

Stephen Protheroe-Hewitt Coursework - Submission

by Steve Protheroe-Hewitt

Submission date: 03-May-2024 11:56PM (UTC+0100)

Submission ID: 232165975

File name: 32355_Steve_Protheroe-Hewitt_Stephen_Protheroe-Hewitt_Coursework_-_Submission_565887_1462298298.docx (775.71K)

Word count: 1413

Character count: 8591

Statistical Programming
(ACCB5021)

Data Analysis

Stephen Protheroe-Hewitt – 2112883

Data and Information Systems

Contents

List of Figures/Tables	3
Introduction.....	4
Code Documentation	4
Task Three	5
Task Four.....	7
Task Five	11
Reflection	15
Conclusion	16
References	17

List of Figures/Tables

Fig. 1 Task Three Flow Chart	5
Fig. 2 'displayComparison' Function Creation	6
Fig. 3 Top 10 Fastest Rollercoasters Bar Chart Code	6
Fig. 4 Top 10 Fastest Rollercoasters Bar Chart	7
Fig. 5 'roller_coasters.csv' File Highlighting Top 10 Rollercoasters	7
Fig. 6 Task Four Flow Chart	8
Fig. 7 'correlateLengthSpeed' Function Creation	9
Fig. 8 Mean, Correlation Coefficient and 'lengthSpeed' Data Frame Calculations and Creation	9
Fig. 9 Mean Height, Mean Length and Speed and Length Correlation Results	10
Fig. 10 'roller_coasters.csv' Mean Height.....	10
Fig. 11 'roller_coasters.csv' Mean Length	10
Fig. 12 'roller_coasters.csv' Calculated Speed and Length Correlation Coefficient	10
Fig. 13 Task Five Flowchart	11
Fig. 14 'compareWoodenSteel' Function Creation	12
Fig. 15 T-Test, Hypotheses and Results Creation.....	12
Fig. 16 'If Else' Statement to Display T-Test Conclusion	13
Fig. 17 T-Test Result.....	13
Fig. 18 'roller_coasters.csv' T-Test Calculation.....	14

Introduction

This report explains how created code works for tasks three to five, and how each code was evaluated to work correctly, while reflecting on challenges experienced, what was learned, what could be done differently, and how AI technology supported codes. The report concludes with a conclusion and future research to enhance codes.

Code Documentation

The documentation is for tasks three to five only. All function areas in flow charts are highlighted green.

Task Three

The flow chart is as follows.

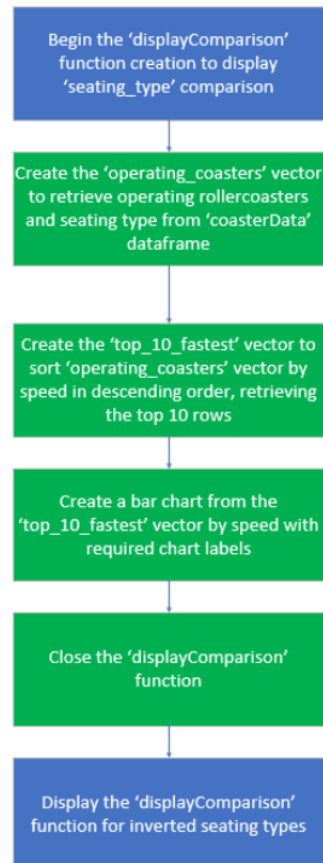


Fig. 1 Task Three Flow Chart

The code works as follows.

- 'displayComparison' function created to begin chart creation for 'seating_type' comparison.

