~~~
---
title: "Combined R Markdown document"
author: "B210741"
date: "18/06/2022"
output:
  pdf_document: default
  html_document: default
  word_document: default
---
```{r setup, include=FALSE}

knitr::opts_chunk$set(echo = TRUE)

~~~
1. Link to Github repository:
https://github.com/B210741/B210741_assessment.git

2. Constructing a data dictionary and appending it to the data

Loading NHSRdatasets

```{r}
install.packages('dataMeta')
library(dataMeta)
library(NHSRdatasets)
library(tidyverse)
library(here)
library(knitr)
library(scales)
library(lubridate)
library(caret)

#Load the ae_attendances data.
data(ae_attendances)

```
Exploring the ae_attendances data

The dataset set I have chosen to manage from the NHSRdatasets package is the NHS England accident and emergency (A&E) attendances and admissions (ae_attendances) data. The ae_attendances data includes reported attendances, four-hour breaches and admissions for all A&E departments in England for 2016/17 through 2018/19 (Apr-Mar). We previously selected a subset of the variables needed for my data capture tool, including period, attendances and breaches, and subsetted the data into test and training data.
```
```

~~~~
○
○ ```{r}
○ data(ae_attendances)
○ ae<-ae_attendances
○ class(ae)
○ #[1] "tbl_df"     "tbl"        "data.frame"
○
○ ae
○
○ ## # A tibble: 12,765 × 6
○ ##    period     org_code type  attendances breaches admissions
○ ##    <date>     <fct>    <fct>       <dbl>    <dbl>      <dbl>
○ ##  1 2017-03-01 RF4      1           21289     2879       5060
○ ##  2 2017-03-01 RF4      2             813       22          0
○ ##  3 2017-03-01 RF4      other        2850        6          0
○ ##  4 2017-03-01 R1H      1           30210     5902       6943
○ ##  5 2017-03-01 R1H      2             807       11          0
○ ##  6 2017-03-01 R1H      other       11352      136          0
○ ##  7 2017-03-01 AD913    other        4381        2          0
○ ##  8 2017-03-01 RYX      other       19562      258          0
○ ##  9 2017-03-01 RQM      1           17414     2030       3597
○ ## 10 2017-03-01 RQM      other        7817       86          0
○ ## # … with 12,755 more rows
○ ```
○ ```{r}
○ glimpse(ae)
○ head(ae)
○ tail(ae, n=4)
○
○ # Missing value
○ ae %>%
○   map(is.na) %>%
○ map(sum)
○
○ #There are no missing data. The data is complete.
○
○ #Add an index link column to ae_attendances data
○ ae <- rowid_to_column(ae, "index")
○
○ #Tabulate the raw data for the report
○
○ ae %>%
○   # Set the period column to show in month-year format
○   mutate_at(vars(period), format, "%b-%y") %>%
○   # Set the numeric columns to have a comma at the 1000's place
○   mutate_at(vars(attendances, breaches, admissions), comma) %>%
~~~~

~~~~
  # Show the first 10 rows
  head(10) %>%
  # Format as a table
  kable()
```
```{r}
#Calculate monthly four hour waiting time target performance for England as a whole

ENG_performance <- ae %>%
  group_by(period) %>%
  summarise_at(vars(attendances, breaches), sum) %>%
  mutate(performance = 1 - breaches / attendances)
glimpse(ENG_performance)
```
```{r}
#Visualise monthly four hour waiting time target

ggplot(ENG_performance, aes(period, performance)) +
  geom_line(color = "darkcyan") +
  geom_point(color = "darkcyan") +
  scale_y_continuous(labels = percent) +
  scale_x_date(date_labels = "%b-%y", date_breaks = "11 month")+
  labs(x = "Month of attendance",
       y = "% of A&E attendances that met the four hour standard",
       title = "NHS England accident and emergency (A&E) four hour performance",
       caption = "Source: NHSRdatasets")
```
```{r}
#Select the ae_attendances data subset for further exploratory analysis

ae %>%
  # set the period column to show in Month-Year format
  mutate_at(vars(period), format, "%b-%y") %>%
  # set the numeric columns to have a comma at the 1000's place
  mutate_at(vars(attendances, breaches), comma) %>%
  # show the first 10 rows
  head(10) %>%
  # format as a table
  kable()
```
```{r}
#save provisional subsetted ae_attendances data to the ‘RawData’ folder

write_csv(ae, here("RawData", "ae_attendances_ENG_4hr_perfom.csv"))
```
~~~~

~~~~
#### Separating provisional ae_attendances_ENG_4hr_perfom data into training and testing sets

Splint the raw data into test and training data sets

```{r}
#The ae_attendances_ENG_4hr_perfom dataset is large
nrow(ae) #rows of data
#[1] 12765

prop<-(1-(15/nrow(ae)))
#The proportion of the raw that needs to be assigned to the training data to ensure there is only 10 to 15 records in the test data is:

print(prop)
#[1] 0.9988249

set.seed(333)

#Partitioning the raw data into the test and training data.
trainIndex <- createDataPartition(ae$index, p = prop,
                                  list = FALSE,
                                  times = 1)
head(trainIndex)

