

champton

by Christopher Hampton

Submission date: 10-May-2024 02:29PM (UTC+0100)

Submission ID: 233136924

File name: 32525_Christopher_Hampton_champton_565887_1349592784.docx (607.92K)

Word count: 2039

Character count: 11009

Trinity Saint David University of Wales

Statistical Programming (ACCB5021)

Author:
Chris Hampton
2116517@student.uwstd.ac.uk

Tutor:
Dr. Nik Whitehead
nicola.whitehead@uwtsd.ac.uk

10/05/2024

Contents

Table of Figures.....	2
1. Introduction	4
2. Flow Diagram	4
2.1 Prerequisites	4
3. Top 10 Comparison	5
3.1 Flow Diagram	5
3.2 Code Review.....	6
3.2.1 Output.....	7
3.2.2 Validation of Results	8
3.3 AI Interpretation	9
4. Coefficient of Correlation.....	9
4.1 Flow Diagram	9
4.2 Code Review.....	10
4.2.1 Output.....	11
4.2.2 Validation of Results	11
4.3 AI Interpretation	12
5. T-Test	13
5.1 Flow Diagram	13
5.2 Code Review.....	14
5.2.1 Output.....	15
5.2.2 Validation of Results	15
5.3 AI Interpretation	15
6. Conclusion.....	16
6.1 Further Steps.....	16
6.1 Personal Reflection	16
References	16

Table of Figures

Figure 1 - Flow Diagram – Overall	4
Figure 2 – Prerequisites	5
Figure 3 - File import.....	5
Figure 4 - Flow Diagram - Top 10 comparison	6
Figure 5 - Top 10 filtered.....	7
Figure 6 - Top 10 output	8

Figure 7 - Top 10 validation 8

Figure 8 - Flow Diagram - Correlation 10

Figure 9 - Output – Correlation 11

Figure 10 – Correlation. Source: - <https://www.scribbr.com/statistics/correlation-coefficient/> 11

Figure 11 - Validation summarise height 12

Figure 12 - Validation summarise length 12

Figure 13 - Flow Diagram - T-Test 14

Figure 14 - Output - T-Test 15

Figure 15 - Validation T-Test 15

Figure 16 - Validation - R T-Test 15

1. Introduction

Using R programming language and RStudio as the IDE perform data analysis tasks using the rollercoaster dataset. These tasks include creating visualisations, coefficient correlations, and T-Tests to gain insights within the dataset.

2. Flow Diagram

This flow diagram shows the over-arching flow of the R script.

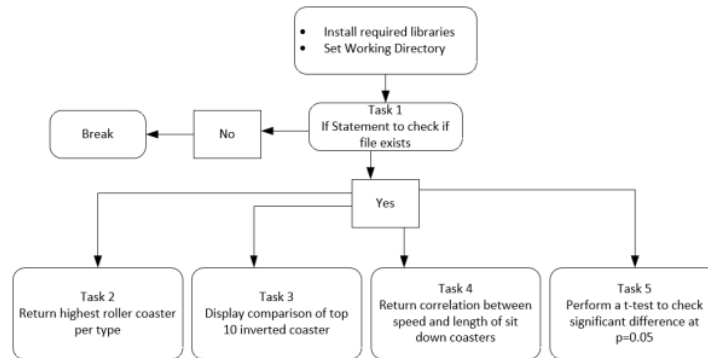


Figure 1 - Flow Diagram – Overall

Before initialising the tasks prerequisites need to be implemented to ensure all libraries and working directories are set accordingly.

2.1 Prerequisites

Initially, to ensure that libraries are installed within the RStudio environment and if statement and require function is used to check if packages are installed. If packages are not installed RStudio will install then using the 'install.packages' command. The two libraries used for the script are 'rstudioapi' and 'ggplot2'.

The 'rstudioapi' library is used to determine the user's current directory path. As the script uses an outside data source, in this case a csv file, the file needs to be within the folder structure of the script. By using the 'getActiveDocumentContext' function the current directory path can be obtained [1]. This can be set as the working directory using the 'setwd' command.

