**[MA1580 Foundations of Data Science]**

|  |  |
| --- | --- |
| **Assessment Task** | Assessment 2 |
| **College** | College of Science and Engineering |

**Student:** Please sign, date, and attach this cover sheet to the front of your assessment task for all hard copy submissions.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Student Family Name** | | **Student Given Name** | **JCU Student Number** | | | | | | | | |
| LI | | YANGYANG | 1 | 3 | | 7 | 6 | 3 | 3 | 3 | 0 |
| **Assessment Title** | | Data Wrangling and Data Tiding Prior to Instrument Calibration | | | | | | | | | |
| **Due Date** | | 08/10/2020 | | | | | | | | | |
| **Lecturer Name** | | Sourav Das | | | | | | | | | |
| **Tutor Name** | |  | | | | | | | | | |
| **Student Declaration**   1. This assignment is my original work and no part has been copied/ reproduced from any other person’s work or from any other source, except where acknowledgement has been made (see *Learning, Teaching and Assessment Policy 5.1*). 2. This work has not been submitted previously for assessment and received a grade OR concurrently for assessment, either in whole or part, for this subject (unless part of integrated assessment design/approved by the Subject Coordinator), any other subject or any other course (see *Learning, Teaching and Assessment Policy 5.9*). 3. This assignment has not been written for me. 4. We hold a copy of this assignment and can produce a copy if requested. 5. This work may be used for the purposes of moderation and identifying plagiarism. 6. We give permission for a copy of this marked assignment to be retained by the College for benchmarking and course review and accreditation purposes.   [Learning, Teaching and Assessment Policy](https://www.jcu.edu.au/policy/learning-and-teaching/learning-teaching-and-assessment-policy) 5.1. A student who submits work containing plagiarised material for assessment will be subject to the provisions of the [Student Academic Misconduct Requirements Policy](https://www.jcu.edu.au/policy/student-services/student-academic-misconduct-requirements-policy).  **Note the definition of plagiarism and self plagiarism in the Learning, Teaching and Assessment Policy:**  **Plagiarism:** reproduction without acknowledgement of another person’s words, work or expressed thoughts from any source. The definition of words, works and thoughts includes such representations as diagrams, drawings, sketches, pictures, objects, text, lecture hand-outs, artistic works and other such expressions of ideas, but hereafter the term ‘work’ is used to embrace all of these. Plagiarism comprises not only direct copying of aspects of another person’s work but also the reproduction, even if slightly rewritten or adapted, of someone else’s ideas. In both cases, someone else’s work is presented as the student’s own. Under the Australian Copyright Act 1968 a copyright owner can take legal action in the courts against a party who has infringed their copyright.  **Self Plagiarism:** the use of one’s own previously assessed material being resubmitted without acknowledgement or citing of the original. | | | | | | | | | | | |
| **Student Signature** |  | | | | Submission Date: 08/10/2020 | | | | | | |

Tutorial Project – Total marks 60

## Part A – Data Wrangling and Data Tiding Prior to Instrument Calibration

## Setting up the RStudio environment

Table 1 **Use** this R code to set-up the RStudio environment

|  |
| --- |
| #~~~~~~~~~~~  # Libraries  #~~~~~~~~~~  library(dplyr)  library(ggplot2)  library(tidyr)  #~~~~~~~~~~~  # Functions  #~~~~~~~~~~  rescale\_01 <-function(x) (x-min(x))/(max(x)-min(x)) -1/2  z\_stand<-function(x) (x-mean(x))/sd(x)  ExpectedBrix <- function(x) (x\*0.21084778699754 + 4.28455310831511)  #~~~~~~~~~~~~  # Thresholds  #~~~~~~~~~~~  Thresh.Brix.min <- 15  Thresh.Brix.max <- 30  Thresh.Pol.min <- 50  Thresh.Pol.max <- 105  ExpectedBrix.delta <- 1  Thresh.Fibre.min <- 4  Thresh.Fibre.max <- 25  Thresh.Fibre.delta <- .25  Thresh.Ash.min <- 0  Thresh.Ash.max <- 8 |

## Fibre Data

Table 2 **Enter** your R code you used to import the Fibre data into a data table called “Lab\_Fibre\_Data” Marks (1):

|  |
| --- |
| Lab\_Fibre\_Data = read.csv(file = "Sugar\_Cane\_Input\_Files/Lab\_Fibre\_Weights.csv",  header = T, sep = ",", dec = ".") |

