DATA423 Assignment 2: Data Processing & Glmnet

Introduction

This analysis report describes the findings of a comprehensive examination on the given COVID-19 dataset. The dataset describes 12 Predictor variables and one outcome variable, death rate, along with one variable to allocate testing or training data, and one identifier variable. This analysis aims to extract meaningful insights from the data through uncovering the relationships between predictor variables and outcome variable. This report will refer to the accompanying R Shiny app, in which sections on exploratory data analysis (EDA), data processing (specifically relating to missing data) and data modelling through Glmnet have been produced. Here discussed is an overview and summary of the dataset, followed by an analysis of the data processing strategies and Glmnet method.

Data Exploration (EDA)

A preliminary overview of the dataset within 'summary' of the EDA tab outlines the general features of the COVID-19 dataset. The variables CODE, POLITICS, HEALTHCARE_BASIS and OBS_TYPE are the 4 categorical (factor) variables describing the ID variable, type of government, type of healthcare system and train/test split variable respectively. The train/test split variable will allow for a 70:30 split, with a total of 452 observations. The remaining variables describe numeric predictor variables. The dataset has been read into the R shiny app in a way that converts common NA place holders such as –, -, -99, NA, N/A into the R value NA.

The Tabplot panel allows for visualisations on dataset homogeneity, allowing us to quickly visualise some main trends when sorting on either POLITICS (Figure 1) or HEALTHCARE_BASIS (Figure 2). When sorting initially on politics, we can observe how those instances with missing observations for politics also display some missingness for values where the proportion of the population is at or below 25 (AGE25_PROP), median age of the population (AGE_MEDIAN), infant mortality rate (INFANT_MORT), and some (yet fewer) instances of number of doctors per 100,000 (DOCS) and the type of healthcare system (HEALTHCARE_BASIS). When sorting on HEALTHCARE_BASIS, we observe strong missingness associated with the 'FREE' level of healthcare system to the HEALTHCARE_COST. It could be the case that this area of missingness in healthcare cost is directly related to the type of healthcare system - in this case, a 'free' healthcare system is likely referring to a fully government subsidised healthcare system where patients do not pay a fee themselves. As such, these missing values could be considered missingness not at random (MNAR).

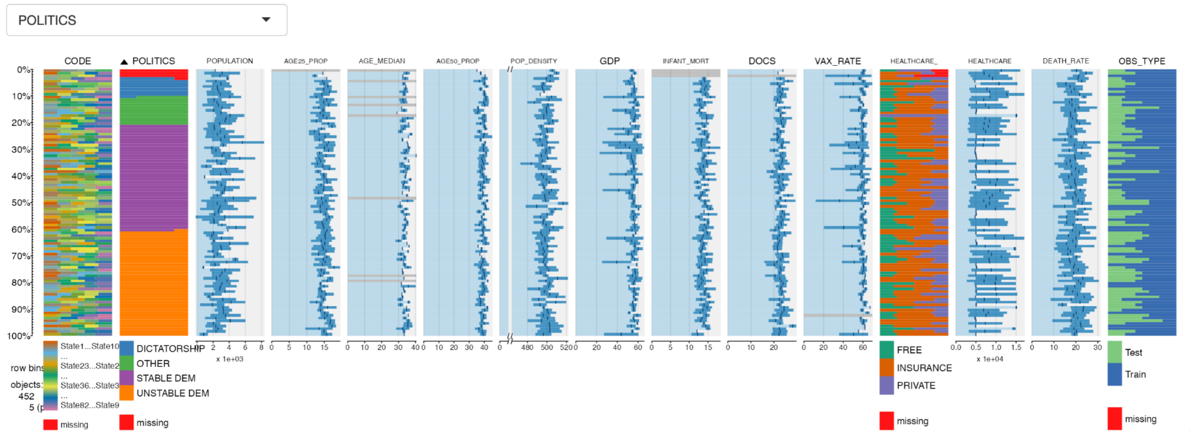


Figure 1: TablePlot of COVID-19 data, sorting on politics variable.

