Exploratory Data Analysis

HARVEY KWONG and JACOB DEROSA, University at Buffalo, USA

1 Introduction and Problem Statement

The rise of antibiotic resistant infections poses a significant challenge to global public health, and gonorrhoea is one of the most concerning pathogens due to its rapidly increasing resistance to available treatments and its difficulty to be detected. This project focuses on predicting antibiotic resistance in Neisseria gonorrhoeae, the bacteria responsible for gonorrhoea, using subsets of its DNA sequences as predictive features. The primary objective of this project is to explore the relationship between bacterial DNA segments and resistance patterns, and to identify key genetic markers that could possibly predict resistance. By performing data cleaning and exploratory data analysis on a dataset containing DNA sequences and resistance outcomes, we aim to address the following question: Which DNA segments are most associated with antibiotic resistance?

This project's contribution is very important as it helps to clarify biological data, making it easier for future studies to focus on effective solutions. By gaining a deeper understanding of the data and identifying key variables, this project serves as a step towards more advanced research on antibiotic resistance.

2 Data Sources

We utilized the "Predicting antibiotic resistance in gonorrhoea" dataset from Kaggle: (https://www.kaggle.com/datasets/nwheeler443/gono-unitigs/data).

3 Data Cleaning/Processing

We have completed 11 total unique data cleaning and processing steps:

1. Removed all rows that were NaN in our target label field - Our target labels we aim to predict down the line are: azm_sr, cfx_sr, and cip_sr. If the labels were missing, then we cannot use that row.
2. Removed unused columns - The dataset contained multiple types of resistances. Because we only focus on 3, the rest cannot can be discarded safely. We also use this step to remove the year column.
3. Removed duplicate rows - Duplicate rows have the ability to skew our data. As such we removed all duplicate rows.
4. Removed all symbols - Our data are either numerical or categorical (text). There is no need for symbols which will only cause problems down the line.
5. Turned all non-numeric data within numerical columns into NaN, to be processed later - We have columns that are supposed to contain only numerical data, however, there are rows where there are letters in there. In this step, we turn them into NaN.
6. Cast all numerical columns into float32 - To establish precision.
7. One hot encode categorical columns - Some columns such as "country" should be one hot encoded as we would want numerical representations of all our data for modelling.

Authors' Contact Information: Harvey Kwong, harveykw@buffalo.edu; Jacob DeRosa, jderosa3@buffalo.edu, University at Buffalo, Buffalo, New York, USA.

8. Handle "Beta.lactamase" special case - It is a numerical column but appears to be categorical in nature. As such we cannot impute missing values like with the other numerical columns. In this step, we set all missing entries to 0, and one hot encode the feature like with the non-numerical columns.
9. Split dataframe into Train/Test - We will be doing additional data processing such as normalization and imputation later on. As such, we need to make the split beforehands for good practice in preventing leakage.
10. Impute missing values in numerical columns - For every NaN in the numerical columns, we use skew based imputation to generate a new value that can substitute the NaN. This ensures that we have enough data to work with as we do not need to discard the entire row.
11. Normalize all numerical columns - We do columnwise normalization on numerical columns. This is to ensure that we do not have extremely large values. This is important for modelling down the line.

4 Exploratory Data Analysis

5 Title Information