## Calculate Percentage Fibre Variables

Table 3 **Calculate** the fibre percentage of the first set of measurements (columns 2 to 4, named SampleWeight\_1, InitialSampleCanWeight\_1, and FinalSampleCanWeight\_1). Name the resulting variable “Fibre1” and add this new variable to the “Lab\_Fibre\_Data” data.frame, using direct assignment using base R (without any package). **Enter** your R code you used: Marks(2):

|  |
| --- |
| Lab\_Fibre\_Data$Fibre1 = 100 \* (Lab\_Fibre\_Data$InitialSampleCanWeight\_1 -  Lab\_Fibre\_Data$FinalSampleCanWeight\_1) / Lab\_Fibre\_Data$SampleWeight\_1 |

Table 4 **Repeat** the above procedure for the second set of measurements (columns 5 to 7, named SampleWeight\_2, InitialSampleCanWeight\_2, and FinalSampleCanWeight\_2), but now using an appropriate function from the dplyr package (rather than direct assignment) to calculate the corresponding fibre percentage, “Fibre2”, and add it as a new variable to the “Lab\_Fibre\_Data” data table. **Enter** your R code you used: Marks(1):

|  |
| --- |
| Lab\_Fibre\_Data = Lab\_Fibre\_Data %>% mutate(Fibre2 = 100 \*  (InitialSampleCanWeight\_2 - FinalSampleCanWeight\_2) / SampleWeight\_2) |

## Filtering Fibre Variables-

Table 5 **Use** a function from the **dplyr** package to remove samples (rows) that contain a missing value in **any** of the weight measurements. Since weights cannot be negative, do that by keeping only the rows that have positive values (> 0) for all the six raw weight measurements. Save the filtered data to a new data table called Lab\_Fibre\_Filtered. **Enter** your R code you used: Marks(1)

|  |
| --- |
| Lab\_Fibre\_Filtered = Lab\_Fibre\_Data %>% filter(InitialSampleCanWeight\_1 > 0 & FinalSampleCanWeight\_1 > 0  & SampleWeight\_1 > 0 & InitialSampleCanWeight\_2 > 0  & FinalSampleCanWeight\_2 > 0 & SampleWeight\_2 > 0) |

Table 6 **UPDATE** your code in table 5 to include this maximum fibre difference limit as an additional filtering criteria. **Enter** your R code you used: Marks(1)

|  |
| --- |
| Lab\_Fibre\_Filtered = Lab\_Fibre\_Data %>% filter(InitialSampleCanWeight\_1 > 0 & FinalSampleCanWeight\_1 > 0  & SampleWeight\_1 > 0 & InitialSampleCanWeight\_2 > 0  & FinalSampleCanWeight\_2 > 0 & SampleWeight\_2 > 0) %>% filter(abs(Fibre1 - Fibre2) < 0.25) |

Table 7 **Calculate** the final fibre estimates by averaging the replicate fibre measurements, “Fibre1” and “Fibre2”. Use the correct **dplyr** function to calculate the average fibre and add it as a new variable named “Fibre” to the Lab\_Fibre\_Filtered data table. **Enter** your R code you used: Marks(1)

|  |
| --- |
| Lab\_Fibre\_Filtered = Lab\_Fibre\_Filtered %>% mutate(Fibre = (Fibre1 + Fibre2)/2) |

Table 8 **Use** an approach prescribed in **dplyr** package to implement the following steps in sequence- filter the measurements in Lab\_Fibre\_Filtered to remove the out-of-range fibre values, first keeping only the rows of the data table for which “Fibre” is greater than Thresh.Fibre.min, and then keeping only the resulting rows for which “Fibre” is less than Thresh.Fibre.max. Save the resulting data into the Lab\_Fibre\_Filtered table. **Enter** your R code you used: Marks(2):

|  |
| --- |
| Lab\_Fibre\_Filtered = Lab\_Fibre\_Filtered %>% filter(Fibre > 4) %>% filter(Fibre < 25) |

