MT5763: Software for Data Analysis
Individual coursework 1

John Joseph Valletta

21 September 2020

Housekeeping

- This **individual** coursework 1 comprises **20%** of your overall module mark.
- This is an **individual** project. The submitted coursework should reflect the work of you as an **individual**. Suspected cases of copying will be taken very seriously, so **please** adhere to the University's guidelines on good academic practice. If you have any uncertainties or questions about this, please contact me.
- I recommend you attempt every part of the assignment; even if you do not complete everything, marks are likely to be awarded for incomplete tasks / code. Remember, I cannot allocate marks to a blank sheet of paper, so help me to help you.
- **All** of the tasks / analysis should be completed in R, specifically making use of packages available in `tidyverse`. There should be **no** manual manipulation of these datasets. The original datasets should remain intact.

Submission

- This coursework is about creating a **succinct** and **reproducible** report with R Markdown, which is version controlled using **Git**, and hosted in a private **GitHub** repository. The report **must** contain a link to the private **GitHub** repository where it is hosted.
- You are required to upload to Moodle a **single** file - a compiled / knitted R Markdown report, either as a PDF or HTML. Name the file `MT5763_1_<ID>`, where `<ID>` is your University student ID e.g. `MT5763_1_12345.pdf` or `MT5763_1_12345.html`.
- Deadline is **Wednesday, 7th October 2020, 23:59 (UK time)**. **PLEASE** do not leave it to the last minute to upload your work.
- The School has a lateness policy. The standard policy is an initial penalty of 15% of the maximum available mark, then a further 5% per 8-hour period, or part thereof.

Marking guidance

The following criteria carry an *equal* amount of marks:

- **Code** - readable, logical, reproducible, tidy and appropriately commented.
- **Report** - succinct and well-presented, with a consistent narrative throughout.
- **Version control** - use of Git / GitHub, exhibiting frequent and well-documented commits.
- **Data wrangling** - use of the pipe operator (`%>%`) for enhanced readability.
- **Data visualisation** - well-annotated graphs and a clear description of the insights gained.
- **Statistical modelling** - appropriate analysis and interpretation.

Assignment

Version control

- Create a **private** GitHub repository called `MT5763_1_<ID>`, where `<ID>` is your University student ID. It is important that you set the repo to private to avoid any temptation of peeking at your colleague's repos.
- You **only** need to version control your `*.Rmd` file, nothing else.
- Make sure you **commit** changes often and include a succinct commit message that clearly describes what you changed and why. You will **not** be penalised for committing changes to fix mistakes in your code - this is one of the reasons why version control is used!
- *Before* submitting your coursework invite me as a collaborator to your repo so that I'm able to access it. Instructions can be found here. My username is `jjvalletta`.
- Include a link to your GitHub repo at the start of your R Markdown report.

Data description

There are two datasets you'll be working on (both available to download from Moodle):

- `BikeSeoul.csv`
- `BikeWashingtonDC.csv`

Rental bike sharing systems have been introduced in many cities worldwide to provide an accessible and sustainable mode of transport. These datasets contain the number of bikes rented at each hour in Seoul, South Korea (`BikeSeoul.csv`) and Washington, D.C., USA (`BikeWashingtonDC.csv`), together with corresponding meteorological and holiday data.

We will use these two cities as examples to explore the relationships between bike usage, weather, time of day and holidays. Understanding these relationships is important to eventually build appropriate statistical models to *predict* bike demand at various times of the year. These predictions can then be used, for example, to schedule bike maintenance.

The datasets contain the following variables:

- `BikeSeoul.csv`
  - `Date` - Day / Month / Year
  - `Rented Bike count` - Number of bikes rented in that *hour*
  - `Hour` - Hour of the day
  - `Temperature` - Air temperature in degree Celsius
  - `Humidity` - As a %
  - `Windspeed` - In m/s
  - `Visibility` - In $10m$ units (i.e. visibility = 2000, means a $20km$ visibility)
  - `Dew point temperature` - In degree Celsius
  - `Solar radiation` - In MJ/m^2
  - `Rainfall` - In mm
  - `Snowfall` - In cm
  - `Seasons` - Winter, Spring, Summer, Autumn
  - `Holiday` - Holiday / No holiday
  - `Functional Day` - Yes / No bike count data collected
- `BikeWashingtonDC.csv`
  - `instant` - Unique record index
  - `dteday` - Day / Month / Year
  - `season` - Season (1: Winter, 2: Spring, 3: Summer, 4: Autumn)

