Clean and Report on a Dataset Students (Demographics and Personalities)

Pair Assignment

In the study program Global Business Germany (EBG4) and the module Data Analytics for Marketing Applications at DCU Business School

Submitted by: Linus Lehr (22108475) and Caroline Anna Marie Ziegler (22108564)

Programme: Global Business Germany (EGB4)

Module Code: CA259

Assignment Title: Clean and Report on a Dataset

Module Coordinator: Prof. Dr. Alan Smeton

Word count: 1,630

Submission Date: 14.02.2024

Declaration on Plagiarism Assignment Submission Form

We declare that this material, which we now submit for assessment, is entirely our own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my work. We understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should we engage in plagiarism, collusion or copying.

We have read and understood the Assignment Regulations. We have identified and included the source of all facts, ideas, opinions, and viewpoints of others in the assignment references. Direct quotations from books, journal articles, internet sources, module text, or any other source whatsoever are acknowledged, and the source cited are identified in the assignment references (note: No direct quotations are allowed for this essay).

This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study. We have read and understood the referencing guidelines found at http://www.dcu.ie/info/regulations/plagiarism.shtml, https://www4.dcu.ie/students/az/plagiarism and/or recommended in the assignment guidelines.

Name: Caroline Ziegler Date: 14.02.2024

Name: L. Lahr Date: 14.02.2024

Table of Contents

1. Data Normalisation and Imputation of Demographics

To facilitate the normalisation and imputation processes for the provided datasets, Python was employed as the analytics tool. Initially, the demographics dataset was imported into a Pandas Data Frame, comprising 122 rows and 16 columns. To gain comprehensive insights into the dataset, an initial assessment was conducted to identify missing values utilizing the *.isna().sum()* method in Python. This examination revealed the presence of 5 missing values within the dataset.

Detecting Missing Values

```
demographics.isna().sum()
```

```
Age (in years)                                  1
CAO Points (100 to 600)                         1
Daily travel to DCU (in km, 0 if on-campus)     1
Average year 1 exam result (as %)               1
Seat row in class                               1
Gender                                          0
Number of older siblings                        0
Number of younger siblings                      0
Old Dublin postcode (0 if outside Dublin)       0
Height (in cm)                                  0
Weight (in kg)                                  0
Eye colour                                      0
Hair colour                                     0
Last 4 digits of your mobile (0000 to 9999)     0
Star sign                                       0
Shoe size                                       0
dtype: int64
```

Notably, all 5 missing values were found to be located in the first 5 rows of the dataset. Consequently, the demographics dataset was divided into two distinct subsets. The first subset encompassed all rows devoid of missing values ("demographics_no_na"), while the second subset exclusively included the five rows containing missing values ("demographics_is_na"). This partitioning strategy was implemented to circumvent the process of imputing values based on previously imputed data, which may introduce heightened levels of error into the dataset. Indeed, imputation inherently entails a degree of uncertainty. Hence, imputing values based on prior imputations could exacerbate this uncertainty, thereby compromising the integrity of the dataset.

Creating the Subset with the Missing Values

```
demographics_is_na = demographics.iloc[:5, :].copy()
demographics_is_na
```

	Age (in years)	CAO Points (100 to 600)	Daily travel to DCU (in km, 0 if on-campus)	Average year 1 exam result (as %)	Seat row in class	Gender	Number of older siblings	Number of younger siblings	Old Dublin postcode (0 if outside Dublin)	Height (in cm)	Weight (in kg)	Eye colour	Hair colour	Last 4 digits of your mobile (0000 to 9999)	Star sign	Shoe size
0	21.0	NaN	10.0	66.0	4.0	Male	0	2	0	178.0	92.0	Green	Brown	5262	Capricorn	9.0
1	22.0	505.0	NaN	68.0	7.0	Female	0	1	0	155.0	55.0	Blue	Brown	7181	Leo	4.0
2	20.0	600.0	30.0	NaN	6.0	Female	1	2	0	180.0	55.0	Blue	Brown	7677	Leo	5.0
3	19.0	543.0	10.0	71.0	NaN	Male	0	1	13	187.0	76.0	Blue	Red	838	Taurus	9.0
4	NaN	407.0	8.0	73.0	1.0	Male	2	0	15	181.0	87.0	Blue	Brown	6290	Cancer	9.5

Creating the Subset with No Missing Values

```
demographics_no_na = demographics.iloc[5:, :].copy()
demographics_no_na
```

	Age (in years)	CAO Points (100 to 600)	Daily travel to DCU (in km, 0 if on-campus)	Average year 1 exam result (as %)	Seat row in class	Gender	Number of older siblings	Number of younger siblings	Old Dublin postcode (0 if outside Dublin)	Height (in cm)	Weight (in kg)	Eye colour	Hair colour	Last 4 digits of your mobile (0000 to 9999)	Star sign	Shoe size
5	20.0	484.0	0.0	68.0	1.0	Male	2	1	0	188.0	73.0	Blue	Blonde	6684	Taurus	9.5
6	21.0	498.0	2.0	82.0	2.0	Male	2	1	2	187.0	105.0	Brown	Brown	7835	Gemini	11.0
7	21.0	498.0	15.0	74.0	1.0	Male	1	1	15	186.0	80.0	Blue	Red	6738	Taurus	9.5
8	19.0	509.0	10.0	64.0	3.0	Male	0	0	3	187.0	68.0	Brown	Brown	608	Aquarius	10.0
9	19.0	566.0	2.0	70.0	10.0	Female	2	0	0	167.0	60.0	Green	Brown	87	Pices	5.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
117	21.0	565.0	7.0	71.0	3.0	Female	2	1	14	170.0	59.0	Blue	Black	1699	Libra	6.0
118	20.0	520.0	7.0	62.0	8.0	Female	0	2	7	165.0	70.0	Blue	Black	5289	Taurus	5.0
119	20.0	510.0	60.0	65.0	5.0	Male	1	1	0	184.0	85.0	Blue	Brown	5317	Libra	10.0
120	23.0	490.0	7.0	70.0	4.0	Female	0	3	7	173.0	73.0	Green	Blonde	4602	Aquarius	7.0
121	22.0	500.0	4.0	65.0	4.0	Female	0	3	0	177.0	73.0	Blue	Brown	6402	Leo	7.0

Furthermore, the *.describe()* method was employed to furnish a comprehensive overview and comprehension of the dataset, encapsulating essential statistical parameters such as count, mean, standard deviation, as well as minimum and maximum values, along with quartiles for each column. Noteworthy observations gleaned from this summary included the maximum values for age, recorded at 58 years, and daily travel to DCU, reaching 103 km. Subsequently, a more exhaustive analysis was undertaken for both columns, utilizing the *.unique()* and *.value_counts()* methods to delineate the unique values within each column and their respective frequencies. The occurrence of 58 years as the maximum age was deemed plausible, given that DCU accommodates individuals across various age demographics, including older adults pursuing educational endeavours. Therefore, this outlier in the age column remained unaltered in the dataset. However, the handling of the outlier in the daily travel column warrants a more nuanced discussion, which will be expounded upon in detail in the subsequent section pertaining to the imputation of missing values within the column.