Table 9 **Use** an appropriate **function** from the **dplyr package** to save the LabID and Fibre variables (1st and last columns) from Lab\_Fibre\_Filtered to a new data table called Lab\_Fibre. **Enter** your R code you used: Marks(1):

|  |
| --- |
| Lab\_Fibre = Lab\_Fibre\_Filtered %>% select(LabID, Fibre) |

## Ash Data

Table 10 **Enter** your R code you used to import the Ash data into a data table called “Lab\_Ash\_Data” Marks (1)

|  |
| --- |
| Lab\_Ash\_Data = read.csv(file = "Sugar\_Cane\_Input\_Files/Lab\_Ash\_Weights.csv",  header = T, sep = ",", dec = ".") |

## Calculate Ash Variables

Table 11 **Use** a **dplyr** based approach to implement the following steps in sequence (a) first filter out the missing values , then sequentially calculate (b) “InitialWeight”, (c) “FinalWeight” and (d) “Ash” as new variables. You **must** only use **dplyr** based functions. Save your result to a new data table Lab\_Ash\_Calculated. **Enter** your R code you used: Marks (3)

|  |
| --- |
| Lab\_Ash\_Calculated = Lab\_Ash\_Data %>% filter(TinWeight > 0 & InitialSampleInTinWeight > 0  & FinalSampleInTinWeight > 0) %>%  mutate(Ash = 100 \* (FinalSampleInTinWeight - TinWeight ) / (InitialSampleInTinWeight - TinWeight) ) |

## Filtering Ash Variables

Table 12 **Update** your **previous data** from table 11 to filter out any out-of-range Ash values. Again using a **function** from the **dplyr package**. **Enter** your R code you used: Marks(1)

|  |
| --- |
| Lab\_Ash\_Filtered = Lab\_Ash\_Calculated %>% filter(Ash > 0) %>% filter(Ash < 8) |

## Summarising Ash Variables

Table 13 **Use appropriate functions** from the **dplyr package** to produce a data table called Lab\_Ash that is grouped by LabID and provides summaries of Ash by taking its grouped mean values. The resulting table, Lab\_Ash, must then have two variables, LabID and Ash, where LabID now contains unique values (no replicates). **Enter** your R code you used: Marks (2)

|  |
| --- |
| Lab\_Ash = Lab\_Ash\_Filtered %>% group\_by(LabID) %>% summarise(Ash = mean(Ash)) |

## Pol and Brix Data

*Table 14* ***Enter*** *your R code you used to import the Pol and Brix data into a data table called “Lab\_PB\_Data”* Marks(1)

|  |
| --- |
| Lab\_Pol\_Brix = read.csv(file = "Sugar\_Cane\_Input\_Files/Lab\_Pol\_Brix.csv",  header = T, sep = ",", dec = ".") |

Table 15 **Use** an appropriate **function** from the **dplyr package** to add to the data table Lab\_PB\_Data a new variable, PredBrix, which uses the ExpectedBrix function with Pol as input. **Enter** your R code you used: Marks(1)

|  |
| --- |
| Lab\_PB\_Data = Lab\_PB\_Data %>% mutate(PredBrix = ExpectedBrix(Pol)) |

Table 16 **Use** anappropriate R functionto visually display the relationship between Brix and Pol. In your plot use two different colours to distinguish the observations based on absolute difference between the measured Brix (variable Brix, 3rd column) and the predicted Brix (variable PredBrix, 4th column) is greater than one (or not). **Enter** your R code **and** the jpeg of the plot: Marks(2)

|  |
| --- |
| z = factor(ifelse(abs(Lab\_PB\_Data$Brix - Lab\_PB\_Data$PredBrix) > 1, 1, 0))  plot(Lab\_PB\_Data$Brix,Lab\_PB\_Data$PredBrix, col = z, pch = 16, main = "Relationship between measured Brix and predicted Brix", xlab = "Measured Brix", ylab = "Predicted Brix")  legend("bottomright", legend = c("z=0","z=1"), col = c("red", "black"),pch = rep(16,2)) |

Table 17 **Use** the correct **dplyr** based approach to sequentially filter out undesirable rows from Lab\_PB\_Data, and then select only a subset of its variables, as follows: first, filter out samples (rows) where (a) the absolute difference between the measured Brix and predicted Brix is greater than one, and/or (b) any value for Pol or Brix are out of range (the min. and max. values are specified in the threshold variables we initially set-up, namely, Thresh.Brix.min, Thresh.Brix.max, Thresh.Pol.min, and Thresh.Pol.max). Then, (c) select only the variables LabID, Pol and Brix to constitute a new data table, called Lab\_PB. **Enter** your R code you used: Marks(2)

|  |
| --- |
| Lab\_PB = Lab\_PB\_Data %>% filter(abs(Lab\_PB\_Data$Brix - Lab\_PB\_Data$PredBrix) < 1) %>%  filter(Brix > 15 & Brix < 30 & Pol > 50 & Pol < 105) %>%  select(LabID, Pol, Brix ) |