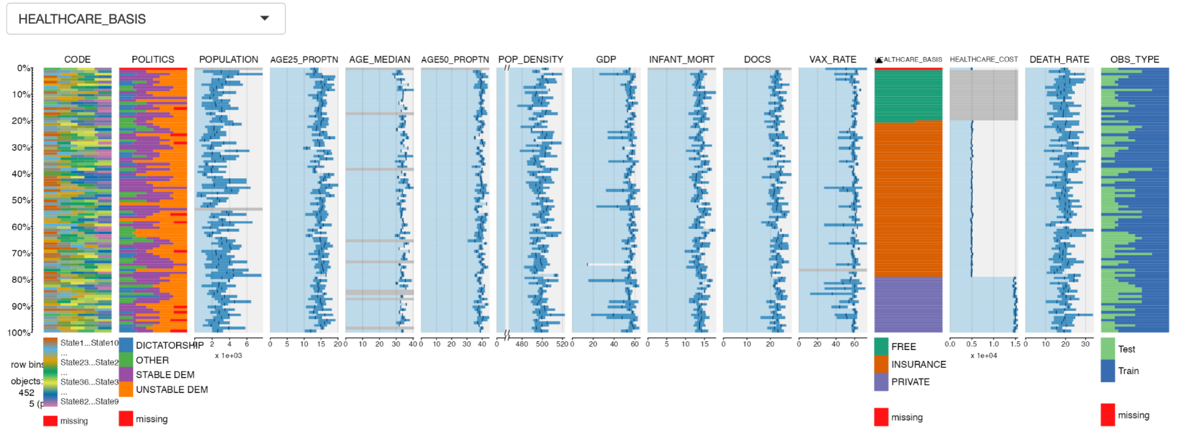
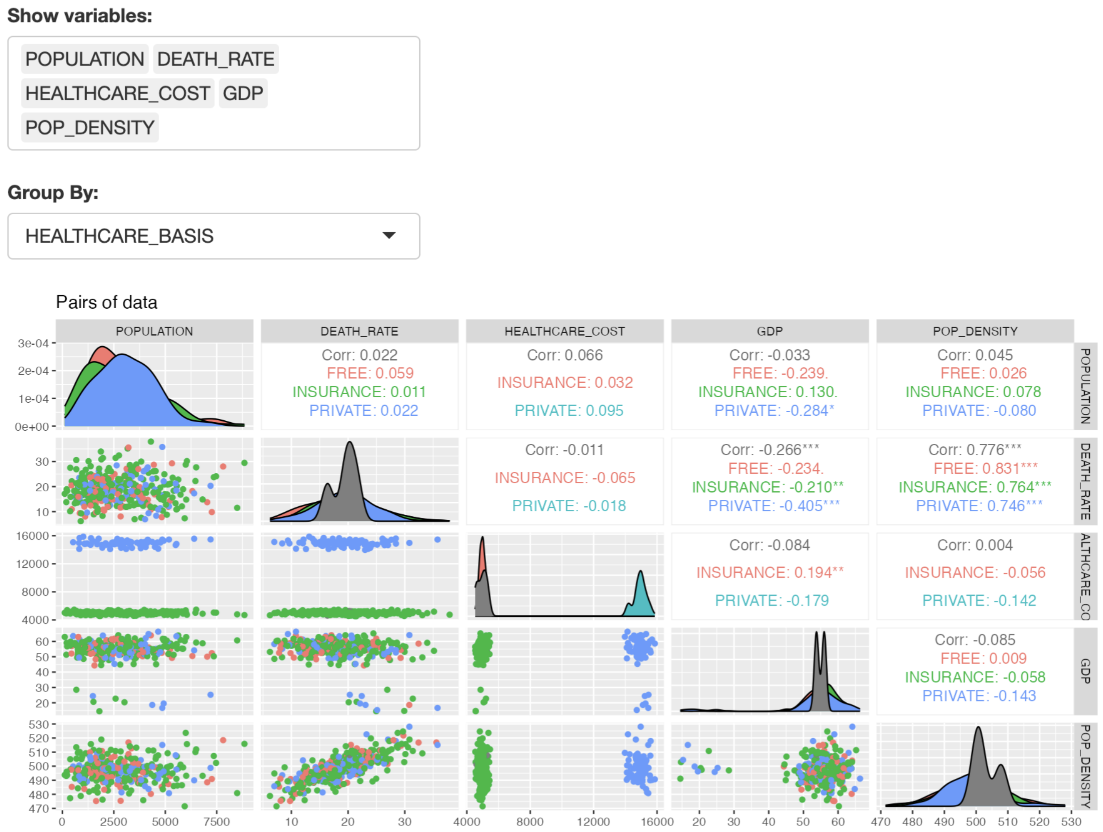


Figure 2: TablePlot of COVID-19 data, sorting on healthcare basis.