- 'operating_coasters' vector created within function to show operating rollercoasters and required 'seating_type' from 'coasterData' data frame.
- 'top_10_fastest' vector created within function, sorting 'speed' data via 'order' (with '-' indicating descending order) and 'head' extracting first 10 rows.

```
# Task Three

# Creating the 'displayComparison' function to display the 'seating_type'
# comparison

displayComparison <- function(seating_type) {
  # Filtering the operating rollercoasters by seating type
  operating_coasters <- coasterData[coasterData$status == 'status.operating' & coasterData$seating_type == seating_type, ]
  # Sorting the operating rollercoasters by 'speed' and selecting the top 10
  top_10_fastest <- head(operating_coasters[order(-operating_coasters$speed), ], 10)
  # Creating a bar chart of the fastest 10 rollercoasters by speed
}
```

Fig. 2 'displayComparison' Function Creation

- Using this data, the chart creation code shows:
 - `barplot(top_10_fastest$speed)` – Creates bar chart and plots top 10 fastest rollercoasters.
 - `horiz = FALSE` – Creates vertical bar chart.
 - `col = "skyblue"` – Fills bars with sky-blue colour.
 - `main = paste("Top 10 Fastest", seating_type, "Roller Coasters")` – Titles bar chart with user preference.
 - `xlab = "Roller Coaster Name"` – Labels horizontal axis.
 - `ylab = "Speed"` – Labels vertical axis.

```
# Creating a bar chart of the fastest 10 rollercoasters by speed
barplot(top_10_fastest$speed,
        names.arg = top_10_fastest$name,
        horiz = FALSE,
        col = "skyblue",
        main = paste("Top 10 Fastest", seating_type, "Roller Coasters"),
        xlab = "Roller Coaster Name",
        ylab = "Speed")
}
```

Fig. 3 Top 10 Fastest Rollercoasters Bar Chart Code

- 'displayComparison' function finished and displayed using 'inverted' rollercoasters.

The resulting bar chart is shown.

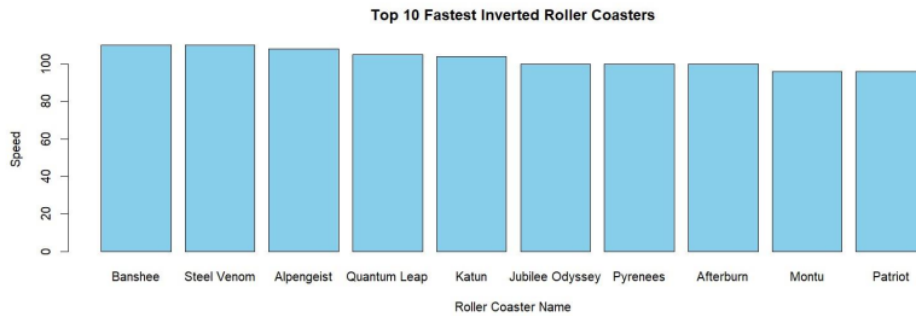


Fig. 4 Top 10 Fastest Rollercoasters Bar Chart

The code was evaluated by comparing data held in 'roller_coasters.csv' and filtering data to show operating inverted rollercoasters, then sorting the speed column from largest to smallest. When compared, the data between 'roller_coasters.csv' file and bar chart was confirmed correct.

	A	B	C	D	J	K
1	name	material	seating	speed	status	
102	Banshee	Steel	Inverted	110	status.operating	
112	Steel Venom	na	Inverted	110	status.operating	
114	Alpenggeist	Steel	Inverted	108	status.operating	
151	Quantum Leap	Steel	Inverted	105	status.operating	
166	Katun	Steel	Inverted	104	status.operating	
179	Afterburn	Steel	Inverted	100	status.operating	
195	Jubilee Odyssey	Steel	Inverted	100	status.operating	
202	Pyrenees	Steel	Inverted	100	status.operating	
251	Montu	Steel	Inverted	96	status.operating	
252	Patriot	Steel	Inverted	96	status.operating	
268	Great Bear	Steel	Inverted	93	status.operating	
270	Talon	Steel	Inverted	93	status.operating	
281	Raptor	Steel	Inverted	92	status.operating	
284	The Monster	Steel	Inverted	92	status.operating	
289	Oziris	Steel	Inverted	91	status.operating	
295	Anaconda	Steel	Inverted	90	status.operating	
315	Phaethon	Steel	Inverted	90	status.operating	
344	Fly the Great Nor'Easter	Steel	Inverted	89	status.operating	
356	Silver Bullet	Steel	Inverted	89	status.operating	
367	Golden Wings in Snowfield	Steel	Inverted	88	status.operating	
368	Hanging Roller Coaster	na	Inverted	88	status.operating	
370	Kumali	Steel	Inverted	88	status.operating	
374	Snow Mountain Flying Dragon	Steel	Inverted	88	status.operating	
385	Happy Angel	na	Inverted	87	status.operating	
411	DrageKongen	Steel	Inverted	85	status.operating	
456	Jaguar	Steel	Inverted	83	status.operating	