# All records that are in the trainIndex are assigned to the training data.

aeTrain <- ae[ trainIndex,]
nrow(aeTrain)
#[1] 12753 records
```
```{r}
CollectedData=read_csv(here("RawData", "CollectedDataAll.csv"))

glimpse(CollectedData)
```
#### Let's tabulate ae_attendances_ENG_4hr_perfom training data for your report
```{r}
aeTrain %>%
  # set the period column to show in Month-Year format
  mutate_at(vars(period), format, "%b-%y") %>%
  # set the numeric columns to have a comma at the 1000's place
  mutate_at(vars(attendances, breaches), comma) %>%
  # show the first 10 rows
  head(10) %>%
  # format as a table
~~~~

-   kable()
-
- write_csv(aeTrain, here("Data", "ae_attendances_ENG_4hr_perfom_train.csv"))

-
  ```
-
- #### Let's extract the ae_attendances_ENG_4hr_perfom test data
-
- ```{r}
- #All records that are not in the trainIndex (`-trainIndex`) are assigned to the test data.
- aeTest  <- ae[-trainIndex,]
- nrow(aeTest)
- #[1] 12
- ```
- There are 12 records in your test data
-
- #### Set aside the first record from the ae_attendances_ENG_4hr_perfom test data
-
- ```{r}
- aeTestMarker  <- aeTest[1,]
- ```
- #### Tabulate ae_attendances_ENG_4hr_perfom marker test data for the report
-
- ```{r}
- aeTestMarker  %>%
-   # set the period column to show in Month-Year format
-   mutate_at(vars(period), format, "%b-%y") %>%
-   # set the numeric columns to have a comma at the 1000's place
-   mutate_at(vars(attendances, breaches), comma) %>%
-   # show the first 10 rows
-   head(10) %>%
-   # format as a table
-   kable()
- ```
- #### Save the ae_attendances_ENG_4hr_perfom marker test data to the working data folder 'Data'
-
- ```{r}
- write_csv(aeTestMarker, here("Data", "ae_attendances_ENG_4hr_perfom_test_marker.csv"))
-
- ```
- #### Setting aside the remaining records for testing data-capture tool.
-
- ```{r}
- aeTest  <- aeTest[2:nrow(aeTest),]
- ```
  ```

````
○ #### Tabulate ae_attendances_ENG_4hr_perfom test data for the report
○ ```{r}
○ aeTest  %>%
○   # set the period column to show in Month-Year format
○   mutate_at(vars(period), format, "%b-%y") %>%
○   # set the numeric columns to have a comma at the 1000's place
○   mutate_at(vars(attendances, breaches), comma) %>%
○   # show the first 10 rows
○   head(10) %>%
○   # format as a table
○   kable()
○ ```
○ #### Save the ae_attendances_ENG_4hr_perfom test data to the working data folder 'Data'
○ ```{r}
○ write_csv(aeTest, here("Data", "ae_attendances_test.csv"))
○ ```
○
○ ### The CollectedData dataset contains:
○
○ **index**: the index column that allows us to link the data collected to the original ae_attendances data in the ‘RawData’ folder.
○
○ **period**: the month that this activity relates to, stored as a date (1st of each month).
○
○ **org_code**: the Organisation data service (ODS) code for the organisation. The ODS code is a unique code created by the Organisation data service within NHS Digital and used to identify organisations across health and social care. ODS codes are required in order to gain access to national systems like NHSmail and the Data Security and Protection Toolkit. If you want to know the organisation associated with a particular ODS code, you can look it up from the following address: https://odsportal.digital.nhs.uk/Organisation/Search. For example, the organisation associated with the ODS code ‘AF003’ is Parkway health centre.
○
○ **type**: the Department Type for this activity, either
○
○   *1: Emergency departments are a consultant-led 24-hour service with full resuscitation facilities and designated accommodation for the reception of accident and emergency patients,
○
○   *2: Consultant-led mono speciality accident and emergency service (e.g. ophthalmology, dental) with designated accommodation for the reception of patients, or
○
````

- *other: Other types of A&E/minor injury activity with designated accommodation for the reception of accident and emergency patients. The department may be doctor-led or nurse-led and treats at least minor injuries and illnesses and can be routinely accessed without an appointment. A service mainly or entirely appointment-based (for example, a GP Practice or Outpatient clinic) is excluded even though it may treat a number of patients with minor illness or injury. Excludes NHS walk-in centres.(National Health Service, 2020)
-
- **attendances**: the number of attendances for this department type at this organisation for this month.
- breaches: the number of attendances that breached the four-hour target.
-
- **admissions**: the number of attendances that resulted in an admission to the hospital.(Chris Mainey, 2021)
-
- **performance**: the performance ([1 - breaches]/attendances) calculated for the whole of England.
-
- **consent**: the consent from the end-user to process and share the data collected with the data capture tool.
-
-
- ## 3. Constructing a data dictionary and appending it to the data
-
- ### Build a linker data frame
-
- #### Variable descriptions
-
- Create a string vector representing the different variable descriptions
-
- ```{r}
- variable_description <- c("The index column that allows us to link the data collected to the original ae_attendances data in the 'RawData' folder.",
- "The month that this activity relates to, stored as a date (1st of each month).",
- "The Organisation data service (ODS) code for the organisation. If you want to know the organisation associated with a particular ODS code, you can look it up from the following address: https://odsportal.digital.nhs.uk/Organisation/Search.",
- "The department type for this activity.",
- "The number of attendances for this department type at this organisation for this month.",
- "The number of attendances that breached the four-hour target.",
- "The number of attendances that resulted in an admission to the hospital.",
- "The performance ([1 - breaches]/attendances) calculated for the whole of England.",
- "The consent from the end-user to process and share the data collected with the data capture tool.")
-