Dataset homogeneity can also be visualised through the Matplot tab, where we can observe that the variables within the dataset are all relatively homogenous when centering and scaling are employed. When exploring missing data in a more general sense, from the missing data tab it is difficult to deduce any apparent visually related patterns in the missingness of the observations. We can however see that the median age of the population (AGE_MEDIAN) looks to be more missing than other variables. The variables CODE, DEATH_RATE and OBS_TYPE all indicate no missing values, an important aspect as these variables are the ID variable, outcome variable and train/test split variable as previously mentioned.

When looking at the continuity of the numeric variables, the rising values chart in the 'Data Gaps' tab is able to visualise variables with discontinuity and continuity. The HEALTHCARE_COST variable appears to have the largest step within this chart, suggesting this variable is one that differs greatly across various countries, with the majority of countries having a healthcare cost of around 5000, and a smaller portion having healthcare cost values of around 15000. This can also be reflected in figure 2, where we observe a strong association between the private healthcare level and an increased healthcare cost when compared to the insurance healthcare level and associated healthcare cost. The rising values chart also indicates the majority of variables (excluding those CODE, DEATH_RATE, OBS_TYPE as previously mentioned) have missingness in their observations, as indicated by the shortened lines for each variable.

The Pairs Plot tab of EDA is able to show the Pairs Plot between various variables and is able to be grouped by the categorical variables POLITICS and HEALTHCARE_BASIS. The Pairs Plot allows for an overview of variables in a matrix format, through which identification of variables with strong relationships and those with potential outliers can be observed. Figure 3 shows the pairs plot of the outcome variable Death Rate, and predictor variables POPULATION, HEALTHCARE_COST, GDP and POP_DENSITY. These variables are interesting to visualise together, as we are able to see the distinct groupings associated with the cost of healthcare. When looking at the outcome variable, death rate, we can see that population density is the variable most strongly correlated, with a correlation value of 0.776, moderately strong. This positive association could be due to the fact that the density of a population is able to influence the spread of COVID-19, where more densely populated countries have higher levels of spread and subsequent higher rates of death.



Show variables:

POPULATION DEATH_RATE HEALTHCARE_COST GDP POP_DENSITY

Group By:

HEALTHCARE_BASIS

Figure 3: Pairs plot of COVID-19 Data with variables population, death rate, healthcare cost, GDP and population density, grouped by healthcare basis.

The correlation tab shows a Corrgram plot, where the relative correlation between numeric variables can be visualised. The corrgram allows for grouping with OLO, HC, GW, or no grouping, and is able to utilise

Pearson's, Spearman's or Kendall correlation, with the option of absolute correlation. Across all measures of correlation, the variable with the largest correlation to death rate was population density. This was commonly followed by the number of doctors. Somewhat surprisingly, vaccination rate did not appear to have a strong correlation with death rate, also exemplified in the pairs plot (not pictured).
The mosaic plot is contained in the EDA for completeness, however in this case it is not able to give further insights into the unexpectedly common or unexpectedly rare associations between the categorical variables with low cardinality contained in this dataset (POLITICS and HEALTHCARE_BASIS).

The Boxplot is able to show the relative distributions of numeric variables in the dataset, with each variable's summary statistics and outliers able to be shown and compared across one visualisation. Initially, the visualisation is set to show the Boxplots of each numeric variable with an IQR multiplier of 1.5, with outliers and centering/scaling (standardised), (Figure 4). Due to the varied ways each variable is measured, it is best to employ centering and scaling when visualising all variables in this way. Without standardising the scale on which these variables are measured, the HEALTHCARE_COST and POPULATION variables are shown while the other variables are truncated, simply due to their larger scale. When outliers and standardisation is employed, we can see that POPULATION, AGE_MEDIAN, AGE50_PROPTN, POP_DENSITY, GDP, INFANT_MORT, DOCS, VAX_RATE and the outcome variable DEATH_RATE all have outliers present at the IQR 1.5 level. When this IQR is increased to 2, only POPULATION, GDP, VAX_RATE and DEATH_RATE display outliers. When increased to 2.2 IQR and onwards, only GDP and VAX_RATE maintain outliers.

