Data Quality Plan

1. Overview

The purpose of this Data Quality Plan is to explore the identified issues in ‘cleaned_data.csv’ (which has been gleaned from ‘covid19-cdc-23222213.csv’), discuss the potential causes of these issues and identify solutions to aforementioned issues. While multiple solutions to a problem may be identified, one solution will be implemented. If the initial solution is not successful, alternate options (as per the data quality plan) will be applied.

Prior to reading this report, it is essential to view the associated Data Quality Report to gain an understanding of the dataset.

2. Issues within the dataset

The following is a brief list of the problematic features in the dataset:

- A high rate of missing data in the dataset.
- A high rate of Not A Number in the dataset.
- A high rate of unknown values in the dataset.
- A high rate of duplicate values in the dataset.
- Box plots (‘case_onset_interval’ and ‘case_positive_specimen_interval’) that have collapsed into vertical lines of dots.
- Histograms (‘case_onset_interval’ and ‘case_positive_specimen_interval’) that generate solely one vertical column.
- A bar chart (‘res_state’) that is entirely unable to visualise.

Each of these issues will be explored in more detail in the subsequent headings.

3. High Rate of Missing Data

While multiple columns of the dataset contain complete values, there are many incidents of the missing data in the dataset. From reading the CDC information, it can be surmised that much of this data is Missing Not At Random to ensure patient anonymity. This may complicate the selection of a solution, as it becomes more difficult to accurately model the missing data. The specific columns of concern are:

Column name	Column type	%missing data*	Bracket of data missing**
res_county	Categorical	5.85	Medium
age_group	Categorical	0.85	Low
sex	Categorical	2.38	Low
race	Categorical	12.83	Medium
ethnicity	Categorical	14.13	Medium
underlying_conditions_yn	Categorical	91.40	High
county_fips_code	Continuous	5.85	Medium
case_positive_specimen_interval	Continuous	46.52	High
case_onset_interval	Continuous	55.65	High

* denotes that for the purposes of this document, the %missing data value has been rounded to two decimal places.
**continue reading this section for further justification of this column and the rationale for each designation

The columns ‘case_onset_interval’, ‘case_positive_specimen_interval’ and ‘underlying_conditions_yn’ are of particular concern, given the extremely high percentage of missing data values present in these columns. While other columns such as ‘sex’ and ‘age_group’ are of less concern due to their lower rate of missing data values, if it is possible to correct this issue in the dataset, for the sake of completeness and improved accuracy it should be attempted. It is possible that ‘underlying_conditions_yn’ may need to be dropped entirely.

The columns that have a lower incidence of missing values present a relatively simple challenge. As there are few values with missing data, those rows could be removed/flagged and filtered out, with the remaining data analysed. Pandas contains a method - dropna() - to do exactly this. Identifying which columns should be considered to have a low rate of missing data is a challenge. As an initial point, performing this operation on columns with <5% missing values seems like a solid strategy. If this strategy needs to be further refined, columns with a higher or lower percentage of missing data values can be selected.

The aforementioned columns with a high rate of missing data are concerning. For the purpose of this analysis, a high rate of missing data is >40%. Initially, the importance of these columns was considered and whether dropping them from the dataset altogether would be an appropriate solution to the problem. However, considering that the objective of this task is to build a data analytics solution for death risk prediction, the knowledge contained in these columns would be useful. Therefore, while if all else fails this is a possibility, it is not optimal.

It must also be noted that there are columns with a Medium incidence of missing data. I have elected to handle Medium and High column classifications in a similar manner. In order to handle the missing values, Extreme Gradient Boosting (XGBoost) for Python seems like a strong solution as it has built in support for handling missing values. The XGBoost algorithm will try different directions to handle missing values and choose the direction that maximises the gain.

However, before using XGBoost (or another machine learning algorithm), the data needs to be imputed. I considered mean/median/mode imputation and K-nearest neighbours imputation as possible strategies. All of these have pros and cons. For the mean/median/mode, while it is simple and easy to implement, it may not be suitable for features with outliers as they could skew the mean or median. K-nearest neighbours considers information from neighbouring data points but is computationally intensive.

To solve this task, utilising the K-nearest neighbour imputation strategy followed by XGBoost seems a sufficient strategy. If it is not effective, use of other imputation techniques such as mean/median/mode will be considered.

4. High Rate of Not A Number

The graph creation (as seen in the Data Quality Report) revealed the presence of Not A Number (NaN) values in the dataset. Not A Number is a floating-point value that represents undefined/unrepresentable values, particularly in numerical operations. In the dataset, we can surmise that it refers to invalid data points. In order to avoid skewing results, the NaN values need to be acted upon.