Overview of the Demographics Dataset

```
round(demographics.describe(),2)
```

	Age (in years)	CAO Points (100 to 600)	Daily travel to DCU (in km, 0 if on-campus)	Average year 1 exam result (as %)	Seat row in class	Number of older siblings	Number of younger siblings	Old Dublin postcode (0 if outside Dublin)	Height (in cm)	Weight (in kg)	Last 4 digits of your mobile (0000 to 9999)	Shoe size
count	121.00	121.00	121.00	121.00	121.00	122.00	122.00	122.00	122.00	122.00	122.00	122.00
mean	22.20	490.71	10.13	68.59	5.67	1.23	0.91	5.28	176.27	72.29	4650.39	8.25
std	5.11	76.00	15.31	8.47	2.73	1.10	0.93	6.03	10.33	12.58	2873.11	2.40
min	18.00	130.00	0.00	42.00	1.00	0.00	0.00	0.00	150.00	51.00	59.00	3.00
25%	20.00	479.00	2.00	65.00	4.00	1.00	0.00	0.00	169.00	62.00	2228.75	6.00
50%	21.00	510.00	5.00	68.00	5.00	1.00	1.00	2.50	177.00	71.00	4641.00	9.00
75%	22.00	534.00	12.00	72.00	8.00	2.00	1.00	9.00	183.00	82.00	7164.50	10.00
max	58.00	600.00	103.00	99.00	12.00	7.00	4.00	24.00	198.00	105.00	9898.00	13.00

Understanding the Age Variable

```
demographics["Age (in years)"].unique()
```

```
array([21., 22., 20., 19., nan, 25., 23., 28., 36., 33., 24., 18., 32.,
       26., 47., 58.])
```

```
demographics["Age (in years)"].value_counts()
```

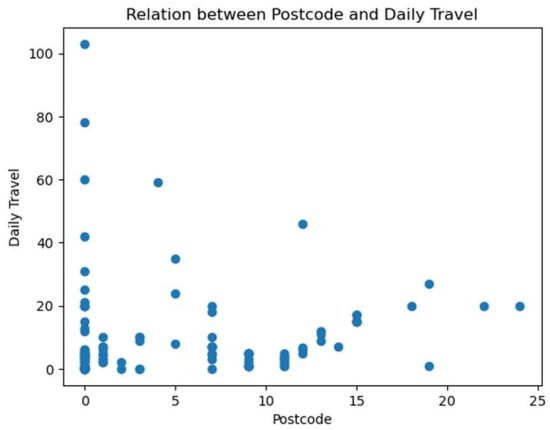
```
20.0    32
21.0    26
22.0    19
19.0    17
23.0     6
24.0     5
25.0     4
28.0     2
36.0     2
33.0     2
26.0     2
18.0     1
32.0     1
47.0     1
58.0     1
Name: Age (in years), dtype: int64
```

The five missing values were handled using various approaches, depending on the variable where the value was missing.

1. **Daily travel to DCU (in km, 0 if on-campus)**
   A logical connect was discovered between the daily travel distance to Dublin City University and the old Dublin postcode, as the proximity to the university campus is contingent upon the geographical location denoted by the postcode. To visualise this relationship, a scatterplot was employed to illustrate the association between these two variables. Furthermore, an exploration of unique values within the "Daily travel to DCU" column was undertaken to uncover potential outliers using the *.unique()* method. It was noted that an outlier distance of 103 km corresponded to the postal code "0". Subsequently, a comprehensive investigation into all recorded daily travel distances associated with postal code "0" was conducted, necessitating the calculation of the count of unique values for this specific postal code using the *.count()* method. As 0 was defined in the survey as "outside" Dublin, but not exactly how far away, external data was used to validate the data point. However, no external data can be used to explain the validity of this outlier, rendering it unsubstantiated, and as 103 km is definitely outside Dublin, it seems to be plausible. Given the apparent fabrication of values within the entire "daily travel" column, the outlier was retained rather than removed, as eliminating implausible values would substantially diminish the dataset's observational pool.

   Visualising the Relationship between Daily Travel and Postcode

```
plt.scatter(demographics_no_na["Old Dublin postcode (0 if outside Dublin)"], demographics_no_na["Daily travel to DCU (in
plt.title("Relation between Postcode and Daily Travel")
plt.xlabel("Postcode")
plt.ylabel("Daily Travel")
plt.show()
```



To address the missing value in the "daily travel" column, mean substitution was employed as an imputation technique, leveraging the average travel distance to DCU per postcode. This involved calculating the mean as well as the maximum values and standard deviation of "Daily travel to DCU" distance for each Dublin postcode using a *.groupby()* statement in Python. Notably, the missing value in the "daily travel" column was associated with the postcode "0", which exhibited an average travel distance of 11.6 km. Despite the acknowledgment that this postcode is fictitious, and many recorded values are implausible, mean substitution was deemed appropriate, as discarding all implausible values would result in the exclusion of a substantial portion of the dataset's observations.

