ~ "Maybe ord~ http~ https://g~ 2025-06-05~ "When ~
 5 <NA>      Gizm~ James~ Some~ "There's o~ http~ https://g~ 2025-06-18~ "The S~
 6 <NA>      Gizm~ Kyle ~ Nint~ "The Switc~ http~ https://g~ 2025-06-20~ "After~
 7 <NA>      Andr~ Nicho~ Moto~ "The Motor~ http~ https://c~ 2025-06-05~ "Why y~
 8 <NA>      Slas~ msmash Game~ "GameStop ~ http~ https://a~ 2025-06-13~ "Cohen~
 9 <NA>      Slas~ Edito~ Nint~ "TweakTown~ http~ https://a~ 2025-06-07~ "Tweak~
10 <NA>      Kota~ Ethan~ Stat~ "Imagine y~ http~ https://i~ 2025-06-05~ "Imagi~
# i 90 more rows
```