NaN values are evident in the bar charts for ‘sex’, ‘race’, ‘ethnicity’, ‘age_group’ and ‘underlying_conditions_yn’. In some columns, they are quite numerous. This means that perhaps the most obvious strategy - dropping the affected columns - would cause the loss of a lot of data. Therefore, this solution is not suited to the problem.

The next solution to consider is forward/backward filling, which is when the NaN value is replaced with the last non-NaN value of the dataset. It is simple and preserves the temporal order of the dataset but if value errors have been entered in the dataset, it may further propagate those errors. However, given that the CDC documentation notes that they have already performed data cleaning which involved correcting such errors, we can be reasonably hopefully of avoiding these occurrences.

It is also possible to interpolate the NaN values, which is estimating the missing values based on surrounding data points. In some respects, it could be seen as similar to forward/backward filling. However, interpolation involves the use of mathematical techniques such as linear interpolation. It works well for sporadic missing data that does not follow a clear pattern but is sensitive to outliers. As there is quite a high incidence of NaN values in some columns, this strategy is not being selected.

The final strategy that has been considered is mode imputation. The NaN values will be replaced with the most frequently occurring value of the respective column. However, this may cause bias within the dataset, which is the reason for it not being selected.

From the above solutions, forward/backward filling has been selected.

5. High Rate of Unknown Values

Unknown values are evident in the column data for ‘sex’, ‘race’, ‘ethnicity’, ‘process’, ‘exposure_yn’, ‘symptom_status’, ‘hosp_yn’ and ‘icu_yn’. These are all categorical variables.

In order to correct the unknown values, a better understanding of the unknown values is needed. The Chi-Square test was conducted (see Jupyter notebook for details of code) and returned the following results.

- The Chi-Square statistic was 606.807686529997. A higher Chi-Square value represents a stronger association, so in this case we can see that there is a strong association between the variables.
- The P-value is 5.915849227345508e-115, which is very small. The P-value indicates strong evidence against the null hypothesis (that there is no association between the variables.

From the above, we can conclude that there is a strong association between the variables. Therefore, the missingness of one variable may be related to the missingness of another variable. This has led to considering two potential strategies: imputation based on missing variables and flagging missing variables.

Imputation based on missing variables would allow the usage of available information in the dataset. As the variables have strong associations with one another, the accuracy of the inputted variables should be relatively high. It would also maintain the relationship between the variables, which could be useful in future modelling tasks. However, it may also propagate errors if there is noise within the dataset, leading to false interpretations.

Flagging missing variables would allow the retention of information about the missingness, which could prove useful in analysing the data and ensuring data transparency. As there will be no imputation, it will also avoid introducing potential bias into the dataset and maintain the overall integrity. However, additional data handling will be necessary in order to account for the missingness, which may render the positives of this approach (retaining data integrity) moot.

Based on the above analysis, the strategy of imputation based on missing variables has been selected. While there is a risk of propagating errors, given that the association between the variables is strong, it is hoped that this can be avoided.

6. High Rate of Duplicate Values

During the initial data cleaning, multiple instances of duplicate rows were detected. The decision was made to retain the first instance of each piece of duplicate data and then drop the other instances, using the drop_duplicates() Pandas method.

While this appears to have resolved the issue of duplicate data, two additional checks will be carried out. For continuous data, scatterplots will be created to cross check the relationships before and after removing duplicates. As there are multiple continuous variables, this will allow the checking that the patterns between the variables remain broadly consistent. For categorical variables, the distribution of unique values will be examined through the use of ‘value_counts()’ to verify that there are no remaining duplicates.

The use of boxplots was also considered for examining data. However, as boxplots are utilised in other sections of this report, it was felt that an alternate strategy would be more beneficial to the analysis. Tests such as the Kolmogorov-Smirnov test to compare the variable distribution was also considered and remains a viable option. It was felt that the visualisation of the data through scatter plots may be less challenging to analyse. However, these tests remain a viable option should issues arise with the scatter plots.

7. Collapsed Box Plots

The issues with the box plot generation was noted in the Data Quality Report. Two of the generated box plots have collapsed and are visualising as a collection of dots in a straight vertical line. As such, it is impossible to see statistical summaries of quartiles etc.

The mean, median and quartiles of the ‘case_onset_interval’ and ‘case_positive_specimen_interval’ were printed out in the Jupyter notebook associated with this document (please see the Jupyter notebook for full details of code). For both iterations of the code, it was revealed that the values were mainly clustered around zero, with little or no variance in the data. Due to the lack of spread in the data, it is impossible for it to visualise as a box plot without collapsing into a straight line of dots. As box plots are designed to identify outliers and show the data distribution, they are not suited to these columns.