## A Single Lab File

Table 18 **Use** the following R code to join the Fibre and Ash tables together.

|  |
| --- |
| Lab <- full\_join(Lab\_Ash, Lab\_Fibre, by=c("LabID" = "LabID")) |

Table 19 **Join** the existing Lab data table to the Lab\_PB data table. **Enter** you R code **and** the first eleven rows of the combined Lab data table. Marks(2)

|  |
| --- |
| **Code:** Lab = full\_join(Lab, Lab\_PB, by=c("LabID" = "LabID"))  **First 11 rows:** Lab[1:11,]  LabID Ash Fibre Pol Brix  1 27 2.4692517 15.57110 78.776 20.759  2 448 3.8789646 14.40024 65.056 18.108  3 1041 3.8664436 14.80769 79.970 20.881  4 1059 1.7511153 15.09137 81.652 21.519  5 1183 2.3769386 17.80312 86.610 22.838  6 2080 2.0699319 NA 71.285 19.091  7 2205 1.9506096 13.87420 79.269 21.254  8 2812 0.4389554 17.18891 86.378 23.225  9 3288 0.7803263 15.13368 75.734 20.560  10 3916 2.0039422 14.77926 78.043 20.998  11 4047 1.1112054 19.72404 87.263 22.214 |

Table 20 **Use** the following R code to save the Lab data table to disk

|  |
| --- |
| write.table(Lab, file = "Lab\_Out.csv", append = FALSE, quote = TRUE, sep = ",",  eol = "\n", na = "NA", dec = ".", row.names = FALSE, col.names = TRUE, qmethod = c("escape", "double"), fileEncoding = "") |

## Making Calibration files

Table 21 **Use** a base R function to transform the Fibre measurements from the Lab\_Fibre table using the provided z\_stand() function. Then **save** the resulting table (containing variables LabID and Fibre) to a file on disk. **Enter** the R code you use to perform both actions. Marks(2)

|  |
| --- |
| Lab\_Fibre = transform(Lab\_Fibre, Fibre = z\_stand(Fibre))  write.table(Lab\_Fibre, file = "Lab\_Fibre\_Out.csv", append = FALSE, quote = TRUE, sep = ",",  eol = "\n", na = "NA", dec = ".", row.names = FALSE, col.names = TRUE,  qmethod = c("escape", "double"), fileEncoding = "") |

Table 22 **Use** a **dplyr** approachwith two subsequent transformation operations to transform the Ash measurements from the Lab\_Ash table, first using log10(), and then using z\_stand(). **Save** the resulting table (containing variables LabID and Ash) to a file on disk. **Enter** the R code you use to perform both actions. Marks(3)

|  |
| --- |
| Lab\_Ash = Lab\_Ash %>% mutate(Ash = log10(Ash)) %>% mutate(Ash = z\_stand(Ash))  write.table(Lab\_Ash, file = "Lab\_Ash\_Out.csv", append = FALSE, quote = TRUE, sep = ",",  eol = "\n", na = "NA", dec = ".", row.names = FALSE, col.names = TRUE,  qmethod = c("escape", "double"), fileEncoding = "") |

Table 23 **Use** the following R code to create a variable which can be subsequently used for stratified sub-sampling:

|  |
| --- |
| Lab\_PB$Bbin <- cut(Lab\_PB$Brix, 40, labels = FALSE) |

Table 24 **Use** the following R code to re-cast the Bbin variable as ordinal, so it can be used for stratified sub-sampling:

|  |
| --- |
| Lab\_PB$Bbin <- as.factor(Lab\_PB$Bbin) |

Table 25 **Use** a **function** from the **dplyr package** to perform stratified sampling on the Brix meansurements, using Bbin as the grouping variable and size=50 for the number of samples in each stratification. **Name** the resulting data table as Lab\_B\_Stratified\_Balanced. Hint: In your sampling function you may need to use attribute replace = TRUE, as not all groups have fifty samples. **Enter** the R code you use to perform both actions. Marks(2)

|  |
| --- |
| Lab\_B\_Stratified\_Balanced = Lab\_PB %>% group\_by(Bbin) %>% summarise(sample\_n(Lab\_PB, size = 50, replace = T)) |