Understanding the Relationship between Daily Travel and Postcode

```
demographics_no_na_post_dist = demographics_no_na[["Daily travel to DCU (in km, 0 if on-campus)", "Old Dublin postcode (
demographics_no_na_post_dist.groupby("Old Dublin postcode (0 if outside Dublin)").mean()
```

Old Dublin postcode (0 if outside Dublin)	Daily travel to DCU (in km, 0 if on-campus)
0	11.603261
1	4.910000
2	1.000000
3	5.800000
4	59.000000
5	22.333333
7	8.027273
9	2.727273
11	2.744444
12	15.925000
13	10.666667
14	7.000000
15	15.666667
18	20.000000
19	14.000000
22	20.000000
24	20.000000

Imputing the Missing Daily Travel Value

```
demographics_is_na.iloc[1,2] = 11.60
demographics_is_na
```

	Age (in years)	CAO Points (100 to 600)	Daily travel to DCU (in km, 0 if on-campus)	Average year 1 exam result (as %)	Seat row in class	Gender	Number of older siblings	Number of younger siblings	Old Dublin postcode (0 if outside Dublin)	Height (in cm)	Weight (in kg)	Eye colour	Hair colour	Last 4 digits of your mobile (0000 to 9999)	Star sign	Shoe size
0	21.0	NaN	10.0	66.0	4.0	Male	0	2	0	178.0	92.0	Green	Brown	5262	Capricorn	9.0
1	22.0	505.0	11.6	68.0	7.0	Female	0	1	0	155.0	55.0	Blue	Brown	7181	Leo	4.0
2	20.0	600.0	30.0	NaN	6.0	Female	1	2	0	180.0	55.0	Blue	Brown	7677	Leo	5.0
3	19.0	543.0	10.0	71.0	NaN	Male	0	1	13	187.0	76.0	Blue	Red	838	Taurus	9.0
4	NaN	407.0	8.0	73.0	1.0	Male	2	0	15	181.0	87.0	Blue	Brown	6290	Cancer	9.5

2. CAO Points (100 to 600)

In the endeavour to estimate the missing CAO Points value, a comprehensive examination of three logical connections was undertaken: Age, Gender, and Average Year 1 Exam Result (as %). Given the categorical nature of Gender data, it was categorised as unsuitable to include these datapoints in a linear regression analysis and thus was excluded from further consideration. Subsequently, employing Age and Average Year 1 Exam Result (as %) as independent variables and CAO Points (ranging from 100 to 600) as the dependent variable, a linear regression model was constructed using the Sklearn library in Python. This analysis was conducted on the dataset having no missing values. Due to the partially made-up nature of the dataset, the attained accuracy score (R^2) of 0.07 suggested a limited explanatory power of the model. Nonetheless, under the assumption that a dataset comprising authentic observations would yield a substantially higher accuracy score, the *.predict()* method was employed to extrapolate the

missing value. Utilizing the available data from the observation, this method returned a predicted CAO Points value of 491, which was then inserted into the dataset.

Linear Regression to Generate Missing CAO Points Value

```
# import libraries
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import accuracy_score

# Split the data columnwise into the independt variables (x) and the dependent variable (y)
xDf = demographics_no_na[['Age (in years)','Average year 1 exam result (as %)']]
yDf = demographics_no_na['CAO Points (100 to 600)']

# Split between train and test set
X_train, X_test, y_train, y_test = train_test_split(xDf, yDf, test_size=0.3, random_state=0)

#fitting the model
reg_lin = LinearRegression().fit(X_train, y_train)

#getting the accuracy
print(reg_lin.score(X_test, y_test))
print(reg_lin.coef_)
print(reg_lin.predict([[21,66]]))
```

```
0.07764305496508461
[-1.15  1.16]
[491.16]
```

Imputing the Value

```
demographics_is_na.iloc[0,1] = 491
demographics_is_na
```

	Age (in years)	CAO Points (100 to 600)	Daily travel to DCU (in km, 0 if on-campus)	Average year 1 exam result (as %)	Seat row in class	Gender	Number of older siblings	Number of younger siblings	Old Dublin postcode (0 if outside Dublin)	Height (in cm)	Weight (in kg)	Eye colour	Hair colour	Last 4 digits of your mobile (0000 to 9999)	Star sign	Shoe size
0	21.0	491.0	10.0	66.0	4.0	Male	0	2	0	178.0	92.0	Green	Brown	5262	Capricorn	9.0
1	22.0	505.0	11.6	68.0	7.0	Female	0	1	0	155.0	55.0	Blue	Brown	7181	Leo	4.0
2	20.0	600.0	30.0	NaN	6.0	Female	1	2	0	180.0	55.0	Blue	Brown	7677	Leo	5.0
3	19.0	543.0	10.0	71.0	NaN	Male	0	1	13	187.0	76.0	Blue	Red	838	Taurus	9.0
4	NaN	407.0	8.0	73.0	1.0	Male	2	0	15	181.0	87.0	Blue	Brown	6290	Cancer	9.5

3. Average year 1 exam result (as %)

To be able to impute the missing the Average Year 1 Exam Result (as %) value, an examination of three logical connections was undertaken: Age, Gender, and CAO Points (ranging from 100 to 600). Given the categorical nature of Gender data, it was deemed unsuitable for inclusion in a linear regression analysis and consequently omitted from further consideration. Subsequently, employing Age and CAO Points (ranging from 100 to 600) as independent variables and Average Year 1 Exam Result (expressed as a percentage) as the dependent variable, a linear regression model was constructed in Python using the Sklearn library. This analysis used the same code and approach as 2. utilising the dataset with no missing values. The attained accuracy score (R^2) of 0.10 suggested a limited explanatory power of the model. The *.predict()* method was then also employed to generate the missing value. Utilizing the available data from the observation, this method returned a predicted Average Year 1 Exam Result of 70.5%, thereby allowing for the completion of the dataset.

Linear Regression to Generate Missing Year 1 Exam Value

```
# Split the data columnwise into the independt variables (x) and the dependent variable (y)
xDf = demographics_no_na[['Age (in years)', 'CAO Points (100 to 600)']]
yDf = demographics_no_na['Average year 1 exam result (as %)']

# Split between train and test set
X_train, X_test, y_train, y_test = train_test_split(xDf, yDf, test_size=0.3, random_state=0)

#fitting the model
reg_lin = LinearRegression().fit(X_train, y_train)

#getting the accuracy
print(reg_lin.score(X_test, y_test))
print(reg_lin.coef_)
print(reg_lin.predict([[20,600]]))
```

```
0.10728121697137338
[0.12 0.02]
[70.54]
```

Imputing the Value

```
demographics_is_na.iloc[2,3] = 70.5
demographics_is_na
```

	Age (in years)	CAO Points (100 to 600)	Daily travel to DCU (in km, 0 if on-campus)	Average year 1 exam result (as %)	Seat row in class	Gender	Number of older siblings	Number of younger siblings	Old Dublin postcode (0 if outside Dublin)	Height (in cm)	Weight (in kg)	Eye colour	Hair colour	Last 4 digits of your mobile (0000 to 9999)	Star sign	Shoe size
0	21.0	491.0	10.0	66.0	4.0	Male	0	2	0	178.0	92.0	Green	Brown	5262	Capricorn	9.0
1	22.0	505.0	11.6	68.0	7.0	Female	0	1	0	155.0	55.0	Blue	Brown	7181	Leo	4.0
2	20.0	600.0	30.0	70.5	6.0	Female	1	2	0	180.0	55.0	Blue	Brown	7677	Leo	5.0
3	19.0	543.0	10.0	71.0	NaN	Male	0	1	13	187.0	76.0	Blue	Red	838	Taurus	9.0
4	NaN	407.0	8.0	73.0	1.0	Male	2	0	15	181.0	87.0	Blue	Brown	6290	Cancer	9.5