Fig. 5 'roller_coasters.csv' File Highlighting Top 10 Rollercoasters

Task Four

The flow chart is as follows.

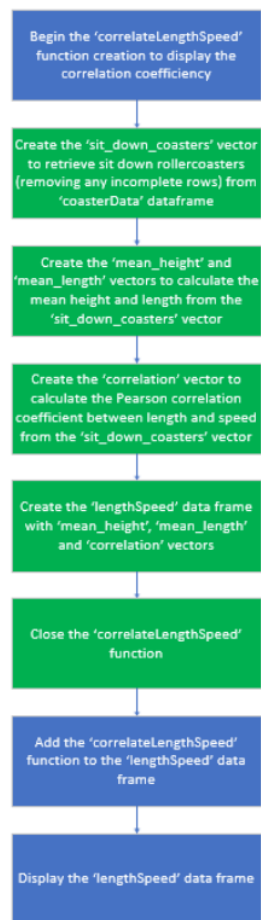


Fig. 6 Task Four Flow Chart

The code works as follows.

- 'correlateLengthSpeed' function created to begin calculation of mean height and length, and correlation coefficient between length and speed.
- 'sit_down_coasters' vector created within function, showing sit-down rollercoasters, and removing any incomplete rows using 'complete.cases' function.

```

# Task Four

# Creating the 'correlateLengthSpeed' function to calculate the correlation
# between length and speed

correlateLengthSpeed <- function(correlation) {

  # Filtering for sit down rollercoasters only, removing any NA fields

  sit_down_coasters <- coasterData[coasterData$seating_type == 'Sit Down', ]
  sit_down_coasters <- sit_down_coasters[complete.cases(sit_down_coasters), ]

```

Fig. 7 'correlateLengthSpeed' Function Creation

- 'mean_height' and 'mean_length' vectors created to calculate mean height and length from 'sit_down_coaster' vector.
- 'correlation' vector created to calculate correlation coefficient between length and speed.
- 'lengthSpeed' data frame created to include mean height and length, and correlation coefficient between length and speed.

```

# Calculating the mean height and length

mean_height <- mean(sit_down_coasters$height)
mean_length <- mean(sit_down_coasters$length)

# Calculating the correlation coefficient between length and speed

correlation <- cor(sit_down_coasters$length, sit_down_coasters$speed)

# Creating the 'lengthSpeed' dataframe

lengthSpeed <- data.frame(Mean_Height = mean_height,
                          Mean_Length = mean_length,
                          Correlation_Length_Speed = correlation)

}

```

Fig. 8 Mean, Correlation Coefficient and 'lengthSpeed' Data Frame Calculations and Creation

- 'correlateLengthSpeed' function finished and added to 'lengthSpeed' data frame, with 'lengthSpeed' displayed.

The resulting calculations from 'lengthSpeed' data frame are below.

	Mean_Height	Mean_Length	Correlation_Length_Speed
1	28.83574	640.3419	0.6582279

Fig. 9 Mean Height, Mean Length and Speed and Length Correlation Results

The code was evaluated by comparing data held in 'roller_coasters.csv' and filtering data to show sit-down rollercoasters. The mean height and length, and the Pearson correlation coefficient (which R defaults to when using 'cor' [1]) were calculated to confirm correct calculations. However, while both figure sets are close, they are not exact and further work is required to understand calculation differences.