Table 26 **Use** a base R function to rescale the Brix measurements with rescale\_01() in Lab\_B\_Stratified\_Balanced. Then, **use** an appropriate **function** from the **dplyr package** to retain only the LabID and Brix variables, and **write** the resulting data table to a csv file. **Enter** the R code you use to perform the three actions. Marks(4)

|  |
| --- |
| Lab\_PB\_sampled = transform(Lab\_B\_Stratified\_Balanced, Brix = rescale\_01(Brix))  Lab\_B\_sampled = Lab\_PB\_sampled %>% select(LabID, Brix)  write.table(Lab\_B\_sampled,file = "Lab\_Brix\_Out.csv", append = FALSE, quote = TRUE, sep = ",",  eol = "\n", na = "NA", dec = ".", row.names = FALSE, col.names = TRUE,  qmethod = c("escape", "double"), fileEncoding = "") |

Table 27 **Use** a **dplyr** approach to repeat the stratified sub-sampling method used for Brix, but now for Pol. Then write the LabID and the (stratified, sub-sampled, rescaled) Pol vales to a file. **Enter** the R code you use to perform both actions. Marks(4)

|  |
| --- |
| Lab\_P\_Stratified\_Balanced = Lab\_PB %>% group\_by(Bbin) %>% summarise(sample\_n(Lab\_PB, size = 50, replace = T))  Lab\_P\_sampled = Lab\_P\_Stratified\_Balanced1 %>% mutate(Pol = rescale\_01(Pol)) %>% select(LabID, Pol)  Lab\_P\_sampled = subset(Lab\_P\_sampled, select = -Bbin)  write.table(Lab\_P\_sampled,file = "Lab\_Pol\_Out.csv", append = FALSE, quote = TRUE, sep = ",",  eol = "\n", na = "NA", dec = ".", row.names = FALSE, col.names = TRUE,  qmethod = c("escape", "double"), fileEncoding = "") |

## Part B – Processing NIRS Predictions

Table 28 **Use** this R code to set-up the RStudio environment

|  |
| --- |
| #~~~~~~~~~~~  # Libraries  #~~~~~~~~~~  library(dplyr)  library(ggplot2)  library(tidyr)  #~~~~~~~~~~~~  # Thresholds  #~~~~~~~~~~~  Thresh.Brix.min <- 15  Thresh.Brix.max <- 30  Thresh.Pol.min <- 50  Thresh.Pol.max <- 105  Thresh.Fibre.min <- 4  Thresh.Fibre.max <- 25  Thresh.Ash.min <- 0  Thresh.Ash.max <- 8 |

Table 29 **Enter** your R code you used to import the NIR data into a data table called “NIRData” Marks(1)

|  |
| --- |
| NIRData = read.csv("Sugar\_Cane\_Input\_Files/NIRPred.csv",  header = T, sep = ",", dec = ".") |

Table 30 **Use** the following R code to correctly assign the DateTime variable to the POSIXct data type. Note that the POSIXct uses input arguments that specify the format which our DateTime uses.

|  |
| --- |
| NIRData$DateTime <- as.POSIXct(NIRData$DateTime, format = "%Y-%m-%d %H:%M:%S") |