4. Seat Row

To infer the missing Seat Row value, four logical connections were examined: Age, Gender, CAO Points (100 to 600), Average year 1 exam result (as %). Due to its categorical nature, Gender was consistently excluded from subsequent linear regression analyses. Employing Age, Average Year 1 Exam Result (expressed as a percentage), and CAO Points (ranging from 100 to 600) as independent variables, and Seat Row as the dependent variable, a linear regression model was implemented in Python utilizing the Sklearn library. This analysis was also performed on the dataset having no missing values. However, the resulting negative accuracy score of -0.07 indicated that a linear regression model was not a suitable fit for predicting seat rows. Given the apparent randomness of Seat Row assignments, a pragmatic approach was adopted. A random number was generated within the range of the variable, specifically between the minimum (1) and maximum (12) values, using the *random.randint()* function of the NumPy library. The number was added to the dataset to address the missing Seat Row value.

Negative Accuracy Score of Linear Regression

```
# Split the data columnwise into the independent variables (x) and the dependent variable (y)
xDf = demographics_no_na[['Age (in years)', 'CAO Points (100 to 600)', "Average year 1 exam result (as %)"]]
yDf = demographics_no_na['Seat row in class']

# Split between train and test set
X_train, X_test, y_train, y_test = train_test_split(xDf, yDf, test_size=0.3, random_state=0)

#fitting the model
reg_lin = LinearRegression().fit(X_train, y_train)

#getting the accuracy
print(reg_lin.score(X_test, y_test))
print(reg_lin.coef_)
print(reg_lin.predict([[19,543,71]]))
```

```
-0.07013573261755024
[ 0.19  0.   -0.04]
[5.2]
```

Random Value to Fill the Missing Seat Row Value

```
#no meaningful correlation which was not expected in any other way
np.random.randint(1,12)
```

```
7
```

```
demographics_is_na.iloc[3,4] = 7
demographics_is_na
```

	Age (in years)	CAO Points (100 to 600)	Daily travel to DCU (in km, 0 if on-campus)	Average year 1 exam result (as %)	Seat row in class	Gender	Number of older siblings	Number of younger siblings	Old Dublin postcode (0 if outside Dublin)	Height (in cm)	Weight (in kg)	Eye colour	Hair colour	Last 4 digits of your mobile (0000 to 9999)	Star sign	Shoe size
0	21.0	491.0	10.0	66.0	4.0	Male	0	2	0	178.0	92.0	Green	Brown	5262	Capricorn	9.0
1	22.0	505.0	11.6	68.0	7.0	Female	0	1	0	155.0	55.0	Blue	Brown	7181	Leo	4.0
2	20.0	600.0	30.0	70.5	6.0	Female	1	2	0	180.0	55.0	Blue	Brown	7677	Leo	5.0
3	19.0	543.0	10.0	71.0	7.0	Male	0	1	13	187.0	76.0	Blue	Red	838	Taurus	9.0
4	21.0	407.0	8.0	73.0	1.0	Male	2	0	15	181.0	87.0	Blue	Brown	6290	Cancer	9.5

5. Age

To address the absence of the one age value within the dataset, a comprehensive review of the dataset's initial description was conducted to enhance comprehension of the variable. Upon careful consideration, mean and median substitution emerged as viable imputation techniques. Considering all given variables, there seems to be no logical causal relationship between age and any of the other variables. For example, neither the number of siblings, gender, nor personal appearance variables have a logical relation to the age of a person. Furthermore, given the predominantly narrow range of age values (with 75% falling between 18 and 22 years), linear regression was deemed an unsuitable approach due to the observed weak correlations within the dataset pertaining to age. Subsequently, median substitution was selected as the preferred imputation technique. Notably, the mean age of 22.2 exceeded the third quartile value of 22 years, primarily due to the presence of a few outliers. Consequently, to better reflect the sample distribution, a median age of 21 years was chosen for substitution.

Median Imputation of Missing Age Value

```
#because we cannot predict the age here we use the median to fill it in
demographics_is_na.iloc[4,0] = 21
demographics_is_na
```

	Age (in years)	CAO Points (100 to 600)	Daily travel to DCU (in km, 0 if on-campus)	Average year 1 exam result (as %)	Seat row in class	Gender	Number of older siblings	Number of younger siblings	Old Dublin postcode (0 if outside Dublin)	Height (in cm)	Weight (in kg)	Eye colour	Hair colour	Last 4 digits of your mobile (0000 to 9999)	Star sign	Shoe size
0	21.0	491.0	10.0	66.0	4.0	Male	0	2	0	178.0	92.0	Green	Brown	5262	Capricorn	9.0
1	22.0	505.0	11.6	68.0	7.0	Female	0	1	0	155.0	55.0	Blue	Brown	7181	Leo	4.0
2	20.0	600.0	30.0	70.5	6.0	Female	1	2	0	180.0	55.0	Blue	Brown	7677	Leo	5.0
3	19.0	543.0	10.0	71.0	7.0	Male	0	1	13	187.0	76.0	Blue	Red	838	Taurus	9.0
4	21.0	407.0	8.0	73.0	1.0	Male	2	0	15	181.0	87.0	Blue	Brown	6290	Cancer	9.5

After employing a comprehensive array of five distinct imputation techniques, the dataset's missing values were imputed utilizing methods best suited to the data type, scrutinized correlations, and the inherent characteristics of the dataset. With the dataset now normalized, a pivotal step towards merging it with the secondary dataset has been achieved, fostering a more comprehensive and holistic analysis.

2. Dataset Merge

After completing the imputation, the demographics dataset previously containing the missing values and the demographics dataset containing no missing values were readded into one dataset using the *.concat()* function.