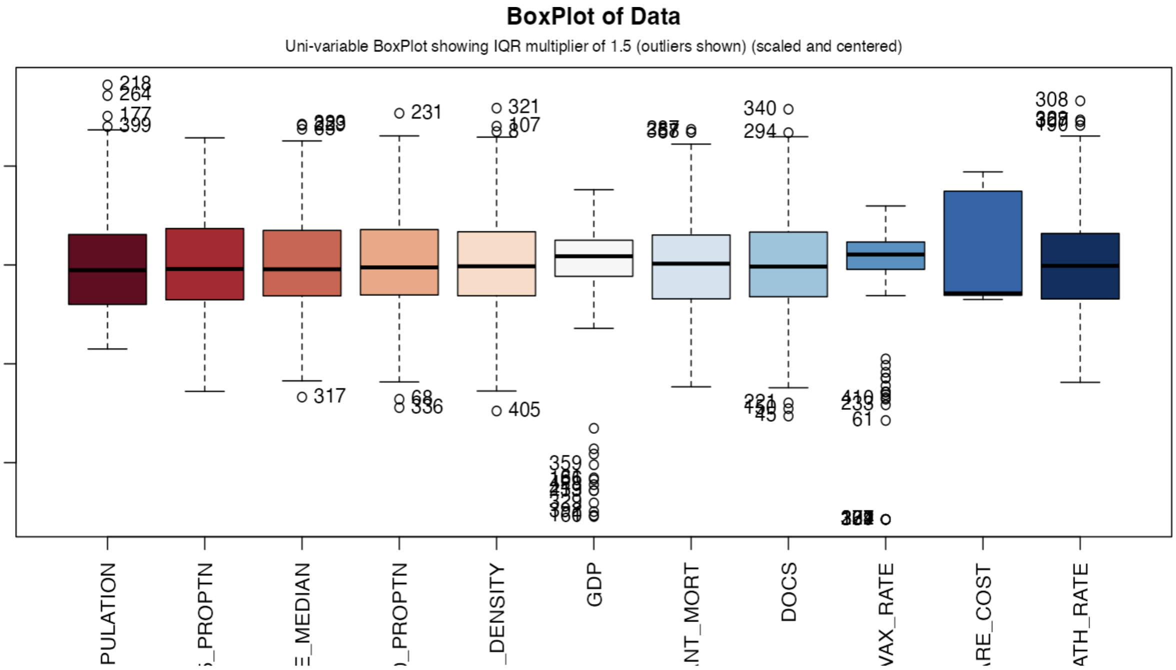


Figure 4: Boxplot of COVID-19 numeric data with outliers shown and standardised.

Data Processing Strategies

Missing Data

Processing this dataset is mainly explored within the ‘Data Processing’ tab of the R shiny app, with some final processing steps undertaken in the ‘Data Modeling’ section. Within data processing, an emphasis is placed on dealing with missingness in the dataset. The initial slider mechanism in the Missing Data tab is able to adjust the threshold of variable missingness, allowing the user to decide to remove variables where the ratio of missing values is greater than some threshold. The second slider mechanism is used to adjust the threshold of observation missingness, where observations are removed when the ratio of an observation’s missing values is greater than the defined threshold. Both mechanisms utilise a pMiss() function, which assesses the extent of missing data in a dataset, and are set to a threshold of 50% initially. This initial threshold removes the variable AGE_MEDIAN, the median age of the population, and has around 44% missingness. The AGE_MEDIAN variable had previously been found not to be significantly correlated with the outcome variable, DEATH_RATE, in the EDA section of this analysis and in the R shiny App.

Options have also been given to the user to impute the variables POLITICS and HEALTHCARE_COST. Imputation of missing POLITICS observations have been given due to the fact that the four available levels for the POLITICS variables are “Stable democratic”, “Unstable democratic”, “Dictatorship” and “Other”. Missing observations have the option of being manually imputed under the new level “None”, with the idea that the observations may be a ‘state’ within a country. This has been given as an option to the user to further explore the following models, with the understanding that this imputation may not be accurate - it is possible that the missing values of this variable are mistakenly omitted and should be categorised in another level. Another option is that any state or country, in order to qualify as such, must run under a type of government by definition, and the missing variables are an incorrect omission. In this case, the missing variables could belong to the “Other” group, however, guessing this value would introduce bias. Ideally, this issue would be followed up with a domain expert.

