Predicting a Biomanufacturing Process

Project Report

Olgu Altintas

14.05.2023

Project Description:

I have been provided with multiple datasets and have been requested to develop a model that is capable of predicting the output of a biomanufacturing process.

What is the working flow?

This project is composed of three primary sections:

1. Data Preprocessing
2. Feature Engineering
3. Applying Model

- To provide justification for the decisions made throughout this project, several figures are included in this report.

Working flow

1-) Data Preprocessing

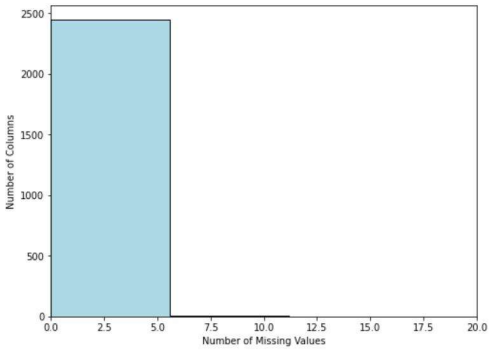
The first stage in this project is data preprocessing, which involves a series of steps to obtain the final versions of the data frames that can be used in models.

The data comprises two primary components: batch_df and observations.

Initially, the batch_id column was considered irrelevant to the task and was therefore removed before proceeding with any further data processing operations.

The next step involved addressing missing values in the data. This was accomplished through data visualization and analysis, which revealed that 14 columns contained a significant number of missing values and were therefore dropped from consideration.

```
Number of columns with missing values greater than 2%: 18
Number of columns with missing values greater than 5%: 14
Number of columns with missing values greater than 10%: 14
Number of columns with missing values greater than 20%: 14
Number of columns with missing values greater than 50%: 14
Number of columns with missing values greater than 80%: 14
```



After reviewing the contents of the batch_df, I observed that there were a total of 66 columns with object data types that required encoding. Given the large number of columns already present, I chose to use label encoding rather than one-hot encoding in order to avoid adding more columns.

During the data pre-processing, it was important to minimize the number of columns and retain only the most crucial ones

With regards to the observations, my initial step involved creating a dataframe to store them. To achieve this, I wrote a function that extracted five key pieces of information from each observation CSV file. These five pieces of information were then assigned to individual columns in the dataframe.

These five columns were:

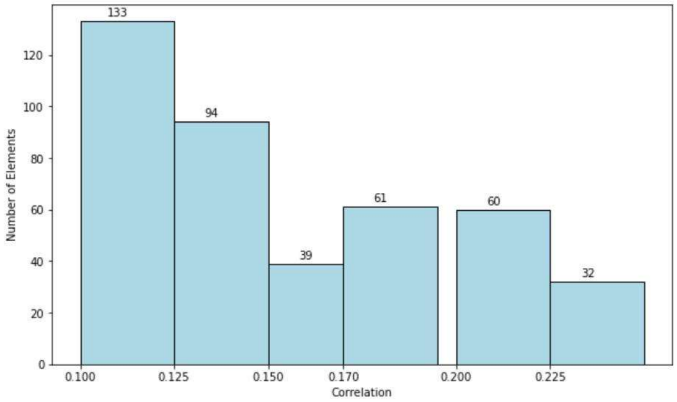
1. mean of value_aprrox
2. maximum of value_aprrox
3. minimum of value_aprrox
4. volume approx in which value_aprrox was maximum
5. volume approx in which value_aprrox was minimum

The resulting observation_df contained a total of 100 columns. To further streamline the data and reduce dimensionality, I applied Principal Component Analysis (PCA) to this dataframe. This allowed me to obtain a final version of the observation_df.

Having obtained final versions of batch_df and observation_df, I merged the two dataframes together and proceeded to perform feature engineering.

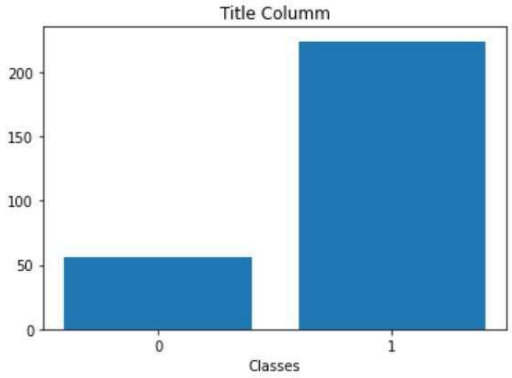
2-) Feature Engineering

During the feature engineering stage, I needed to apply a statistical approach to the data in order to identify and retain only the most important columns, while discarding those that were irrelevant. After attempting several different approaches, I decided to utilize a correlation threshold.



To select the most important features, I applied a correlation threshold of greater than 0.25. This threshold allowed me to identify the most relevant features and ultimately retain 159 columns for further analysis.

Before proceeding with the model building process, I created a visualization to examine the distribution of the target column. As there were relatively few rows in comparison to the number of columns, I wanted to ensure that there was no mismatch in the data. However, there was a mismatch!



3-) Applying Model

The final dataset comprised 280 rows, which were partitioned into 224 rows for training and 56 rows for validation purposes.

As mentioned above there was a mismatch problem. To address the issue of a mismatch in the data, I applied the Synthetic Minority Over-sampling Technique (SMOTE) to the training data, resulting in a total of 360 rows.

Subsequently, I tried several machine learning models and neural networks to determine the most effective result. Through this analysis, I discovered that Logistic Regression provided the best balanced classification rate (BCR) score without overfitting the data.

Average BCR score was 0.886 after 10-fold cross validation.