Merging the Two Subsets to One Normalised Demographics Dataset

```
normalised_demographics = pd.concat([demographics_is_na, demographics_no_na], axis = 0)
normalised_demographics
```

	Age (in years)	CAO Points (100 to 600)	Daily travel to DCU (in km, 0 if on-campus)	Average year 1 exam result (as %)	Seat row in class	Gender	Number of older siblings	Number of younger siblings	Old Dublin postcode (0 if outside Dublin)	Height (in cm)	Weight (in kg)	Eye colour	Hair colour	Last 4 digits of your mobile (0000 to 9999)	Star sign	Shoe size
0	21.0	491.0	10.0	66.0	4.0	Male	0	2	0	178.0	92.0	Green	Brown	5262	Capricorn	9.0
1	22.0	505.0	11.6	68.0	7.0	Female	0	1	0	155.0	55.0	Blue	Brown	7181	Leo	4.0
2	20.0	600.0	30.0	70.5	6.0	Female	1	2	0	180.0	55.0	Blue	Brown	7677	Leo	5.0
3	19.0	543.0	10.0	71.0	7.0	Male	0	1	13	187.0	76.0	Blue	Red	838	Taurus	9.0
4	21.0	407.0	8.0	73.0	1.0	Male	2	0	15	181.0	87.0	Blue	Brown	6290	Cancer	9.5
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
117	21.0	565.0	7.0	71.0	3.0	Female	2	1	14	170.0	59.0	Blue	Black	1699	Libra	6.0
118	20.0	520.0	7.0	62.0	8.0	Female	0	2	7	165.0	70.0	Blue	Black	5289	Taurus	5.0
119	20.0	510.0	60.0	65.0	5.0	Male	1	1	0	184.0	85.0	Blue	Brown	5317	Libra	10.0
120	23.0	490.0	7.0	70.0	4.0	Female	0	3	7	173.0	73.0	Green	Blonde	4602	Aquarius	7.0
121	22.0	500.0	4.0	65.0	4.0	Female	0	3	0	177.0	73.0	Blue	Brown	6402	Leo	7.0

To merge the demographics and the personality dataset, the latter one was loaded into a Pandas Data Frame in Python. As Primary Key of the personality dataset, the last four-digit phone numbers were identified being also the Primary Key in the imputed demographics dataset. Firstly, three four-digit phone numbers were identified as duplicates being not unique in the dataset (4397, 1699, 1462). By comparing the duplicates to clue was provided on whether there was a relation or false entry. Subsequently, only the first instance of the personality data was kept, assuming that these were the original entries.

Deleting Duplicate Rows in Personalities Dataset

```
personalities[personalities["Last 4 digits of your mobile (0000 to 9999)"] == 4397]
```

	Last 4 digits of your mobile (0000 to 9999)	Extraversion	Intuition	Thinking	Judging	Assertive
22	4397	74	48	43	29	63
53	4397	65	70	70	65	60

```
personalities.drop(53, axis =0, inplace = True)
personalities
```

	Last 4 digits of your mobile (0000 to 9999)	Extraversion	Intuition	Thinking	Judging	Assertive
0	7703	71	66	65	31	81
1	5406	60	68	30	35	33
2	7181	32	62	23	35	31
3	971	46	48	34	64	42
4	7263	29	79	23	83	39
...	...	...	...	...	...	...
98	4602	29	23	32	60	32
99	3839	70	50	25	24	46
100	5085	22	93	44	83	8
101	6402	78	68	53	71	63
102	1573	81	49	55	61	64

102 rows × 6 columns

```
personalities[personalities["Last 4 digits of your mobile (0000 to 9999)"] == 1699]
```

	Last 4 digits of your mobile (0000 to 9999)	Extraversion	Intuition	Thinking	Judging	Assertive
14	1699	64	69	63	58	60
89	1699	64	69	63	58	60

```
personalities[personalities["Last 4 digits of your mobile (0000 to 9999)"] == 1462]
```

	Last 4 digits of your mobile (0000 to 9999)	Extraversion	Intuition	Thinking	Judging	Assertive
49	1462	68	68	46	71	49
97	1462	78	72	46	60	37

```
personalities.drop(89, axis =0, inplace = True)
personalities
```

	Last 4 digits of your mobile (0000 to 9999)	Extraversion	Intuition	Thinking	Judging	Assertive
0	7703	71	66	65	31	81
1	5406	60	68	30	35	33
2	7181	32	62	23	35	31
3	971	46	48	34	64	42
4	7263	29	79	23	83	39
...	...	...	...	...	...	...
98	4602	29	23	32	60	32
99	3839	70	50	25	24	46
100	5085	22	93	44	83	8
101	6402	78	68	53	71	63
102	1573	81	49	55	61	64

101 rows × 6 columns

```
personalities.drop(97, axis =0, inplace = True)
personalities
```

	Last 4 digits of your mobile (0000 to 9999)	Extraversion	Intuition	Thinking	Judging	Assertive
0	7703	71	66	65	31	81
1	5406	60	68	30	35	33
2	7181	32	62	23	35	31
3	971	46	48	34	64	42
4	7263	29	79	23	83	39

Then an inner join was performed in Python with the *.merge()* method specified with *how="inner"* on the column for the four-digit phone number and saved to a new data frame ("demo_perso"). This yielded only 65 rows that were in the demographics data set and in the personality data set, which were saved in a new data frame. As this seemed suspiciously low, this was additionally confirmed by counting the number of unique values in both datasets with a for loop, which yielded the same result. The remaining rows from both datasets (demographics and personalities) where then merged through an outer join through the *.merge()* method specified with *how="outer"*, having all these entries in a separate data frame ("demo_perso_outer"). Overall, the outer join dataset counts 157 rows being significantly larger than the one with no missing values indicating that the surveys conducted have lost participants and that many participants could potentially not remember the 4 digits from the first survey. This dataset, however, is far from normalised, but can be seen as a save copy for all entries.

Inner Join of the Two Datasets

```
)_perso = normalised_demographics.merge(personalities, on = "Last 4 digits of your mobile (0000 to 9999)", how = "inner")
)_perso
```