The option to impute HEALTHCARE_COST has also been given to the user after EDA processes found a link between the HEALTHCARE_BASIS variable and HEALTHCARE_COST variable, in which those instances where HEALTHCARE_BASIS was “Free”, there was a missing observation in HEALTHCARE_COST. Through this understanding, the missingness in HEALTHCARE_COST is suspected to be Missing not at Random (MNAR). The option to impute HEALTHCARE_COST is based on the logic that if the HEALTHCARE_BASIS level was “Free” and the HEALTHCARE_COST was missing, the value given to HEALTHCARE_COST is 0. Ideally, this issue would also be followed up with a domain expert, as at this stage we cannot verify that the cost of healthcare was left unrecorded because of its value.

Patterns in Missingness

In order to check the patterns in missingness, an rpart model has been constructed. This is in an attempt to check if variable missingness can be predicted, and thus whether the variable is not missing at random. The rpart model is able to generate an explainable, non-linear model that is able to suggest a relationship between observation missingness and variables. The results of this model can be observed in figure 5. The outcome shows that the POP_DENSITY, population density variable is the singular node produced with two leaves, whether population density is greater than 500 or not. As there is one node which produces a split between the data, suggesting missingness can be predicted, and so it is not entirely justifiable to judge the data missingness as missingness completely at random (MCAR), because the POP_DENSITY element does allow for differentiation in the dataset. As a split has been found, we cannot fully rule out missing at random (MAR), and could further suggest missingness may relate to population density in some way. It would be prudent to gather domain expertise on this matter.

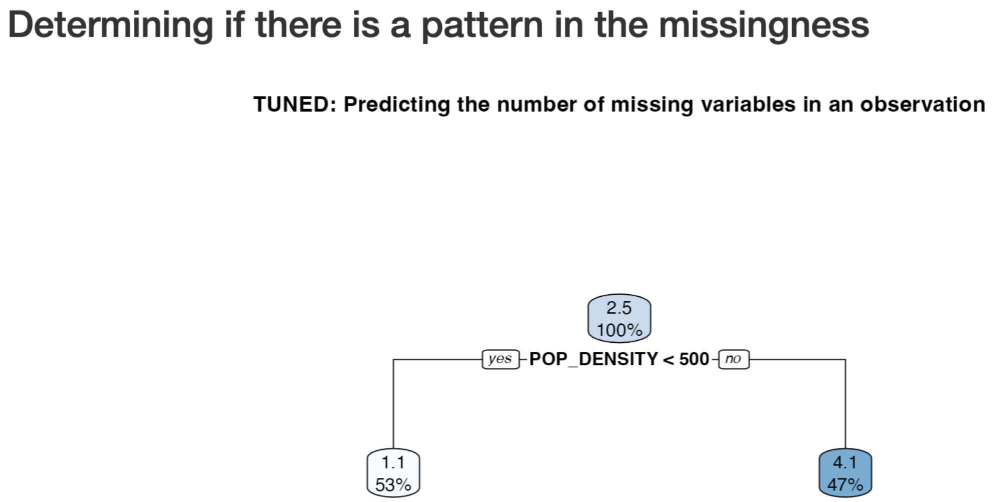

Determining if there is a pattern in the missingness
TUNED: Predicting the number of missing variables in an observation
2.5
100%
yes
POP_DENSITY < 500
no
1.1
53%
4.1
47%

Figure 5: Predicting the number of missing variables in an observation. Population density is able to split the data based on observation missingness.

Data Table

The processed data table can also be visualised in the Data Processing section of the R shiny App. This DataTable allows the user to observe the dataset with optional added imputations on the POLITICS and HEALTHCARE_COST variables, and allows the user to observe the inclusion of shadow variables.

