Word Count: 586

PERRY, LUKE (Student)

Data Cleaning and Analysis Report

# Problem 1

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Issue | Cleaning Actions | Justifications |
| model | NaN values | Drop records | Need to uniquely identify records |
| cpu\_speed | Overwhelming majority of NaN values | Drop column | Ambiguous cpu naming and lack of data makes column useless [1] |
| color | Irrelevant data | Drop column | Doesn’t affect laptop quality |
| all | Leading/trailing whitespace and inconsistent capitalisation | Remove whitespace and decapitalise strings | Standardising string format will enable accurate comparison |
| harddisk | Inconsistent units | Convert all values to gb | Using consistent scale enables accurate comparison + removal of unit |
| screen\_size, harddisk, ram, price | Unnecessary string inclusions | Remove all strings and convert to float | Removing strings allows values to be treated as numbers and used for computation/ comparison |
| screen\_size | NaN values | Impute values with Decision Tree Regressor trained with price. Rounded to 1dp | Decision tree regression evaluation revealed model was an accurate predictor [2] |
| ram, harddisk | NaN values | Fill with mean rounded to nearest power of 2 (capped at 32, 2048) | Fill with mean to not (dis)credit the record unnecessarily. Storage units increase in powers of 2, and implausible values useless |
| model, brand | Unnecessary split | Merge columns | Model implies brand - redundant |
| cpu,os, special\_features | Unstandardised values | Standardise with custom matching algorithm | Without detailed information, grouping is more useful. Also eliminates mildly erroneous values. Features value converted to array. |
| graphics,graphics\_coprocessor | Graphics\_coprocessor implies graphics | Merge columns, where you prioritise coprocessor, assume integrated | Reduces unnecessary data |
| rating | NaN values and low rated records | Fill NaN with mean of values, remove records with rating less than average | Not (dis)crediting records unnecessarily + narrowing recommendation pool to only the best |
| price | NaN values + values > 1500 | Impute values with Decision Tree Regressor | Decision tree regression evaluation revealed model was an accurate predictor [3] + laptops with price > 1500 are outside of scope |
| cpu | NaN values | Remove NaN records | Too important to be omitted and cannot be imputed accurately from data |

A black background with white text

Description automatically generatedA screenshot of a cell phone

Description automatically generated

Figure 3

Figure 2

Figure 1

# Problem 2

My general process for discovering appropriate laptops is to calculate an overall score for the laptop by analysing the performance values in each feature, and scaling these scores by a series of weights which can be modified to better represent the needs of a given use-case of customer. The combined score from each feature generates an *overall score.* Then by comparing the records by their overall scores, whilst taking into account the OS and/or any special features that may qualitatively improve the “score”, so I can make more accurate recommendations.

More specifically, this process firstly involves converting all columns into their equivalent z-scores; this will intrinsically roughly normalise the data into a gaussian distribution where a score of 0 indicates an average score and positive values indicate good performance, whilst negative values indicate poor performance. This can then be converted into a percentile score between 0-1 by reference with a z-table – making all overall scores fall between 0-(Σ wi) where wi denotes the ith weight. This both allows different features to be compared by their relative performance, and reduces the affect of an outlier on the calculation for the laptops overall score.

To calculate the performance values of both the CPU and GPU, I used benchmark performance scores from the websites: <https://cpu.userbenchmark.com/>, <https://versus.com/en>, <https://www.cpu-monkey.com/en/>, and <https://www.notebookcheck.net/>. Where I primarily used <https://cpu.userbenchmark.com/> for benchmark scores, and used the other websites to fill in processors not part of the Userbenchmark database by comparing a relative score of a processor provided by the additional website database with a score already present in the Userbenchmark database, rescaling the value appropriately. I entered this data into a dictionary and mapped the categorial values in the pandas DataFrame with the equivalent z score from the benchmark values.

## Influencer