	Age (in years)	CAO Points (100 to 600)	Daily travel to DCU (in km, 0 if on-campus)	Average year 1 exam result (as %)	Seat row in class	Gender	Number of older siblings	Number of younger siblings	Old Dublin postcode (0 if outside Dublin)	Height (in cm)	...	Eye colour	Hair colour	Last 4 digits of your mobile (0000 to 9999)	Star sign	Shoe size	Extraversion	Intu
0	22.0	505.0	11.6	68.0	7.0	Female	0	1	0	155.0	...	Blue	Brown	7181	Leo	4.0	32	
1	19.0	543.0	10.0	71.0	2.0	Male	0	1	13	187.0	...	Blue	Red	838	Taurus	9.0	48	
2	21.0	557.0	15.0	50.5	10.0	Male	2	0	15	178.0	...	Green	Brown	4397	Aries	7.0	74	
3	22.0	397.0	5.0	65.0	5.0	Female	2	0	0	153.0	...	Brown	Brown	8626	Sagittarius	5.0	25	
4	25.0	532.0	20.0	72.0	8.0	Male	1	1	22	192.0	...	Green	Brown	4768	Aquarius	12.0	56	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
60	21.0	565.0	7.0	71.0	3.0	Female	2	1	14	170.0	...	Blue	Black	1699	Libra	6.0	64	
61	20.0	520.0	7.0	62.0	8.0	Female	0	2	7	165.0	...	Blue	Black	5289	Taurus	5.0	61	
62	20.0	510.0	60.0	65.0	5.0	Male	1	1	0	184.0	...	Blue	Brown	5317	Libra	10.0	25	
63	23.0	490.0	7.0	70.0	4.0	Female	0	3	7	173.0	...	Green	Blonde	4602	Aquarius	7.0	29	
64	22.0	500.0	4.0	65.0	4.0	Female	0	3	0	177.0	...	Blue	Brown	6402	Leo	7.0	78	

Outer Join to Save Incomplete Rows of Both Datasets

```
)_outer = normalised_demographics.merge(personalities, on = "Last 4 digits of your mobile (0000 to 9999)", how = "outer")
)_outer
```

	Age (in years)	CAO Points (100 to 600)	Daily travel to DCU (in km, 0 if on-campus)	Average year 1 exam result (as %)	Seat row in class	Gender	Number of older siblings	Number of younger siblings	Old Dublin postcode (0 if outside Dublin)	Height (in cm)	...	Eye colour	Hair colour	Last 4 digits of your mobile (0000 to 9999)	Star sign	Shoe size	Extraversion	Int
0	21.0	491.0	10.0	66.0	4.0	Male	0.0	2.0	0.0	178.0	...	Green	Brown	5262	Capricorn	9.0	NaN	
1	22.0	505.0	11.6	68.0	7.0	Female	0.0	1.0	0.0	155.0	...	Blue	Brown	7181	Leo	4.0	32.0	
2	20.0	600.0	30.0	70.5	6.0	Female	1.0	2.0	0.0	180.0	...	Blue	Brown	7677	Leo	5.0	NaN	
3	19.0	543.0	10.0	71.0	2.0	Male	0.0	1.0	13.0	187.0	...	Blue	Red	838	Taurus	9.0	48.0	
4	21.0	407.0	8.0	73.0	1.0	Male	2.0	0.0	15.0	181.0	...	Blue	Brown	6290	Cancer	9.5	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
152	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	9893	NaN	NaN	71.0	

3. Unified Dataset Analysis

Following the merger of the two datasets, an exhaustive examination of the unified dataset was conducted, encompassing an assessment of correlations, linear regression, and hypothesis testing.

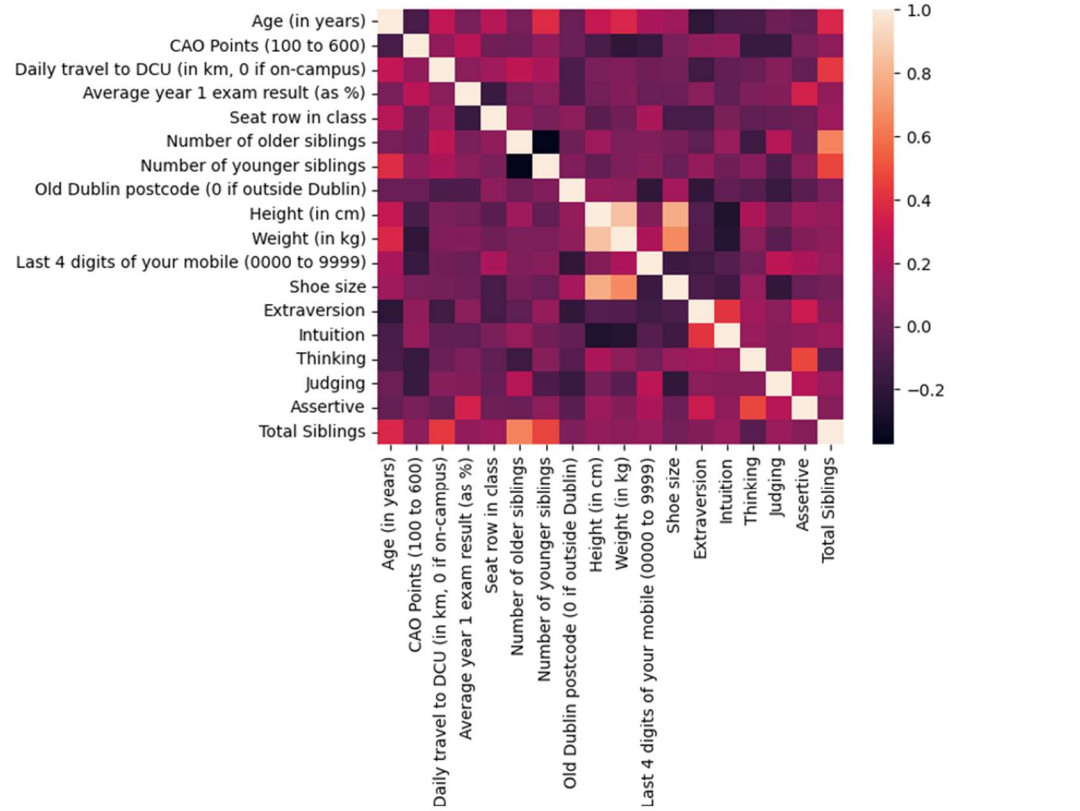
3.1 Correlations for Numerical Columns

Numeric columns were plotted against each other and correlation coefficients were computed using the *.corr()* method to enhance understanding of variable relationships. Given the substantial number of potential pairs, a filtering script was implemented to discern correlations falling within the range of 0.6 to 1, thereby excluding low correlations and the trivial perfect correlation between mirrored columns. This analysis revealed significant correlations primarily concentrated among the height, weight, and shoe size variables. However, most correlations exhibit relatively low magnitudes.