The 'ggplot2' [2] library is used to create the box chart for task three. The 'ggplot2' library has a more extensive options for creating visualisations in R compared to the built in functionality.

```

#Check to see if packages are already installed
#If packages are not installed - install packages
if (!require("rstudioapi")) {
  install.packages("rstudioapi")
}
if (!require("ggplot2")) {
  install.packages("ggplot2")
}

#import libraries
library(rstudioapi)
library(ggplot2)

#get the current path of the working directory
#this is to ensure that the data file and script are in the same folder path
current_path = rstudioapi::getActiveDocumentContext()$path

#set working directory to the current path
setwd(dirname(current_path))

```

Figure 2 – Prerequisites

To ensure that a file is present within the working directory an if statement is used to check. If the file does not exist, the console will print out letting the user know that the file is not present. The script will stop any further execution using the 'break' command.

```

if(!file.exists("roller_coasters.csv")) {
  print("No CSV file found")
  break
} else {
  coasterData <- data.frame(read.csv("roller_coasters.csv",header=T, colClasses=c(material_type="factor",seating_type="factor",manufacturer="factor",status="factor"))
  coasterData
}

```

Figure 3 - File import

If the file is present, the data from the csv will be imported into a dataframe, also, by specifying the material_type, seating_type, manufacturer, and status fields are imported as factors.

3. Top 10 Comparison

This task involved creating a function to create a box chart to show the top 10 inverted roller coasters based on their speed.

3.1 Flow Diagram

The flow diagram shows the steps needed to output the correct visualisation for the top 10 inverted coasters.

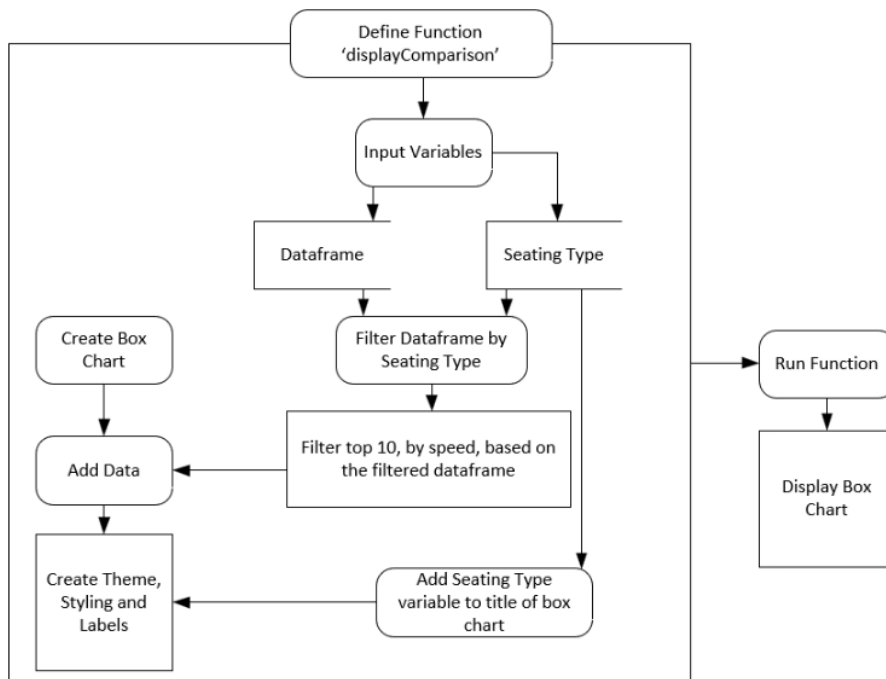


Figure 4 - Flow Diagram - Top 10 comparison

3.2 Code Review

For the 'displayComparison' function several filters is used to select the correct data for the visualisation.

Firstly, by filtering the whole dataset to the desired 'seating_type'. In this case 'Inverted' rollercoasters.

The following code snippet will filter the data based on the seating_type variable:

```
df <- df[df$seating_type==seating_type,]
```

The next filter orders the data by speed decreasing and limits the data to 10. This is denoted by **[1:10,]** appended to the filter.

```
df <- df[order(df$speed, decreasing = TRUE), ][1:10, ]
```

