Benjamin Cole s3412349

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Criteria** | **Description** | | | | | | | | | | | | **Points 19/20** |
| **Source and Description** | *Excellent* | | | *Good* | | | *Average* | | | *Poor* | | | 3/3 |
| Clear data source reference and description. | | |  | | |  | | |  | | |
| **Goals and Objectives** | *Excellent* | | | *Good* | | | *Average* | | | *Poor* | | | 3/3 |
| Clearly stated. | | |  | | |  | | |  | | |
| **Data Pre-processing** | *Outstanding* | *Excellent* | | | *Good* | | *Average* | | *Below Average* | | | *Poor* | 4/5 |
|  | The data wrangling was methodological, but the codes were prone to error. Since you loaded “tidyverse” package, you should have used pipeline operators. Example 1: the following can be written as:  advertsing\_train\_long\_num <- adversting\_train %>%  select(-case\_id, -countryId, -dow)    Tip: you can use summariseColumns from mlr package to compute in-depth summary statistics.  Example 2: you could use “mutate” from tidyverse to convert the variables below to factor types.    Example 3: is.infinite(x) will return Boolean value. So, you should not use “== T” again. In some cases, it might result in errors.    The above are recommendations and so no penalty. Now, here are a few things that cost you some points.  While box-cox transformation was well-implemented, you ought to be more careful in log-transformation (which is a special case of box-cox with lambda = 0). Some numeric variables contain zeroes and resulted in infinite value after log-transformation. So, your data analysis would be incomplete after removing infinite values. To mitigate, you can introduce a small “noise” in the log function. For example, say the lowest value of *y* is 0.01. Then you can take y’ = log(*y*+*c*) where *c (the small noise)* is 0.005 – slighter higher than 0.01. When it comes to interpretation, inverse-transform it by taking account of c. That is y = exp(y’) - c    The warning should not be ignored. I doubt some of the items were replaced properly during the min/max normalisation.    Two solutions:   1. Use normalisation function from mlr package. 2. See the attached R script which uses “base” R functions to min/max normalise each numeric column. | | |  | |  | |  | | |  |
| **Data Analysis** | *Outstanding* | *Excellent* | | | *Good* | | *Average* | | *Below Average* | | | *Poor* | 4/4 |
| The analysis was methodologically sound. |  | | |  | |  | |  | | |  |
| **Overall Quality** | *Outstanding* | | *Excellent* | | | *Good* | | *Average* | | | *Poor* | | 4/4 |
| Well formatted and professionally written. Good job! | |  | | |  | |  | | |  | |