1	name	material	seating	speed	height	length	num_in	manufacturer	park	status
1260	Yankee Cannonball	Wooden	Sit Down	56	20	610	0	Philadelphia Toboggan Coaster	Canada's Wonderland	status.operating
1261	Yukon Quaiid	Steel	Sit Down	85	12	1000	0	Intamin	Le Pal	status.operating
1262	Yukon Striker	Steel	Sit Down	130	75	1105	4	B&M	Canada's Wonderland	status.operating
1263	Zach's Zoomer	Wooden	Sit Down	14	460	0	CCI	Michigan's Adventure	status.operating	
1264	Zadra	Hybrid	Sit Down	115	63	1300	3	RHC	Energylandia	status.construction
1265	Zatum	Steel	Sit Down	129	63	400	0	Intamin	Space World	status.closed.definitely
1266	ZweiStang	Steel	Sit Down	32	6	99	0	Zierer	Dippin'Dice FamilyPark	status.closed.definitely
1267	Zeus	Wooden	Sit Down	97	27	864	0	CCI	Mt. Olympus Water & Theme Park	status.operating
1268	ZigZag	Steel	Sit Down	45	14	370	0	Mack	Waligator Parc	status.closed.definitely
1269	Zigzag	Steel	Sit Down	40	13	270	0	Zamperla	Walibi Sud Ouest	status.operating
1270	Zippin Pippin	Wooden	Sit Down	68	21	715	0	Gravity Group	Bay Beach Amusement Park	status.operating
1271	Zombie Ride	na	Sit Down	99	46			Premier Rides	Boisjeu M&Gico	status.operating
1272	Zonga	Steel	Sit Down	87	35	1120	4	Schwarzkopf	Six Flags Discovery Kingdom	status.closed.definitely
1273	Zoomerang	Steel	Sit Down	75	35	285	3	Vekoma	Lake Compounce	status.operating
1274	Zoomerang	Steel	Sit Down	76	36	285	3	Vekoma	Alabama Adventure	status.closed.definitely
1275	Zydeco Scream	Steel	Sit Down	75	36	267	3	Vekoma	Six Flags New Orleans	status.closed.definitely
1276					26.53218					

Fig. 10 'roller_coasters.csv' Mean Height

1	A	B	C	D	E	F	G	H	I	J	
	name	material_type	seating	speed	height	length	num_in	manufacturer	park	status	
1233	X Coaster	Steel	Sit Down	105	46	150	2	Maurer	Magic Springs & Crystal Falls	status.operating	
1234	Xcelerator	Steel	Sit Down	122	62	671	0	Intamin	Knoct's Berry Farm	status.operating	
1235	Xpress 1 Platform 13	Steel	Sit Down	90	26	996	3	Vekoma	Walibi Holland	status.operating	
1236	X-Treme Racer	Steel	Sit Down	56	16	400	0	Mack	Legland Billund	status.operating	
1237	Yamaha Racing Coaster	Steel	Sit Down	100	50	275	1	Premier Rides	Trans Studio Indoor Theme Park	status.operating	
1238	Yankee Cannonball	Wooden	Sit Down	56	20	610	0	Philadelphia Toboggan Coaster	Canada Lake Park	status.operating	
1239	Yukon Quad	Steel	Sit Down	85	12	1000	0	Intamin	Le Pal	status.operating	
1240	Yukon Striker	Steel	Sit Down	130	75	1105	4	B&M	Canada's Wonderland	status.operating	
1241	Zach's Zoomer	Wooden	Sit Down	14	460	0	CCI	Michigan's Adventure		status.operating	
1242	Zadra	Hybrid	Sit Down	115	63	1300	3	RMC	Energylandia		status.construction
1243	Zatum	Steel	Sit Down	129	63	400	0	Intamin	Space World		status.closed.definitely
1244	ZweiStang	Steel	Sit Down	32	6	99	0	Zierer	Dippin'Dice FamilyPark		status.closed.definitely
1245	Zeus	Wooden	Sit Down	97	27	864	0	CCI	Mt. Olympus Water & Theme Park		status.operating
1246	Zig Zag	Steel	Sit Down	45	14	370	0	Mack	Waligator Parc		status.closed.definitely
1247	Zigzag	Steel	Sit Down	40	13	270	0	Zamperla	Walibi Sud-Ouest		status.operating
1248	Zippin Pippin	Wooden	Sit Down	68	21	715	0	Gravity Group	Bay Beach Amusement Park		status.operating
1249	Zonga	Steel	Sit Down	87	35	1120	4	Schwarzkopf	Six Flags Discovery Kingdom		status.closed.definitely
1250	Zoomerang	Steel	Sit Down	75	35	285	3	Vekoma	Lake Compounce		status.operating
1251	Zoomerang	Steel	Sit Down	76	36	285	3	Vekoma	Alabama Adventure		status.closed.definitely
1252	Zydeco Scream	Steel	Sit Down	75	36	267	3	Vekoma	Six Flags New Orleans		status.closed.definitely
1253						602.9658					