- print(variable_description)

-
  ```
-
- #### Variable types
- ```{r}
- glimpse(CollectedData)
- ```
- We have five quantitative values (measured values) variables and four fixed values (allowable values or codes) variables.
-
- ```{r}
- variable_type <- c(0, 1, 1, 1, 0, 0, 0, 0, 1)
- print(variable_type)
-
- linker<-build_linker(CollectedData, variable_description, variable_type)
-
- print(linker)
- ```
-
- ### Data dictionary
-
- Using the build_dict() function from the dataMeta to constructs a data dictionary for a CollectedData data frame with the aid of the linker data frame between.
-
- ```{r}
- dictionary <- build_dict(my.data = CollectedData, linker = linker)
-
- glimpse(dictionary)
-
- dictionary[6,4]<-"C82010: Prescribing Cost Centre - OAKHAM MEDICAL PRACTICE."
- dictionary[7,4]<-"RDZ: NHS Trust - The Royal Bournemouth and Christchurch Hospitals NHS Foundation Trust."
- dictionary[8,4]<-"RVR: NHS Trust - EPSOM AND ST HELIER UNIVERSITY HOSPITALS NHS TRUST."
- dictionary[9,4]<-"RQM: NHS Trust - CHELSEA AND WESTMINSTER HOSPITAL NHS FOUNDATION TRUST."
- dictionary[10,4]<-"R1F: NHS Trust - ISLE OF WIGHT NHS TRUST."
- dictionary[11,4]<-"RE9: NHS Trust - SOUTH TYNESIDE NHS FOUNDATION TRUST."
- dictionary[12,4]<-"RNL: NHS Trust - NORTH CUMBRIA UNIVERSITY HOSPITALS NHS TRUST."
- dictionary[13,4]<-"RJ1 - NHS Trust - GUY'S AND ST THOMAS' NHS FOUNDATION TRUST."
- dictionary[14,4]<-"RKB - NHS Trust - UNIVERSITY HOSPITALS COVENTRY AND WARWICKSHIRE NHS TRUST."
  ```

- dictionary[15,4]<-"NLO12 - Independent Sector H/c Provider Site - OAKHAM URGENT CARE CENTRE."
- dictionary[27,4] <-"other: Other types of A&E/minor injury activity with designated accommodation for the reception of accident and emergency patients."
- dictionary[28,4] <-"1: Emergency departments are a consultant-led 24-hour service with full resuscitation facilities and designated accommodation for the reception of accident and emergency patients."
- dictionary[29,4] <-"2: Consultant-led mono speciality accident and emergency service (e.g. ophthalmology, dental) with designated accommodation for the reception of patients."

-
  ```
- #### Save the data dictionary for CollectedData to the 'RawData' folder
- ```{r}
- write_csv(dictionary, here("RawData", "CollectedData_DataDictionary.csv"))
- ```
-
- #### Incorporating attributes as metadata to the CollectedData as metadata using the ‘incorporate_attr()’ function from the dataMeta package.
- ```{r}
- main_string <- "This data describes the NHS England accident and emergency (A&E) attendances and breaches of four-hour wait time target data from the *NHSRdatasets* package collected by the data capture tool."
-
- main_string
- ```
-
- #### Incorporate attributes as metadata
-
- Using the ‘incorporate_attr()’ function to return an R dataset containing metadata stored in its attributes. The attributes we are going to add include: * a data dictionary * number of columns * number of rows * the name of the author who created the dictionary and added it, * the time when it was last edited * a brief description of the original dataset.
- ```{r}
- complete_CollectedData <- incorporate_attr(my.data = CollectedData, data.dictionary = dictionary,
- main_string = main_string)
-
- #Change the author name
-
- attributes(complete_CollectedData)$author[1]<-"B210741"
- complete_CollectedData
- attributes(complete_CollectedData)
- ```
- #### Save the CollectedData with attributes
-
  ```

- ```{r}
- save_it(complete_CollectedData, here("RawData", "complete_CollectedData"))
-
- complete_CollectedData<-readRDS(here("RawData", "complete_CollectedData.rds"))

-
  ```
- ## 4. Data capture tool: Collecting data using interactive Jupyter widgets
-
- *Description of the code:
- Language (Python) in Jupyter
- The data is collected using interactive widgets
- The data is stored in the RawData folder
-
-
  ```