|  |
| --- |
| **Code:** NIRData$LabID = floor(NIRData$ScanID)  **First 15 rows:** NIRData [1:15,]  ScanID DateTime NIR\_Pol NIR\_Brix NIR\_Fibre NIR\_Ash GH NH LabID  1 15022.02 2023-02-15 03:33:29 72.8551 19.6316 13.7370 1.20550 0.88448 0.043454 15022  2 15022.03 2023-02-15 03:34:03 69.6456 19.0395 14.3537 2.04980 0.54922 0.083930 15022  3 15022.04 2023-02-15 03:34:40 73.2184 19.9639 13.6891 1.33870 0.84666 0.253390 15022  4 15022.05 2023-02-15 03:35:18 73.3640 19.8254 12.8837 1.32610 0.35808 0.293050 15022  5 15022.06 2023-02-15 03:35:55 70.8986 19.2695 13.2262 1.57760 0.41520 0.275710 15022  6 15022.07 2023-02-15 03:36:31 71.6570 19.4196 13.9711 1.34050 0.64383 0.114360 15022  7 15022.08 2023-02-15 03:37:09 72.2982 19.4596 14.0231 1.27340 0.21158 0.327720 15022  8 15022.09 2023-02-15 03:37:45 70.9503 18.9749 13.7746 1.32660 0.48190 0.279020 15022  9 15022.10 2023-02-15 03:38:19 76.0537 20.4080 13.8258 1.31220 0.79724 0.049488 15022  10 15022.11 2023-02-15 03:38:55 72.2094 19.2536 13.2881 0.97978 0.83986 0.274500 15022  11 15022.12 2023-02-15 03:39:33 70.8547 19.1248 14.1651 1.28000 0.74084 0.177820 15022  12 15022.13 2023-02-15 03:40:10 73.3150 19.7778 13.5854 1.01960 0.63034 0.305050 15022  13 15023.01 2023-02-15 03:41:51 90.0699 23.7222 15.2947 1.65870 0.57679 0.056305 15023  14 15023.02 2023-02-15 03:42:24 73.4581 19.3872 14.4277 2.32480 0.25767 0.172760 15023  15 15023.03 2023-02-15 03:43:00 73.0571 19.5685 13.7557 2.11350 0.42905 0.053967 15023 |

Table 31 **Use** **a base R function** with the floor() function applied to ScanID to create a new variable called LabID in the NIRData table. **Enter** your R code you used to create the new variable, **then** enter the first fifteen rows of the updated NIRData table. Marks(3)

Table 32 **Use** a **dplyr** approach to sequentially filter the NIR data by filtering out any (a) GH values greater than 3.5, (b) NH values greater than 2, (c) any out-of-range values for Pol, Brix, Fibre and Ash and (d) any sample that has a ScanID equal to -1. Save the filtered data to a new data table called NIRData\_Filtered. **Enter** your R code: Marks(2)

|  |
| --- |
| NIRData\_Filtered = NIRData %>% filter(GH < 3.5 & NH < 2) %>%  filter(NIR\_Brix > 15 & NIR\_Brix < 30 & NIR\_Pol > 50 & NIR\_Pol < 105 &  NIR\_Fibre > 4 & NIR\_Fibre < 25 & NIR\_Ash > 0 & NIR\_Ash < 8) %>%  filter(ScanID > 0) |

Table 33 **Use** functionsfrom the **dplyr package,** sequentially,on the NIRData\_Filtered tableto produce a data table called NIR\_Final which is grouped by LabID and contains, in addition to the grouped variable, the first DateTime for each group as well as the corresponding mean values for Pol, Brix, Fibre and Ash (i.e. the group means). Hint: the min() function returns the earliest date/time when applied to a date/time type variable. **Enter** your R code you used **then** enter the first fifteen rows of the updated NIR\_Final table: Marks(3)

|  |
| --- |
| **Code:** NIR\_Final = NIRData\_Filtered %>% group\_by(LabID) %>%  summarise(DateTime = min(DateTime),NIR\_Pol = mean(NIR\_Pol), NIR\_Brix = mean(NIR\_Brix),  NIR\_Fibre = mean(NIR\_Fibre), NIR\_Ash = mean(NIR\_Ash))  **First 15 rows:** NIRData [1:15,]  LabID DateTime NIR\_Pol NIR\_Brix NIR\_Fibre NIR\_Ash  *<dbl>* *<dttm>* *<dbl>* *<dbl>* *<dbl>* *<dbl>*  1 15022 2023-02-15 03:33:29 72.3 19.5 13.7 1.34  2 15023 2023-02-15 03:41:51 74.3 19.9 14.0 1.97  3 15024 2023-02-15 03:58:42 73.4 19.8 14.2 1.95  4 15025 2023-02-15 04:11:29 74.2 19.8 14.6 2.13  5 15026 2023-02-15 04:26:36 83.3 21.7 14.8 1.57  6 15027 2023-02-15 04:37:22 81.4 21.5 15.0 1.89  7 15028 2023-02-15 04:47:25 83.8 22.0 14.5 1.47  8 15029 2023-02-15 05:00:20 72.1 19.4 15.1 2.46  9 15030 2023-02-15 05:18:37 73.5 19.6 14.9 2.37  10 15031 2023-02-15 05:32:40 72.7 19.6 15.2 1.76  11 15032 2023-02-15 05:45:35 71.1 19.3 14.1 1.87  12 15033 2023-02-15 05:57:05 73.8 19.8 14.1 1.83  13 15034 2023-02-15 06:17:56 75.5 20.0 13.7 1.72  14 15035 2023-02-15 06:21:01 73.5 19.8 14.9 1.79  15 15036 2023-02-15 06:30:59 79.1 20.9 14.9 1.96 |