Data Modeling (Glmnet) and Theoretical Performance

Glmnet Method

The Glmnet method is used within the train function from the Caret package in R, used in training machine learning models. Glmnet is useful with regression problems, where the data has many predictors and there is a continuous outcome variable (in this case, Death Rate). The Glmnet method fits models using regularised generalised linear models, where the regularisation comes about through the use of a maximum likelihood penalty. The regularisation parameter, Lambda, controls the strength of the regularisation. L1 regularisation adds a penalty based on the absolute value of the coefficients, while L2 regularisation adds a penalty based on the sum of squares of coefficients. These differences mean that L1 tends to force predictor selection in the model, while L2 forces correlated predictors to have similar coefficients. The elastic net addition to the Glmnet method introduces Alpha, which controls the elastic net penalty, that is, whether L1 regression (alpha = 1), L2 regression (alpha = 0) is used, or balanced somewhere between these values (Hastie, 2023). The advantage of using Glmnet is that it allows for the benefits of both L1 and L2 regression depending on the needs of the data and the model. The caret package allows for automated alpha and lambda tuning, and allows for cross validation while still using the Glmnet method.

Model Performance

A Glmnet model is trained in the R shiny app using pre-processed data. The pre-processed data is gathered from the dataset altered in the missing data processing stage, where the amount of variable and observation missingness has been addressed via the user-defined threshold. Should the user wish to impute POLITICS and HEALTHCARE_COST, this will also be included in the pre-processed dataset. This dataset is split into the train/test split according to OBS_TYPE before being used within the caret training function. A recipe is constructed, in which the CODE variable is updated to be identified as the ID role, and the OBS_TYPE variable is updated to be identified as the splitting variable. These will be excluded as predictors from the model. The recipe then allows for user-defined imputation methods, should there still be missing observations in the pre-processed dataset to impute. The user is able to select from a variety of imputation methods:

- K-Nearest Neighbour imputation (KNN). This method aims to impute missing variables based on the information of the 5 nearest neighbours.
- Average imputation wherein numeric variables are imputed using the median value and the nominal variables are imputed using the mode value. These have been selected due to their robustness.
- Bag Imputation. This technique, bootstrap aggregation, uses bagged tree models to impute the data. This method is able to deal with patterned missingness well.
- Partial Deletion. Partial deletion is present for completeness, however is not fully recommended due to this leading to fewer observations on which to train the model, and can introduce bias into the model, especially when considering some variables are indicating a pattern or reason to their missingness (POP_DENSITY and HEALTHCARE_COST specifically).
- No Imputation. This value will not return a model unless all missing data has been dealt with during pre-processing.

Once an imputation method is chosen, the user is able to train the model. The recipe will also dummy encode nominal predictors so as to be in suitable binary format for the Glmnet model, and add indicator variables for missing values for all predictors, prior to imputation.

Below the output of the trained model, the user will also be able to visually assess variable importance in the variable importance plot. It is important to note that the scale on this plot is arbitrary, and the threshold for considering a variable 'important' or not is thus arbitrary also. However, this is a good visualisation for the user to get an idea of which variables have contributed most strongly to the outcome of the Glmnet model. With the COVID-19 dataset, it is interesting to note that the variable with the highest importance is found to be the missingness indicator for POP_DENSITY, again reflecting earlier findings that there may be a pattern to this missingness (MAR).

The user will be able to assess the performance of their model visually through the Model Performance tab. The 'Generate Plot' icon will generate a model of the DEATH_RATE predictions of the COVID-19 testing data. This visualisation (Figure 6) shows the predicted vs actual results for the testing data based on the previously created Glmnet model. This visualisations allows the residuals to be understood on the scale of either axis. The optimal model was ultimately selected using the smallest RMSE.