The results of this filter are displayed below:

	name	material_type	seating_type	speed	height	length	num_inversions	manufacturer
park	status							
250	Volcano, the Blast Coaster	Steel	Inverted	112	47	840	4	Intamin
ominion	status.closed.definitely							
1726	Banshee	Steel	Inverted	110	51	1260	7	B&M
Island	status.operating							
1909	Steel Venom	na	Inverted	110	56	192	0	Intamin
leyfair	status.operating							
217	Alpengeist	Steel	Inverted	108	60	NA	6	B&M
amsburg	status.operating							
2660	Quantum Leap	Steel	Inverted	105	58	367	3	Vekoma
hi-Park	status.operating							
76	Katun	Steel	Inverted	104	50	1200	6	B&M
llandia	status.operating							
130	Jubilee Odyssey	Steel	Inverted	100	51	891	5	Vekoma
ldmells	status.operating							
512	Pyrenees	Steel	Inverted	100	45	1234	6	B&M
village	status.operating							
300	Afterburn	Steel	Inverted	100	34	900	6	B&M
owinds	status.operating							
1951	Blue Harvest	Steel	Inverted	100	34	NA	0	B&M
ha Lund	status.announced							

Figure 5 - Top 10 filtered

When plotting the chart, several aesthetic commands are used to style the chart. For example:

```
theme(
  plot.title = element_text(color="Black", size=12, face="bold", hjust=0.5),
  plot.subtitle = element_text(color="red",size=14,face="bold",hjust=0.5),
  axis.title.x = element_text(color="#993333", size=12),
  axis.title.y = element_text(color="#993333", size=12)
)
```

The code snippet above shows how the labels and titles can be styled using the 'theme' command of 'ggplot2' library.

Finally, to add labels dynamically, the 'labs' function can be used.

```
labs(
  title="Top 10 Roller Coasters by speed \n where the seat type is:",
  subtitle=seating_type,
  x="Speed",
  y="Roller Coaster"
)+
```

By using the variable, 'seating_type', that has been passed into the function, the subtitle can be dynamically set.

```
displayComparison(coasterData,"Inverted")
```

By calling the function, as above, the user can change the 'seating_type, with the second variable that is passed in.

3.2.1 Output

The output from the function is shown below:

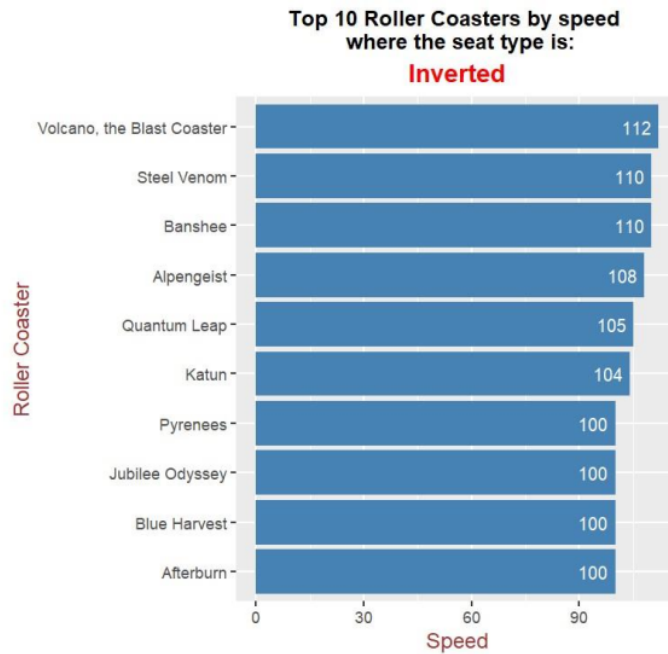


Figure 6 - Top 10 output

The output has used the styling that has been selected in the script and also dynamically set the subtitle to the user's input.

3.2.2 Validation of Results

To validate the data, to ensure integrity, the following pivot table shows the top 10, by speed, inverted rollercoasters.

seating_type	Inverted
Row Labels	Max of speed
Volcano, the Blast Coaster	112
Banshee	110
Steel Venom	110
Alpengeist	108
Quantum Leap	105
Katun	104
Afterburn	100
Jubilee Odyssey	100
Blue Harvest	100
Pyrenees	100
Grand Total	112

Figure 7 - Top 10 validation

3.3 AI Interpretation

Using Microsoft's Co-Pilot this shows the explanation of code for the function of the display comparison.

