THE UNIVERSITY OF AUCKLAND

TERM TEST - SEMESTER 2, 2019
Campus: City

STATISTICS

Data Science Practice

(Time allowed: 50 Minutes)

INSTRUCTIONS

- Attempt ALL questions.
- Total marks are 40.

1. **[10 marks]**

```
testError <- function() {
    test <- sample(1:10, 1)
    fit <- lm(duration ~ distance, trips[-test, ])
    pred <- predict(fit, trips[test, ])
    (pred - trips$duration[test])^2
}
```

[7 marks]

```
> sqrt(mean(sapply(1:100, function(i) testError())))
```

```
[1] 393.0031
```

This expression is basically estimating the Root Mean Square Error for the simple linear regression model for the `trips` data frame. The `testError()` function is calculating a single estimate of test error, `sapply()` is used to call `testError()` 100 times (which generates 100 estimates of test error), then `mean()` averages those estimates, and `sqrt()` takes the square root.

[3 marks]

2. **[10 marks]**

```
head -1 trips.csv > subset.csv
grep scooter trips.csv >> subset.csv
wc -l subset.csv
```

We call the `head` command, which returns just the first line (because of the `-1` option) of the file `trips.csv` and we redirect the result to a new file called `subset.csv`.

Next, we call the `grep` command, which searches the file `trips.csv` for lines that contain the word `scooter`. The result is the matching lines, which we redirect *and append* (because of the *double* greater than) to the file `subset.csv`.

Finally, we call the command `wc` which counts (only) the number of lines in the result (because of the `-l` option), and prints that number (one header line, plus nine scooter lines = `10`).

[5 marks]

```
report.html: report.Rmd
        Rscript -e "rmarkdown::render(\"report.Rmd\")"
```

The first line consists of a "target", `report.html`, and a "dependency", `report.Rmd`. The target is a file that we want to create and the dependency is a file that we use to create the target.

The second line is the "recipe" used to create the target from the dependency. This recipe runs the `Rscript` command, which runs R and evaluates the R expression given in the `-e` option. That R expression runs the `render()` function from the **rmarkdown** package to build the `report.html` HTML file from the `report.Rmd` R Markdown file. The recipe is run only if the dependency is newer than the target.

The `touch` command modifies the file `report.Rmd`. When we type `make` the recipe is run and the file `report.html` is created (assuming no errors). When we type `make` a second time the recipe is *not* run (we get a message about the target being "up to date") because the target is now newer than the dependency.

[5 marks]

3. **[10 marks]**

```
<months>
{
  for $i in doc("pets.xml")//row/row
    let $n := number($i/pets_adopted)
    where $n < 200
    order by $n
    return $i/month
}
</months>
```

The month elements are literal XML output. The parentheses { } bracket an enclosed FLWOR expression. This expression selects every row element that has a row element parent, defines a new variable $n that contains the pets_adopted element within the row, eliminates any row elements that have pets_adopted element with content 200 or more, orders the remaining row elements by the content of their child pets_adopted elements, and returns the child month elements of those row elements.

[5 marks]

```
<months>
  <month>Mar</month>
  <month>Apr</month>
  <month>Jan</month>
  <month>May</month>
  <month>Feb</month>
</months>
```

[5 marks]

4. **[10 marks]**

```
> library(jsonlite)
> fromJSON(readLines("luke.json"))

$name
[1] "Luke Skywalker"

$height
[1] "172"

$mass
[1] "77"

$hair_color
[1] "blond"

$skin_color
[1] "fair"

$eye_color
[1] "blue"

$gender
[1] "male"

$homeworld
[1] "https://swapi.co/api/planets/1/"

$films
[1] "https://swapi.co/api/films/2/" "https://swapi.co/api/films/6/"
[3] "https://swapi.co/api/films/3/" "https://swapi.co/api/films/1/"
[5] "https://swapi.co/api/films/7/"
```

[3 marks]

```
> m <- mongo("starwars")
> m$find(query='{ "gender": "male" }',
+        fields='{ "_id": 0, "name": 1, "height": 1, "mass": 1 }',
+        limit=5)
```

[7 marks]