Correlation Heatmap

```
correlations = demographics_personalities.drop(columns = ['Gender', 'Hair colour', 'Star sign', 'Eye colour']).corr()
correlations
```

```
sns.heatmap(correlations)
```




```
tpllst = []
for i in range(0,18):
    for j in range(0,18):
        if correlations.iloc[i,j] > 0.6 and correlations.iloc[i,j] < 1.0:
          tpllst.append((correlations.index[i], correlations.columns[j], correlations.iloc[i, j]))
tpllst
```

```
[('Number of older siblings', 'total_siblings', 0.647470523325822),
 ('Height (in cm)', 'Weight (in kg)', 0.8542174262263178),
 ('Height (in cm)', 'Shoe size', 0.776188771867279),
 ('Weight (in kg)', 'Height (in cm)', 0.8542174262263178),
 ('Weight (in kg)', 'Shoe size', 0.6657007827143665),
 ('Shoe size', 'Height (in cm)', 0.776188771867279),
 ('Shoe size', 'Weight (in kg)', 0.6657007827143665),
 ('total_siblings', 'Number of older siblings', 0.647470523325822)]
```

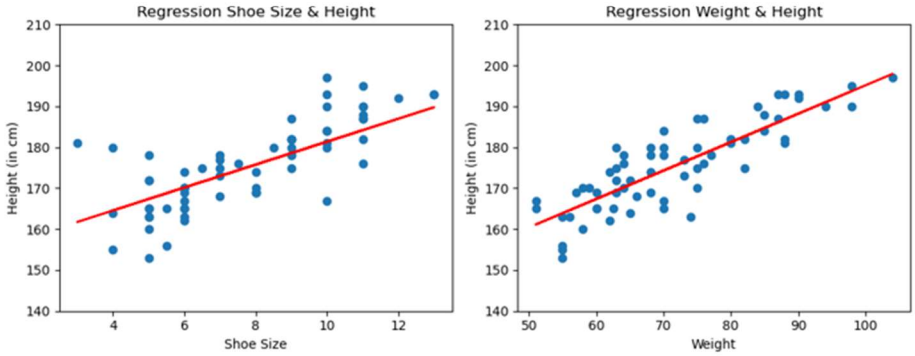
3.2 Height as a Function of Weight and Shoe Size

Building upon our observations concerning the correlations within the dataset, a deeper examination was conducted focusing on the variables of height, weight, and shoe size. Utilising a linear regression model sourced from the SKlearn library, weight and shoe size were designated as independent variables, while height served as the dependent variable within the linear model. The analysis revealed a robust association, as indicated by a coefficient of determination (R^2) of 0.88, signifying a high degree of predictability in height based on shoe size and weight. Furthermore, two scatterplots – including one regression each – with height as the dependent variable and either shoe size or weight as the only independent variable were created to better visualise the findings.

Independent Regression and Plotting

```
# Split the data columnwise into the independt variables (x) and the dependent variable (y)
xDf = demo_perso[['Weight (in kg)']]
yDf = demo_perso['Height (in cm)']

# Split between train and test set
X_train, X_test, y_train, y_test = train_test_split(xDf, yDf, test_size=0.3, random_state=0)

#fitting the model
reg_lin_weight = LinearRegression().fit(X_train, y_train)
intercept_weight = reg_lin_weight.intercept_

#getting the accuracy
print(reg_lin_weight.score(X_test, y_test))
print(reg_lin_weight.coef_)
print(intercept_weight)

# Split the data columnwise into the independt variables (x) and the dependent variable (y)
xDf = demo_perso[['Shoe size']]
yDf = demo_perso['Height (in cm)']

# Split between train and test set
X_train, X_test, y_train, y_test = train_test_split(xDf, yDf, test_size=0.3, random_state=0)

#fitting the model
reg_lin_shoe = LinearRegression().fit(X_train, y_train)
intercept_shoe = reg_lin_shoe.intercept_

#getting the accuracy
print(reg_lin_shoe.score(X_test, y_test))
print(reg_lin_shoe.coef_)
print(intercept_shoe)


fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(10, 4))

# Plot the first line graph on the first subplot
ax1.scatter(demo_perso["Shoe size"], demo_perso["Height (in cm)"])
ax1.plot(demo_perso["Shoe size"], intercept_shoe + reg_lin_shoe.coef_*demo_perso["Shoe size"], color = "red")
ax1.set_xlabel('Shoe Size')
ax1.set_ylabel('Height (in cm)')
ax1.set_title('Regression Shoe Size & Height')
ax1.set_ylim(140,210)


# Plot the second line graph on the second subplot
ax2.scatter(demo_perso["Weight (in kg)"], demo_perso["Height (in cm)"])
ax2.plot(demo_perso["Weight (in kg)"], intercept_weight + reg_lin_weight.coef_*demo_perso["Weight (in kg)"], color = "re
ax2.set_xlabel('Weight')
ax2.set_ylabel('Height (in cm)')
ax2.set_title('Regression Weight & Height')
ax2.set_ylim(140,210)


plt.tight_layout()

plt.show()
```

```
0.7898408272863814
[0.69]
125.7143292319387
0.7640584023409857
[2.8]
153.34343434343435
```



3.3 Personality Differences Between the Genders

Analysing the mean values of each personality trait categorized by gender unveiled apparent disparities in personality profiles between genders. To delve deeper into this phenomenon, two distinct data frames were crafted: one comprising solely the personality observations of male participants and the other containing personality traits for female participants. Subsequently, independent two-sample t-tests were conducted for each personality trait to ascertain whether the observed means for men and women exhibited genuine differences in the broader population or were merely attributable to sampling variability. The SciPy library facilitated the execution of these statistical tests (the same code was used as in 3.4). The null hypothesis for each of the five tests conducted consistently posited the following assertion.

H_0: There is no Difference Between Men and Women in the Examined Personality Trait

Choosing a significance level of 95% a p-value of lower than 0.05 was required to reject the null hypothesis. For none of the five tests this was the case, the p-values were as followed:

Personality Trait	P-Value
EXTRAVERSION	0.74
INTUITION	0.12
THINKING	0.22
JUDGING	0.40
ASSERTIVE	0.12

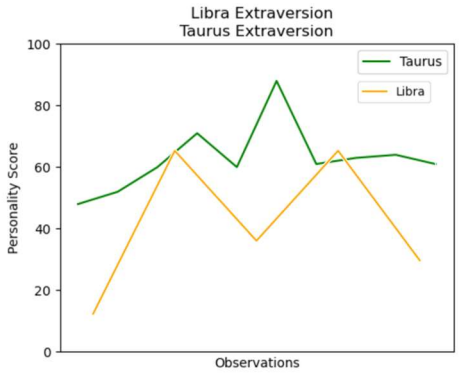
3.4 Personality Differences Between the Star Signs

A similar analysis was conducted regarding the variable star signs. Firstly, the mean for all-star signs for each personality trait was calculated using the *.groupby()* and *.mean()* method. Then the star sign with the highest and lowest mean were compared with an independent sample t test using the SciPy library for scientific python calculations. Here the Cancer star sign was excluded as there was only one observation for this star sign.