The displayComparison function in the R script is designed to create a bar chart that displays the top 10 fastest roller coasters of a specific seating type. Here's a step-by-step explanation of what the function does:

1. **Function Definition:** displayComparison is defined with two parameters: df for the dataframe and seating_type for the type of seating arrangement you want to filter by.
2. **Data Filtering:** The dataframe df is filtered to include only rows where the seating_type matches the specified variable. This subset is stored back into df.
3. **Ordering and Slicing:** The filtered dataframe is then ordered by the speed column in decreasing order, and only the top 10 rows are selected.
4. **Plotting with ggplot2:** The ggplot function is used to initiate the plotting of the data. The aes function defines speed as the x-axis and name as the y-axis, with the names reordered based on speed to ensure proper ordering in the plot.
5. **Bar Chart Creation:** geom_bar is used to create the actual bar chart, setting the bars to be filled with a steel blue color.
6. **Adding Text Labels:** geom_text adds text labels to the bars, which display the speed values. These are positioned inside the bars with slight adjustments for visibility.
7. **Labels and Titles:** labs is used to add a title, subtitle (which dynamically displays the seating_type), and axis labels to the chart.
8. **Styling:** The theme function customizes the appearance of the plot's text elements, such as the title, subtitle, and axis titles.
9. **Function Execution:** Finally, the function is called with coasterData (the main dataframe containing roller coaster information) and "Inverted" as arguments to display the top 10 fastest inverted roller coasters.

Here's how the function would look visually when executed:

[Bar Chart]

Title: Top 10 Roller Coasters by speed where the seat type is: Inverted

X-axis: Speed

Y-axis: Roller Coaster Names

[Each bar represents a roller coaster, ordered by speed, with the fastest at the top]

This function is a great example of how to use ggplot2 for data visualization in R, particularly for creating informative and styled bar charts. Remember to replace coasterData with your actual dataframe name when using this function in your own analysis.

The explanation from AI is fairly accurate. The explanation is comprehensive and can understand the code.

4. Coefficient of Correlation

This task involves determining the mean height and length of roller coasters where the type is 'Sit Down'. Also, to include the coefficient of correlation of length and speed.

The correlation between two values can be shown to on a linear scale. The values of the correlation can be between -1 and 1.

- -1 shows that there is that when one value increases the other value will decrease proportionately.
- 0 shows that there is no correlation between the two values.
- 1 shows that as one value increases/decrease then the other values will increase/decrease at the same rate.

4.1 Flow Diagram

The flow diagram shows the steps needed to perform the task.

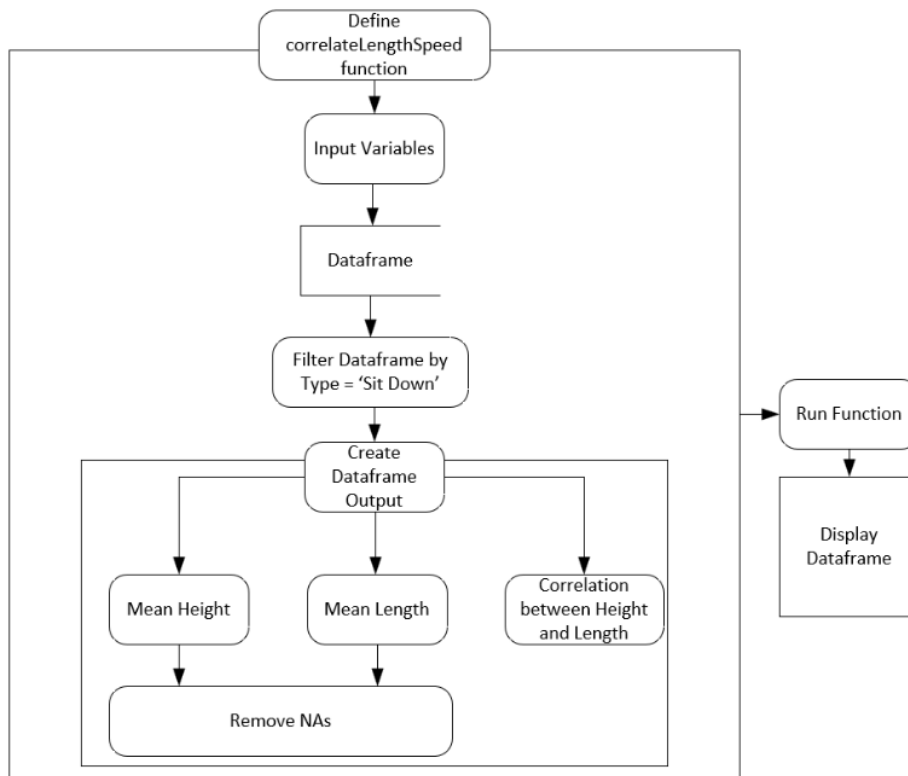


Figure 8 - Flow Diagram - Correlation

4.2 Code Review

As with 'displayComparison' function the 'correlateLengthSpeed' shares some initial steps. Firstly, to filter the data to show 'Sit Down' rollercoasters.

```
sit_down_coasters <- df[df$type == 'Sit Down',]
```

To calculate the mean height and length of the 'Sit Down' coasters, the in-built R function can be used. In this case the 'mean()' function.