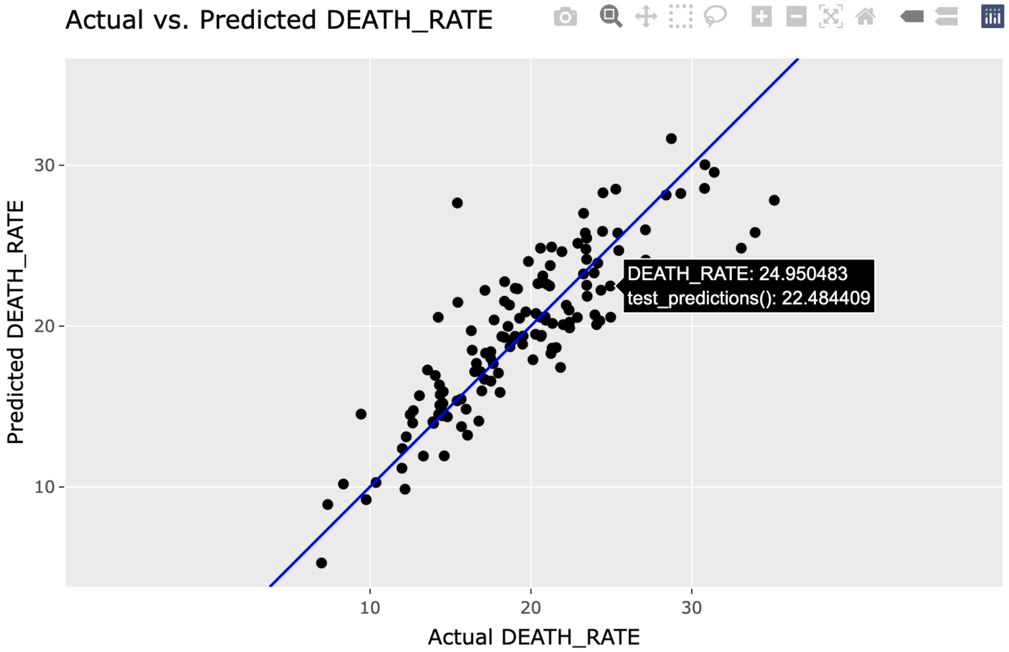


Figure 6: Actual vs Predicted Death_Rate of COVID-19 Data based on KNN imputation with 50% threshold for variable and observation missingness.

The Glmnet model's performance on future unseen data can depend on a number of factors pertaining to both the model itself and the underlying assumptions of the model based on how the training data has been processed. When considering the Glmnet model in general, theoretically the Glmnet use of regularisation makes this model better at generalising to unseen data than a model without regularisation, as this allows the model to balance model complexity (number of predictors) and the fit to the training data. The use of caret's automated cross validation and hyperparameter tuning (of alpha and lambda) will also help to improve the model's generalisability to future data.

When considering the pre-processing of data in this case, the model's performance will ultimately depend on how the user has chosen to pre-process the training data for the model. Should the user have imputed POLITICS and HEALTHCARE_COST manually, this may introduce some bias into the model, as the user would be making the assumptions as outlined above without having verified the true nature of these missing variables with a domain expert. The thresholds for variable and observation missingness chosen by the user will also impact the bias of the model, as removing a large proportion of variables and observations from a dataset can result in a weaker model that is unable to generalise well due to being trained on insufficient data.

The choice of imputation method will also impact the generalisability of the model. For example, if partial deletion is used, this will remove those observations with missing values, resulting again in a weaker model. This is especially true if the user did not choose to remove variables with excessive missingness in the pre-processing stages, as every observation with this missingness would be ultimately deleted. Should the user have retained the greater than 50% missingness variable threshold, this would allow for the training dataset to only include those variables with less than 50% of the observations missing to be included, thus increasing the total number of observations included in the final dataset.

Residual analysis

The residuals of the model are able to be visualised through the use of Boxplots in the Residual Box Plots tab. Here, we can visualise training residuals, testing residuals, and both combined on one Boxplot. In the majority of modelling cases (with different imputation and pre-processing methods), the training residuals (Figure 7) show fewer outliers in the boxplot when compared to the outliers of testing residuals (Figure 8).