- `yr` - Year (0: 2011, 1:2012)
- `mnth` - Month
- `hr` - Hour
- `holiday` - 0: no holiday, 1: holiday
- `weekday` - Day of the week
- `workingday` - 0: holiday / weekend, 1: otherwise
- `weathersit` - Weather condition
  1. clear, few clouds, partly cloudy
  2. mist & cloudy, mist & broken clouds, mist & few clouds, mist
  3. light snow, light rain & thunderstorm & scattered clouds, light rain & scattered clouds
  4. Heavy rain & ice pellets & thunderstorm & mist, snow & fog
- `temp` : Normalised air temperature in degree Celsius. The values are computed via $\frac{t-t_{min}}{t_{max}-t_{min}}$ where $t_{min} = -8°C$ and $t_{max} = +39°C$
- `atemp`: Normalised feeling temperature in degree Celsius. The values are computed via $\frac{t-t_{min}}{t_{max}-t_{min}}$, where $t_{min} = -16°C$ and $t_{max} = +50°C$
- `hum`: Normalised humidity. The values are divided by 100
- `windspeed`: Normalised wind speed. The values are divided by $67km/h$ (max)
- `casual`: Number of bikes rented by casual users
- `registered`: Number of bikes rented by registered users
- `cnt`: Total number of bikes rented in that *hour* (i.e. casual + registered)

Data wrangling

After reading the data in, first step is to clean it for downstream analysis. In particular, perform the following operations:

`BikeSeoul.csv`

- Remove the following columns: visibility, dew point temperature, solar radiation, rainfall and snowfall[1].
- Filter out observations for which no bike count data was collected, then remove the functioning day column as it is no longer required.
- Where necessary, change the name of the columns to the following names (you will do the same for the Washington data to have a consistent set of variable names across both datasets):
  - `Date` - Day / Month / Year
  - `Count` - Number of bikes rented in that hour
  - `Hour` - Hour of the day
  - `Temperature` - Air temperature in degree Celsius
  - `Humidity` - As a %
  - `WindSpeed` - In m/s
  - `Season` - Winter, Spring, Summer, Autumn
  - `Holiday` - Holiday / No holiday
- Convert `Date` to a date object.
- Create a new variable called `FullDate` which includes the hour in it (set minute and second to zero). For example, if `Date` = `2017-12-01` and `Hour` = 15, then `FullDate` = `2017-12-01 15:00:00`. *Hint*: Check the `make_datetime` function in the `lubridate` package.
- Change the factor levels of `Holiday` to `Yes` / `No` (use this *order*).
- Change the *order* of the `Season` factor levels to `Spring`, `Summer`, `Autumn` and `Winter`.

[1] Yes, these can be important predictors of bike usage but for the purpose of this assignment you are going to concentrate on a smaller subset of covariates.

`BikeWashingtonDC.csv`

- Remove the following columns: unique record index, year, month, day of the week, working day, weather condition, normalised *feeling* temperature and number of bikes rented by casual and registered users (i.e. keep only the total count).
- Change the name of the columns to match the ones for Seoul.
- Convert `Humidity` to a %.
- Convert `Temperature` to degrees Celsius.
- Convert `WindSpeed` to m/s.
- Change the factor levels of `Season` to `Spring`, `Summer`, `Autumn` and `Winter` (in this *order* to match Seoul's one).
- Change the factor levels of `Holiday` to `Yes` / `No` (use this *order*).
- Convert `Date` to a date object.
- Create a new variable called `FullDate` which includes the hour in it (set minute and second to zero). For example, if `Date` = `2017-12-01` and `Hour` = 15, then `FullDate` = `2017-12-01 15:00:00`. *Hint*: Check the `make_datetime` function in the `lubridate` package.

The Seoul and Washington data frame objects should now have the same set of consistently named and comparable columns.

Data visualisation

Next, explore (visually) the associations between bike usage, weather, time of day and holidays for **both** the Seoul and Washington datasets. Produce any number of relevant plots to answer the following questions, and comment on the similarities / differences between Seoul and Washington.

- How does air temperature varies over the course of a year?
- Do seasons affect the average number of rented bikes?
- Do holidays increase or decrease the demand for rented bikes?
- How does the time of day affect the demand for rented bikes?
- Is there an association between bike demand and the three meteorological variables (air temperature, wind speed and humidity)?

Statistical modelling

For **both** the Seoul and Washington datasets do the following:

- Fit a linear model with log count as outcome, and season, air temperature, humidity and wind speed as predictors. Print out a summary of the fitted models, comment on the results and compare across the two cities.
- Display the 97% confidence intervals for the estimated regression coefficients. Do you think these confidence intervals are reliable?
- Assuming the model is trustworthy, what's the expected number of rented bikes in winter when the air temperature is freezing ($0°C$), in the presence of light wind ($0.5m/s$) and a humidity of 20%. Provide the 90% **prediction** intervals and comment on the results. *Hint*: Use the `interval` argument of the `predict` function.