The two following code snippets use the mean function to calculate the mean value based on the height and length. By including 'is.na()' function will exclude and NA values in the column. *N.b. the exclamation mark prepending the is.na() function will return any values that are not true for is.na().*

```
Mean_Height = mean(sit_down_coasters[!is.na(sit_down_coasters$height),]$height)
```

```
Mean_Length = mean(sit_down_coasters[!is.na(sit_down_coasters$length),]$length)
```

The correlation between speed and length can be calculated using the 'cor()' function in R.

```
Correlation_Length_Speed = cor(df$length, df$speed, use="pairwise.complete.obs")
```

The code snippet shows the `cor()` function accepting two factors following by in-built variables. The 'pairwise.complete.obs' variable is used to let the function know that there are NA values in the factors, and these can be removed and discounted from the calculation.

4.2.1 Output

The output produced by the function is shown below:

```
Mean_Height Mean_Length Correlation_Length_Speed
1      26.72585      606.1475              0.6648729
> |
```

Figure 9 - Output – Correlation

As seen from the above snippet, the correlation is 0.6648729. Reviewing the table below it can be inferred that Correlation coefficients is strong and positive. i.e. the length of a sit-down roller coaster is correlated to the speed. The hypothesis of the longer the rollercoaster the higher the speed can be correctly assumed.

Correlation coefficient	Correlation strength	Correlation type
-.7 to -1	Very strong	Negative
-.5 to -.7	Strong	Negative
-.3 to -.5	Moderate	Negative
0 to -.3	Weak	Negative
0	None	Zero
0 to .3	Weak	Positive
.3 to .5	Moderate	Positive
.5 to .7	Strong	Positive
.7 to 1	Very strong	Positive

Figure 10 – Correlation. Source: - <https://www.scribbr.com/statistics/correlation-coefficient/>

4.2.2 Validation of Results

By using the summarise functionality in excel, validation can be performed from the results. As can be shown the mean value shown below is very similar to the mean produced within the function. The slight differences could be down the rounding and how each programme handles numerical values.

Column1	
Mean	26.73529412
Standard Error	0.857947542
Median	23
Mode	13
Standard Deviation	35.01855591
Sample Variance	1226.299258
Kurtosis	468.1966931
Skewness	18.98417302
Range	902
Minimum	0
Maximum	902
Sum	44541
Count	1666

Figure 11 - Validation summarise height

Column1	
Mean	606.1474627
Standard Error	9.623054905
Median	500
Mode	420
Standard Deviation	393.8404957
Sample Variance	155110.3361
Kurtosis	1.639023668
Skewness	1.060764432
Range	2921
Minimum	-1
Maximum	2920
Sum	1015297
Count	1675

Figure 12 - Validation summarise length

4.3 AI Interpretation

Using Microsoft's Co-Pilot this shows the explanation of code for the function of the correlateLegnthSpeed.

Again, as with the Top 10 comparison, the AI functionality has interpreted the code fairly accurately.

This R code defines a function called `correlateLengthSpeed` that accepts a dataframe `df` as an argument and performs several operations:

1. It filters the dataframe to only include rows where the `type` column is 'Sit Down'. This filtered dataframe is stored in the variable `sit_down_coasters`. However, this filtered dataframe is not used in the rest of the function.
2. It calculates the mean height of the coasters, excluding any rows where the `height` is NA (not available). This value is stored in the `Mean_Height` column of the returned dataframe.
3. It calculates the mean length of the coasters, excluding any rows where the `length` is NA. This value is stored in the `Mean_Length` column of the returned dataframe.
4. It calculates the correlation between the `length` and `speed` of the coasters, excluding any pairs of values where either is NA. This value is stored in the `Correlation_Length_Speed` column of the returned dataframe.

The function returns a dataframe with these three calculated values.

Finally, the function `correlateLengthSpeed` is called with `coasterData` as the argument. The result of this function call would be a dataframe with the mean height, mean length, and correlation between length and speed of 'Sit Down' type coasters in the `coasterData` dataframe. Please note that the variable `sit_down_coasters` is defined but not used in the calculations. If the intention was to perform these calculations only for 'Sit Down' type coasters, you might need to replace `df` with `sit_down_coasters` in the `mean` and `cor` function calls.