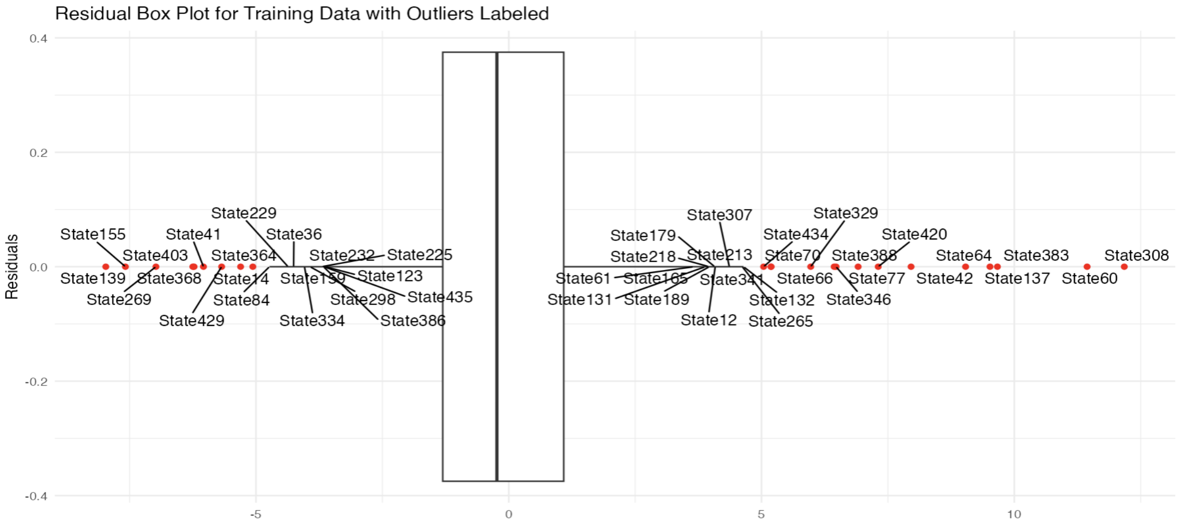


Figure 7: Residual BoxPlot for Training Data with KNN (K = 5) imputation.

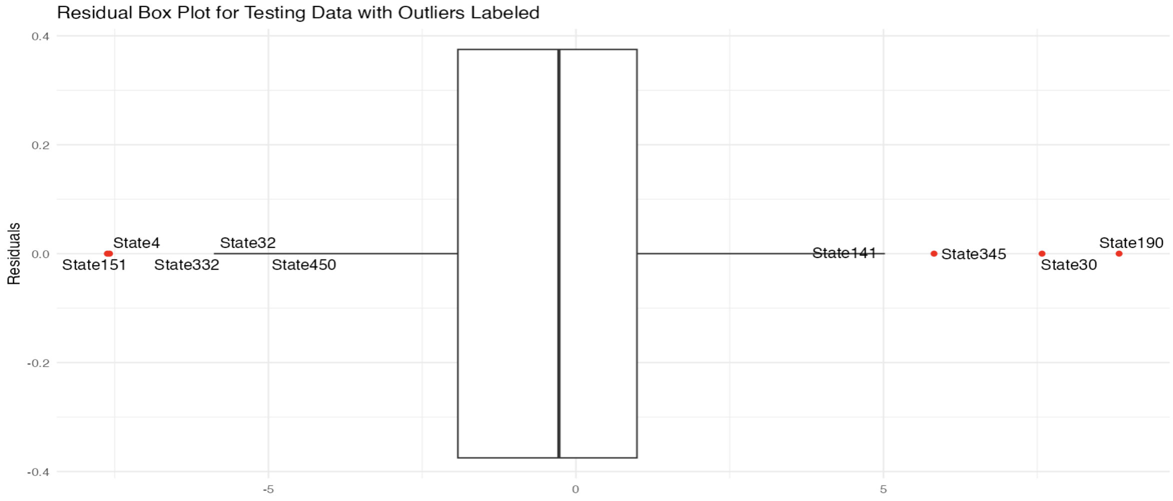


Figure 8: Residual BoxPlot for Testing Data with KNN (K = 5) imputation.

This curious result indicates that the model is perhaps underfitting to the training data. Another possible explanation for this result is that data leakage has occurred through the pre-processing of the dataset, or bias has otherwise been introduced during pre-processing and training. If the model is underfitting the training data, this would mean that the model is too simple, suffering from high bias and is not capturing the underlying patterns in the dataset. It could be the case that too many variables are being imputed in the training process - guessing missing observations even through imputation could be introducing unintended bias to the model. This is especially true when considering some variables are in need of domain expertise to verify the nature of their missingness, especially HEALTHCARE_COST and POP_DENSITY, suspected MNAR and MAR respectively. Imputation of these variables could be causing bias to be introduced to the model. It is therefore likely that for any future unseen data, the model in this case may not be able to capture all the underlying patterns in the dataset. Future work to remedy this would involve investigating the cause and patterns within the missingness of POP_DENSITY, hopefully with the aid of a domain expert.

Works Cited

Hastie, T. (2023, March 27). *An Introduction to `glmnet`* (J. Qian & K. Tay, Eds.). Glmnet.stanford.edu. https://glmnet.stanford.edu/articles/glmnet.html