Fig. 11 'roller_coasters.csv' Mean Length

N1	A	B	C	D	E	F	G	H	I	J	K	L	M	N
	name	material	seating	speed	height	length	num_in	manufacturer	park	status			Correlation between speed and length	0.642585
1	10 Inverso Steel	Sit Down	72	30	850	10	Intamin	Chimelong	status.operating					
3	Abismo Steel	Sit Down	105	46	450	2	Maurer	Parque de	status.operating					
4	Abyss na	Sit Down	85	30	630	3	Gerstlauer	Adventure	status.operating					
5	Achterbahn Steel	Sit Down	32	6	199	0	Zierer	Amusement	status.operating					
6	Achterbahn Steel	Sit Down	50	15	560	0	Schwarzkopf	Skyline Pal	status.operating					
7	Aconagua Steel	Sit Down	93	36	751	0	Schwarzkopf	Parque de	status.closed.definitely					
8	Adrenaline Steel	Sit Down	72	22	320	3	Gerstlauer	Oaks Amu	status.operating					
9	Adventure Steel	Sit Down	56	19	903	0	Arrow	Kings Islan	status.operating					
10	African Spl Steel	Sit Down	30	7	113	0	SBF	Tatmania	status.operating					
11	Aftershock Steel	Sit Down	106	58	367	3	Vekoma	Silverwood	status.operating					
12	Alpengeist Steel	Sit Down	45	6	264	0	Mack	Europa Pal	status.operating					
13	Alpina Bah Steel	Sit Down	80	27	910	0	Schwarzkopf	Foire	status.operating					
14	Alpina Blitz Steel	Sit Down	100	30	800	0	Mack	Nigoland	status.operating					
15	Alpina Bob Steel	Sit Down	50	14	538	0	Schwarzkopf	Holly Park	status.closed.definitely					
16	Altair na	Sit Down	85	33	875	Intamin	CineCitta	1	status.operating					
17	Alucinakis Steel	Sit Down	30	8	250	0	Zamperla	Terra M&A	1	status.operating				
18	American i Wooden	Sit Down	106	38	1417	0	Intamin	Six Flags G	status.operating					
19	American i Wooden	Sit Down	77	25	827	0	GCI	Six Flags Si	status.operating					
20	Anaconda Wooden	Sit Down	85	35	1200	0	William J. t	Waligator	status.operating					

Fig. 12 'roller_coasters.csv' Calculated Speed and Length Correlation Coefficient

Task Five

The flow chart is as follows.