5. T-Test

A t-test is used to quantify the differences between the mean average of two variables. In this case the average speed of a wooden roller coaster compared to a steel rollercoaster.

5.1 Flow Diagram

The flow diagram shows the steps needed to perform the task.

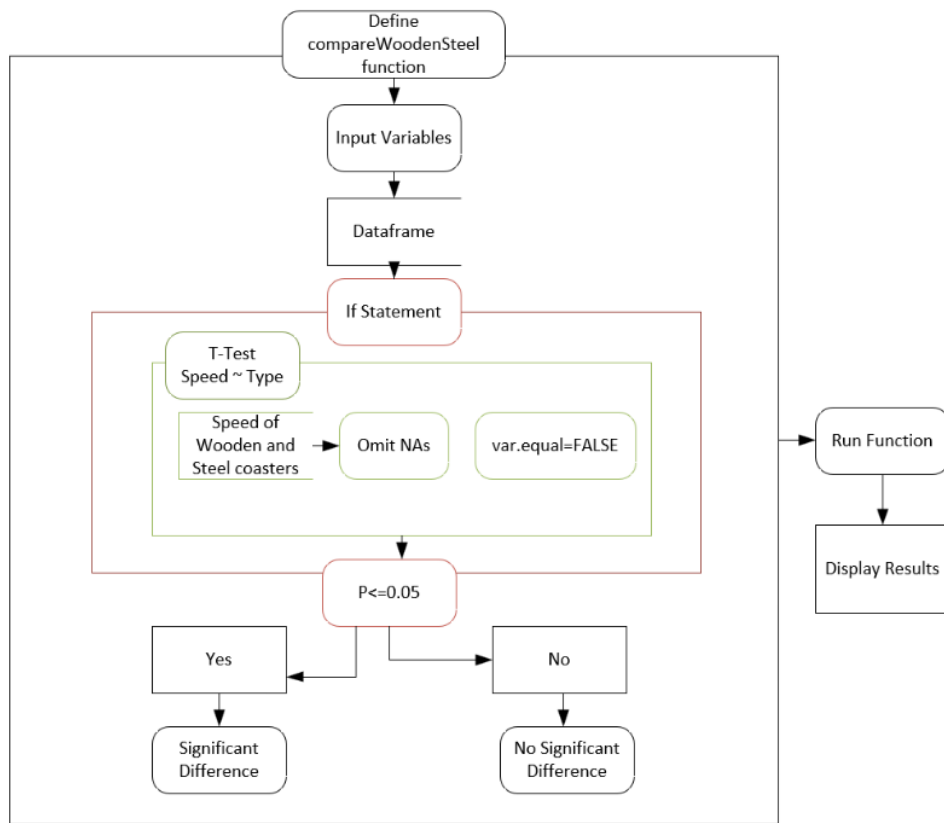


Figure 13 - Flow Diagram - T-Test

5.2 Code Review

R has an inbuilt function for T-Tests. This will be utilised for this function.

```
t.test(speed ~ type, na.omit( data.frame( )),var.equal=FALSE)$p.value
```

The code snippet above will determine the p.value between speed and type. Speed and Type are defined in the data variable as below:

```
type = rep(c("Steel", "Wooden")),  
speed =(df[df$material_type=="Steel"],)$speed,df[df$material_type=="Wooden"],$speed)
```

The dataframe is filtered using the [] and denoting the values in which the filter needs to apply. In the case above 'material_type' is equal to 'Steel' and 'Wooden'.

The T-Test is wrapped around an if statement to meet to requirements of the p value at 0.05 to accept or reject the hypothesis of any significance between a steel rollercoaster and a wooden rollercoaster.

```
if([t-test p.value] <= 0.05) { result <- "Significant Difference" }  
else {result <- "No Significant Difference"}
```

By including the **'var.equal=FALSE'** flag in the T-Test a Welch T-Test can be performed.