Table 34 **Enter** your R code to save the NIR\_Final data table to a csv file on disk. Marks(1 )

|  |
| --- |
| write.table(NIR\_Final,file = "NIR\_Final\_Out.csv", append = FALSE, quote = TRUE, sep = ",",  eol = "\n", na = "NA", dec = ".", row.names = FALSE, col.names = TRUE,  qmethod = c("escape", "double"), fileEncoding = "") |

## Visualization

## Part C- The next questions are on exploratory visualization on the saved file, NIR\_Final to assess feature relationships and variations.

Table 35 Marks (3)

***Code:*** his1 = ggplot(data = NIR\_Final) +

geom\_histogram(mapping = aes(NIR\_Brix),fill = "red", bins = 100,alpha = 0.5) +

theme\_bw() + labs(x = "Pol", y="Frequency")

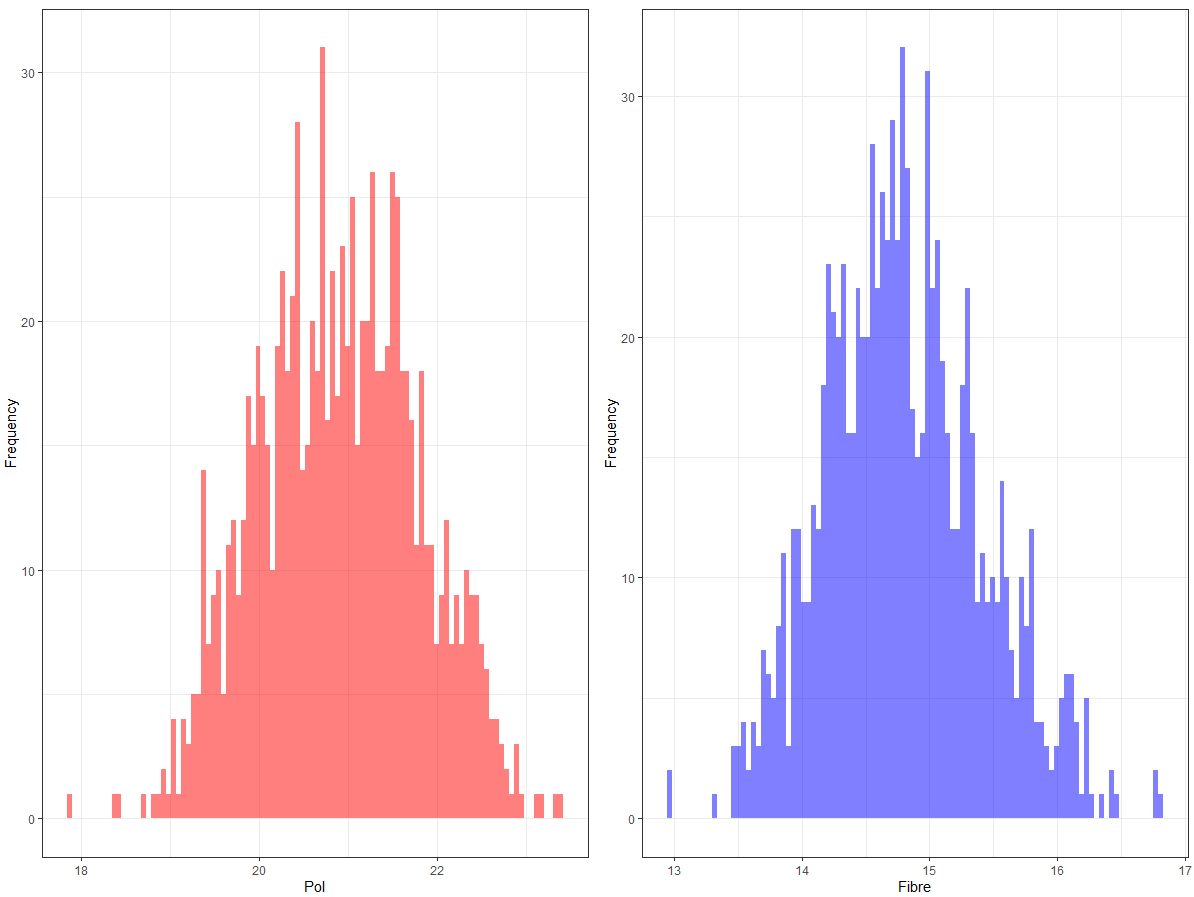
his2 = ggplot(data = NIR\_Final) +

geom\_histogram(mapping = aes(NIR\_Fibre),fill = "blue", bins = 100,alpha = 0.5) +

theme\_bw() + labs(x = "Fibre", y="Frequency")

gridExtra::grid.arrange(his1,his2,ncol = 2)

***Plot:***



As shows from the 2 plots, the mode of **Brix** and **Fibre** in the samples are all in the centre of the plot which suggests that the market value of sugarcane that grows in Queensland are significantly larger.

Table 36 For each variable (**Brix** and **Fibre**) use the appropriate dplyr function to provide quantitative descriptive summary of the above visual depiction. Marks (3)

> summary(NIR\_Final$NIR\_Brix,NIR\_Final$NIR\_Fibre)