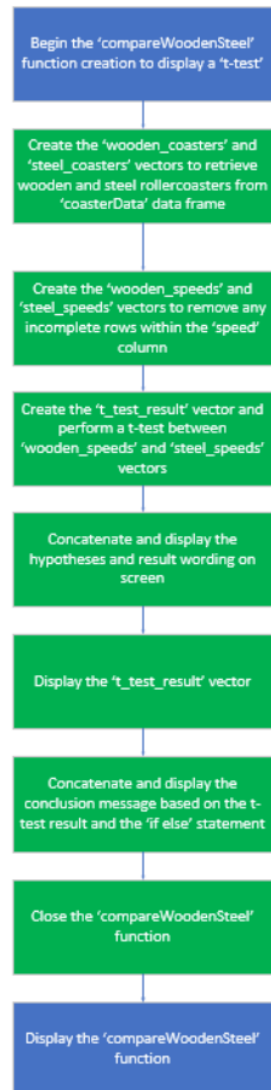


Fig. 13 Task Five Flowchart

The code works as follows.

- 'compareWoodenSteel' function created to begin t-test calculations of speed differences.
- 'wooden_coasters' and 'steel_coasters' vectors created within function to show required rollercoasters, removing incomplete rows in speed column using 'complete.cases' function.

```
# Task Five

# Creating the 'compareWoodenSteel' function to calculate a t-test between
# wooden and steel rollercoaster speeds

compareWoodenSteel <- function(t_test) {

  # Filtering for wooden and steel rollercoasters only

  wooden_coasters <- coasterData[coasterData$material_type == 'Wooden', ]
  steel_coasters <- coasterData[coasterData$material_type == 'Steel', ]

  # Removing any NA values within the 'speed' column

  wooden_speeds <- wooden_coasters$speed[complete.cases(wooden_coasters$speed)]
  steel_speeds <- steel_coasters$speed[complete.cases(steel_coasters$speed)]
```

Fig. 14 'compareWoodenSteel' Function Creation

- 't_test_result' vector created for results of t-test between 'wooden_speeds' and 'steel_speeds' vectors, with hypotheses to show any significant differences between wooden and steel rollercoaster speeds, and t-test results provided.

```
# Performing a t-test between wooden and steel rollercoaster speeds

t_test_result <- t.test(wooden_speeds, steel_speeds)

# Printing the hypotheses and results of the t-test

cat("Hypotheses:\n")
cat("H0: There is no significant difference between the speeds of
    wooden and steel roller coasters.\n")
cat("H1: There is a significant difference between the speeds of
    wooden and steel roller coasters.\n\n")

cat("T-test Results (p=0.05):\n")

# Displaying the results of the t-test

print(t_test_result)
```

Fig. 15 T-Test, Hypotheses and Results Creation

- 'If else' statement created, showing result message to display depending on returned p-value returned.
- 'compareWoodenSteel' function finished and displayed.

```

if (t_test_result$p.value < 0.05) {
  cat("\nConclusion: Reject the null hypothesis. There is a significant
      difference between the speeds of wooden and steel rollercoasters.\n")
}

else {
  cat("\nConclusion: Fail to reject the null hypothesis. There is no significant
      difference between the speeds of wooden and steel rollercoasters.\n")
}

}

# Displaying the results of the 'compareWoodenSteel' function

compareWoodenSteel()

```

Fig. 16 'If Else' Statement to Display T-Test Conclusion

The t-test result is as follows.

Hypotheses:

H0: There is no significant difference between the speeds of wooden and steel roller coasters.

H1: There is a significant difference between the speeds of wooden and steel roller coasters.

T-test Results (p=0.05):

Welch Two Sample t-test

data: wooden_speeds and steel_speeds

t = 8.2243, df = 277.44, p-value = 7.657e-15

alternative hypothesis: true difference in means is not equal to 0

95 percent confidence interval:

10.21077 16.63700

sample estimates:

mean of x mean of y

82.30719 68.88330

Conclusion: Reject the null hypothesis. There is a significant difference between the speeds of wooden and steel rollercoasters.

Fig. 17 T-Test Result

The code was evaluated by comparing data held in 'roller_coasters.csv' file, filtering the 'material_type' for wooden and steel rollercoasters, and copying data into a new worksheet. Any

blanks and zeroes within 'speed' column were removed like with the script. Due to unequal variances between both sets of data, the 't-test: Two-Sample Assuming Unequal Variances' analysis was performed, to which parts of data like degree of freedom and t Stat were similar but not exact, resulting in further work and investigations to understand calculation differences.