5.2.1 Output

The output shows that there is no significant difference between wooden coasters and steel coasters. This can prove the hypothesis that regardless of the material used to create the rollercoaster there is no bearing on the average speed that can be achieved.

```
> # run 'compareWoodenSteel' function with coasterData as a variable
> compareWoodenSteel(coasterData)
[1] "No Significant Difference"
```

Figure 14 - Output - T-Test

5.2.2 Validation of Results

Using the T-Test data analysis tool in excel, a T-Test can be performed to compare to the function in R.

t-Test: Two-Sample Assuming Unequal Variances		
	Steel	Wooden
Mean	68.88330494	82.30718954
Variance	832.09464	299.1747506
Observations	1174	153
Hypothesized Mean Difference	0	
df	277	
t Stat	-8.224284181	
P(T<=t) one-tail	3.84774E-15	
t Critical one-tail	1.650373154	
P(T<=t) two-tail	7.69549E-15	
t Critical two-tail	1.968565046	

Figure 15 - Validation T-Test

```
Welch Two Sample t-test
data: speed by type
t = 0.17257, df = 1313, p-value = 0.863
alternative hypothesis: true difference in means between group Steel and group wooden is not equal to 0
95 percent confidence interval:
 -2.763388  3.296437
sample estimates:
mean in group Steel mean in group Wooden
 70.56682          70.30030
```

Figure 16 - Validation - R T-Test

As can be seen from the two results shown above, there is vast disparity between the two. Further reflection can be used to understand how each of these functions are used and whether the input to the functions is aligned or whether the outputs are calculated differently.

5.3 AI Interpretation

Using Microsoft's Co-Pilot this shows the explanation of code for the function of the `compareWoodenSteel`.

The code you've shared is written in R and it defines a function `compareWoodenSteel` that performs a t-test on the `speed` of two types of coasters: `Steel` and `Wooden`. The function takes a dataframe `df` as input, which should contain a column `material_type` with values `Steel` or `Wooden`, and a column `speed` with the corresponding speeds.

Here's a brief overview of what the function does:

1. It creates a new dataframe with `type` and `speed` columns, where `type` is either `Steel` or `Wooden`, and `speed` is the corresponding speed from the input dataframe `df`.
2. It performs a t-test on the `speed` grouped by `type`. The t-test is used to determine if there is a significant difference between the means of two groups.
3. If the p-value from the t-test is less than or equal to 0.05, it returns the string "Significant Difference", indicating that there is a statistically significant difference in speed between the `Steel` and `Wooden` coasters. If the p-value is greater than 0.05, it returns "No Significant Difference".
4. Finally, the function is called with `coasterData` as the input dataframe.

Please note that the function assumes that the input dataframe `df` has the exact structure and column names as expected. If the column names or structure are different, the function may not work as expected.

Also, the function uses `var.equal = FALSE` in the t-test, which means it assumes unequal variances. If you have reason to believe the variances are equal, you might want to set this to `TRUE`.

Overall, the code is well-structured and follows good practices of commenting and readability. Great job! 👍

Again, the AI bot can interpret the code and give a useful synopsis of the function.

6. Conclusion

6.1 Further Steps

Further immediate steps that can be taken with the tasks above include reproducing the correlation validation within excel. By performing this the correlation can be validated and further analysis can be undertaken. The same can be performed for the T-Test. Due to the vast differences as mentioned above there are discrepancies in how each test is performed. By having these figures accurate and matching there can be a high level of confidence in the results outputted.

6.1 Personal Reflection

As with any projects that perform data and statistical analysis it is important to understand what analysis is being performed and what the benefits are. For example, a T-Test is a proven statistical calculation that is used, to better understand what it is used for can broaden the knowledge gained through the tasks and how they can apply to real life circumstances.

With the further development of generative AI, a lot can be learnt from it's uses. Instead of tentatively approaching the use of AI it can be embraced and used as a tool to enhance analytical subjects. As a reflection for the use of AI it can be sceptical. The code that has been fed into the chat bot contained comments and it can be suggested that AI is able to interpret the comments more precisely that the code to give a more enhanced answer. In any case the future is learning to adopt AI to better understand the subjects that are being researched.

References

- [1] J. A. H. W. G. R. Kevin Ushey, "Retrieve Information about an RStudio Editor," Github, [Online]. Available: <https://rstudio.github.io/rstudioapi/reference/rstudio-editors.html>. [Accessed 05 2024].
- [2] Rdocumentation, "ggplot2," datacamp, [Online]. Available: <https://www.rdocumentation.org/packages/ggplot2/versions/3.5.0>. [Accessed 05 2024].

FINAL GRADE

GENERAL COMMENTS

86/100

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

PAGE 15

PAGE 16

PAGE 17