Personality by Star Sign

```
demographics_personalities[['Star sign','EXTRAVERSION','INTUITION','THINKING','JUDGING','ASSERTIVE']]
.groupby('Star sign').mean()
```

Star sign	EXTRAVERSION	INTUITION	THINKING	JUDGING	ASSERTIVE
Aquarius	53.285714	55.285714	47.714286	48.857143	38.571429
Aries	56.666667	46.000000	46.000000	57.333333	45.666667
Cancer	64.000000	49.000000	36.000000	39.000000	51.000000
Capricorn	54.750000	52.750000	43.000000	69.750000	47.000000
Gemini	58.428571	50.571429	54.142857	55.571429	50.142857
Leo	52.888889	60.666667	50.666667	52.666667	50.000000
Libra	38.200000	40.800000	40.600000	44.200000	27.600000
Pices	56.000000	68.750000	47.000000	52.750000	44.500000
Sagittarius	50.666667	57.333333	39.833333	58.500000	55.000000
Scorpio	48.400000	54.800000	46.000000	66.000000	49.000000
Taurus	62.800000	53.400000	47.800000	53.300000	51.300000
Virgo	55.750000	52.250000	51.750000	62.250000	33.250000




```
taur_extra = demographics_personalities[demographics_personalities['Star sign'] == 'Taurus']['EXTRAVERSION']
libra_extra = demographics_personalities[demographics_personalities['Star sign'] == 'Libra']['EXTRAVERSION']

t_statistic, p_value = stats.ttest_ind(taur_extra, libra_extra)

print("t-statistic:", t_statistic)
print("p-value:", p_value)
```

```
t-statistic: 2.6831956256764093
p-value: 0.018786154657515874
```

The pairs with their means, Null hypotheses and the corresponding p-values are listed in the table below.

Personality Trait	Star sign 1	Mean	Star sign 2	Mean	H0	P-Value	Result
EXTRAVERSION	Taurus	62.8	Libra	38.2	Taurus are not more extravert on avarege then Libra	0.019	H0 rejected; Taurus are on average more extravert then Libra
INTUITION	Pices	68.8	Libra	40.8	Pices are not more intuitive than Libra on average	0.074	H0 not rejected
THINKING	Gemini	54.1	Sagittarius	39.8	Gemini are not more thinking than Sagittarius on average	0.022	H0 rejected; Gemini are on average more thinking than Sagittarius
JUDGING	Capricorn	69.8	Libra	44.2	Capricorn are not on average more judging then Libra	0.058	H0 not rejected
ASSERTIVE	Sagittarius	55.0	Virgo	33.3	Sagittarius are not more assertive on average than Virgos	0.117	H0 rejected; Sagittarius are more assertive on average than Virgos

Keeping in mind that the number of observations used is quite limited, two conclusions emerge:

Gemini are on average more thinking than Sagittarius.

Sagittarius are more assertive on average than Virgos.

3.5 The Darkness of Eye Colour Depending on the Darkness of Hair Colour

Given the genetic nature of both hair and eye colours, an intrinsic link between these variables appeared plausible. However, owing to their categorical nature, a preliminary transformation of the data was imperative. Hair and eye colour variables underwent conversion into numerical values via label encoding. While typically reserved for ordinal data, this methodology was deemed appropriate by interpreting hair and eye colours as varying degrees of darkness (e.g., blond denoting light shades, black representing darker hues), thereby rendering the data amenable to an ordinal scale. To effect this transformation, a new column was appended to the dataset, and an encoding dictionary was applied to map the categorical values using the *.map()* method.

Label Encoding

```
encode_hair_dict = {'Blonde' : 0,
                    'Red' : 1,
                    'Brown' : 2,
                    'Black' : 3}
demographics_personalities['Hair colour encoded'] = demographics_personalities['Hair colour'].map(encode_hair_dict)
demographics_personalities
encode_eye_dict = {'Blue' : 0,
                    'Green' : 1,
                    'Brown' : 2}
demographics_personalities['Eye colour encoded'] = demographics_personalities['Eye colour'].map(encode_eye_dict)
demographics_personalities
```

	Number of older siblings	Number of younger siblings	Old Dublin postcode (0 if outside Dublin)	Height (in cm)	...	Star sign	Shoe size	EXTRAVERSION	INTUITION	THINKING	JUDGING	ASSERTIVE	total_siblings	Hair colour encoded	Eye colour encoded
e	0	1	0	155.0	...	Leo	4.0	32	62	23	35	31	1	2	0
e	0	1	13	187.0	...	Taurus	9.0	48	41	57	49	36	1	1	0
e	2	0	15	178.0	...	Aries	7.0	74	48	43	29	63	2	2	1
e	2	0	0	153.0	...	Sagittarius	5.0	25	55	43	61	44	2	2	2
e	1	1	22	192.0	...	Aquarius	12.0	56	68	27	47	49	2	2	1
..	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
e	2	1	14	170.0	...	Libra	6.0	64	69	63	58	60	3	3	0
e	0	2	7	165.0	...	Taurus	5.0	61	55	66	49	67	2	3	0
e	1	1	0	184.0	...	Libra	10.0	25	24	49	44	10	2	2	0
e	0	3	7	173.0	...	Aquarius	7.0	29	23	32	60	32	3	0	1
e	0	3	0	177.0	...	Leo	7.0	78	68	53	71	63	3	2	0

In the next step a linear regression was employed with hair darkness as the independent variable and eye darkness as dependent variable. However, the returned score of only 0.014 suggest only a limited relationship.

3.6 Further Examinations and Failed Linear Regressions

Additional analyses with linear regression showed inconclusive results (i.e., negative R^2). For instance, using total siblings as independent and CAO score as dependent variables. Separate examinations of older and younger siblings on test scores yielded no conclusive findings. Finally, linear regression with personality traits as independent variables and Year 1 Exam score as dependent resulted in a low score of 0.1.

4. Conclusion

Having analysed the data, it becomes clear that telling a story about the dataset is quite difficult due to the made-up nature of the data entries and the resulting data distribution. This can also be best seen considering the correlation heatmap of all numeric variables underlining low correlations among the great majority of variable pairs. The only exception seems to be the correlations between shoe size, height and weight assuming that the survey participants inserted for these variables their actual personal data. Moreover, two hypotheses regarding personality differences and star signs were not rejected allowing for the conclusion that Gemini are on average more thinking than Sagittarius, while Sagittarius are more assertive on average than Virgos.