As such, an alternate visualisation or transformation of the data is required. A transformation could allow for the data distribution to be modified. An example of this is the cube root transformation, which would preserve the magnitude of the original data while producing a stable distribution that can be interpreted. However, some elements of the data may be negative and as such require transformations that can handle negative values, such as the Box-Cox transformation.

A density plot could be an appropriate alternative for visualising the data. It would offer more flexibility and customisation, meaning that bandwidth (and other parameters) can be adjusted to control the smoothness of the graph. However, the interpretation of the generated plot can be subjective and as such, the analysis of said material may be difficult to complete. A Cumulative Distribution Function (CDF) was also considered as a possible option, as it would allow the visualisation of the data, including the spread and variability. However, they can require a level of data analytics knowledge that may exceed this writer’s present capabilities. For this reason, the cube root transformation has been selected for this task (along with potentially using the Box-Cox transformation, if negative values need to be handled).

8. Undecodable Histogram Data

As noted in the Data Quality Report, two of the generated histograms could not be analysed to glean insights on the dataset. They were generating a single column. The two histograms are based on the same columns as in the previous section, so it is clear that this is the same issue as the previous one, just generated in another form of chart.

Although the points are clustered around zero with very little variance, it may be possible to visualise them as a histogram successfully. Using smaller bin sizes may manage to visualise the data successfully. Min-max scaling of the data to a specific range may also help to correct the issue. Both of these approaches will be attempted in order to solve the issue.

The usage of alternate visualisation plots was considered, such as a kernel density plot or a violin plot. However, these will not solve the inherent issue of the data being clustered by themselves. Therefore, they are not the optimal solution to the problem.

9. Indecipherable Bar Chart

The illegible bar chart for ‘res_county’ has been noted in the data quality report. It is entirely illegible. No bars have been generated and the text denoting the bars is a mass of black. As res_county contains 962 unique entries, it is entirely understandable that this cannot be generated on one bar chart.

Initially, the concept of increasing the bar chart size or using multiple bar charts to visualise the data was considered. However, given the number of unique values involved, this seemed an impossible task. Even if multiple charts were to be generated, the amount of charts that would be needed to render 962 values in a legible size would be so many as to overwhelm the reader.

Therefore, an alternate method of visualising the data has been selected. In conducting research for this data quality plan, multiple approaches were experimented with through code. All forms - scatter plots, dividing data into multiple bar plots, violin plots - proved unsatisfactory for working with such a plethora of data. Therefore, this analysis will evaluate the count through establishing the average incidence of appearances of the county in the data, then counting the number of outliers (both positive and negative) that appear in the data. This will be performed via Pandas methods such as value_counts().

10. Recap on Selected Solutions.

Correcting the missing values in the dataset.

- For columns with a Low incidence of missing data (<5%), those rows with missing values will be removed using dropna().
- Drop the column ‘underlying_conditions_yn’ due to the overwhelming amount of missing data in it.
- For columns with a Medium incidence of missing data (<40%) or with a High incidence of missing data (>40%):
    - Impute the missing values of the dataset using the K-nearest neighbours imputation strategy.
    - Utilise the XGBoost for Python on the dataset renderest by K-nearest neighbours.

Resolving the NaN values of the dataset.

- The variables in focus are ‘sex’, ‘race’, ‘ethnicity’ and ‘age_group’.
- Utilise forward/backward filling to handle the NaN categorical values.

Managing the Unknown values of the dataset.

- Utilise the Chi-Square test to evaluate the association between the variables (already conducted).
- Conducting imputation based on unknown variables in order to create a functioning dataset.

High Rate of Duplicate Values

- Much of this was already corrected during the initial data cleaning via pandas drop_duplicates().
- Continuous data: plot the data on scatter plots to ensure that the patterns of relationships between variables are broadly consistent.
- Categorical data: use ‘value_counts()’ to ensure that there are no duplicates remaining.

Collapsed Box Plots

- It was discovered in this report that box plots are entirely unsuitable for this dataset.
- The data will be interpreted through use of the cube root transformation method. If negative data values are encountered, the Box-Cox method has been identified as a suitable alternative method.

Undecodable Histograms

- Initially adjust bin size to evaluate if this solves the issue.
- If this does not work, attempt to visualise the data with normalisation.

Indecipherable Bar Chart

- Develop code to display the average incidence of each residential county in the dataset, as well as count the number of counties that exceed and fall below the average incidence.

These solutions are all subject to change and further exploration within the Jupyter notebook.