	A	B	C
1	t-Test: Two-Sample Assuming Unequal Variances		
2			
3		<i>Variable 1</i>	<i>Variable 2</i>
4	Mean	82.30718954	69.35591767
5	Variance	299.1747506	805.0019692
6	Observations	153	1166
7	Hypothesized Mean Difference	0	
8	df	274	
9	t Stat	7.962235224	
10	P(T<=t) one-tail	2.26238E-14	
11	t Critical one-tail	1.650433793	
12	P(T<=t) two-tail	4.52477E-14	
13	t Critical two-tail	1.968659628	

Fig. 18 'roller_coasters.csv' T-Test Calculation

Reflection

Creating code was complex due to function requirements, though easier than creating Pell codes due to understanding required data for each task, including analytical requirements like correlation between length and speed. There is confidence in the code for all tasks, though due to variances in sample data for the t-test, the Shapiro test (confirming normally distributed or skewed data [2]) should be used to identify and remove outliers.

There were two parts learned from code creation.

- Functions must be used to create code due to ease of use, once created, with single lines of code, improving readability and efficiency.
- Created code is more understandable as it is more readable than previous iterations, being more clearly defined to what each code area is doing within each task.

An area to approach differently is analysing and understanding why differences exist between data from 'roller_coasters.csv' and the code. While slight differences between data may not be significant, it is important to understand why differences occur if data accuracy and reliability are questioned.

ChatGPT [3] assisted with understanding code areas not clearly defined. For example, using 'complete.cases' removed incomplete rows, allowing code to be cleansed and negating 'na.rm=TRUE' requirements, while 'head' provides the first 6 rows by default unless specified.

ChatGPT also assisted in creating and understanding analytical calculations for tasks four and five, as required functions like 'mean', 'cor' and 't-test' could be described as to how they work, allowing data cleansing to occur, providing improved accuracy for further analysis.

Conclusion

Data analysis is vital in an evolving world of continuous information, allowing analysts to make informed decisions based on analysis, evaluation, and conclusion of data. If no effective analysis is performed, informed decisions cannot be made and no end value provided, as evidenced by Ma [4], whereby the UK Government of the late 1990s envisioned an integrated NHS with involvement from stakeholders to improve patient care and reduce costs, but resulted in continuation of the norm, with NHS expenditure based on old data and minimal analysis of measurable outcomes that improved patient care.

Future research involves improving code efficiency as while it provides the majority of the data, there are areas where code could be minimised and provide the same result (for example, combining mean calculations in task four, or combining the wooden and steel rollercoaster code to one or two lines) to improve readability, reliability and efficiency for data analysts, ensuring accurate data is retrieved for analysis.

References

- [1] A. Soetewey, "Correlation coefficient and correlation test in R", *Stats and R*, May 28, 2020
[Online] Available: <https://statsandr.com/blog/correlation-coefficient-and-correlation-test-in-r/#correlation-coefficient> [Accessed May 2, 2024].
- [2] A. Monter-Pozos and E. González-Estrada, "On testing the skew normal distribution by using Shapiro–Wilk test", *Journal of computational and applied mathematics*, vol. 440, p. 115649, 2024 [Online] Available: [10.1016/j.cam.2023.115649](https://doi.org/10.1016/j.cam.2023.115649) [Accessed May 3, 2024].
- [3] OpenAI, "ChatGPT", *OpenAI* [Online] Available: <https://chat.openai.com/auth/login>. [Accessed Apr. 1, 2024].
- [4] R. Ma, *Clinical costing techniques and analysis in modern healthcare systems*, Hershey, Pennsylvania: Medical Information Science Reference/IGI Global, 2019 [E-book].

Stephen Protheroe-Hewitt Coursework - Submission

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

/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