Min. 1st Qu. Median Mean 3rd Qu. Max.

17.85 20.25 20.90 20.89 21.53 23.37

> summary(NIR\_Final$NIR\_Fibre)

Min. 1st Qu. Median Mean 3rd Qu. Max.

12.94 14.35 14.76 14.79 15.18 16.80

As the quantitative descriptive summary of both variables **Brix** and **Fibre** this is increases the certainty that the market value of sugarcane is substantially larger in Queensland as there is not a huge difference between minimum, 1st Qu., median, mean, 3rd Qu., maximum.

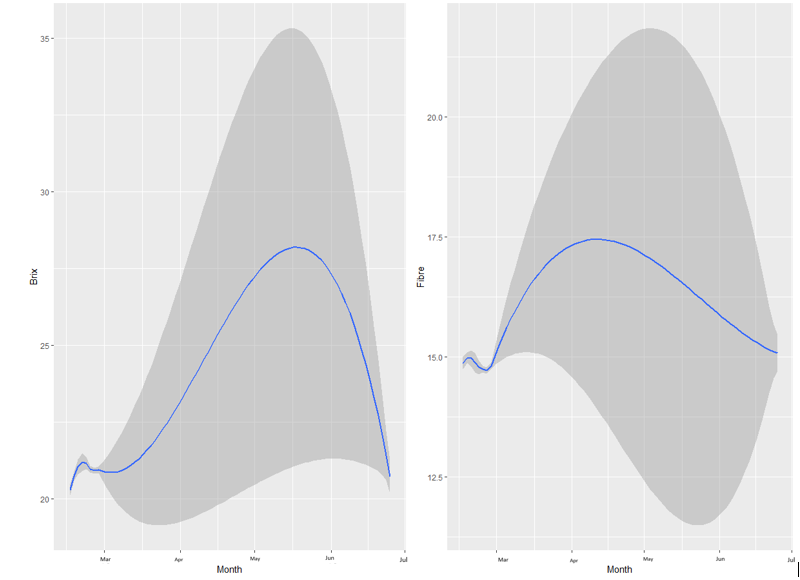
Table 37 The factory manager would like to know which of these variable(s) could be used to predict the features, **Brix** and **Fibres**. Use plotting tools from ggplot2 to justify your answer. Show your working (code) in R. Marks (4)

***Code:*** line1 = ggplot(data = NIR\_Final, mapping = aes(DateTime, NIR\_Brix)) + geom\_smooth(mapping = aes(DateTime, NIR\_Brix)) + labs(x = "Month", y="Brix")

line2 = ggplot(data = NIR\_Final, mapping = aes(DateTime, NIR\_Fibre)) + geom\_smooth(mapping = aes(DateTime, NIR\_Fibre)) + labs(x = "Month", y="Fibre")

gridExtra::grid.arrange(line1,line2,ncol = 2)

***Plot:***



As show in the plot, that between May to June, the sugarcane have the highest Brix and between April to May, the sugarcane have the highest Fibre compare with others, this suggest that in the future that factory manager would be tall the company to put more worker during these months.