Karthik Sabhanayakam
January 20th 2024

Audax Labs Internship Final Report

Introduction

From October 2023 to January 2024, I have been working on a project about Water Fault Detection. Using industrial sensor data compiled from numerous sources, I have been able to implement a machine learning model to successfully predict whether the water has an excess of unwanted particles or not.

Preprocessing

The data was compiled from many sources, and as such was split among eleven different Excel sheets. Additionally, each of the sheets needed to undergo many steps of preprocessing to clean it. I had to remove unneeded categorical values, fill in an excess of missing data, and scale all the data to appropriately represent the quantitative values that the sensors detected.

The first step in preprocessing, was to compile all of the data sources into one large Pandas dataframe. Then, began the Exploratory Data Analysis (EDA), where I used features like ". describe()" and graphs like boxplots and bar charts to try to get a complete understanding of the data. Afterwards, I was able to truly begin preprocessing. I removed the dataframe columns that had an excess of missing values, and the ones that had an excess of zeros, by deciding on a threshold of one half the column length. Then I filled in all of the remaining missing values with scikit-learn's built in imputer, selecting the mean as my strategy based off my visualization of the data in the EDA. Next, I dropped the four unwanted categorical values and realized that the current dataframe was severely unbalanced, favoring the negative cases fifteen-fold. I then imported the Synthetic Minority Oversampling Technique, or SMOTE for short, (an oversampling technique) from the imbalanced-learn library. After applying it to the dataframe, I got the achieved result of having the same number of positive and negative cases (1448 and 1448). I saved that dataframe using Pandas and was ready to begin the model selection stage.

Model Selection

In this stage, I wanted to incorporate as many models as possible to seek the best accuracies for predictive performance. Eventually, I decided on using the K-Nearest Neighbors, Decision Tree, Naïve Bayes, Support Vector Machines, XGBoost, and Random Forest classifiers. First, I imported all the required libraries and set each of the models into a JSON list that stored the model's instant and some base hyperparameters. Then, I created a function that would iterate through the list and use the GridSearchCV library to optimize each of the models hyperparameters. Then I would print out the accuracies, mean squared error, R^2 score, and ideal hyperparameters, to identify the best performing model. After roughly thirty minutes, the function had finished running and I found the best performing models to be XGBoost and Random Forest classifiers, with accuracies of 92% and 87% respectively. I then saved the model with its corresponding hyperparameters with Pickle for usage in the web app.

Conclusion

The model has since been deployed with a web app hosted on Streamlit. Try the API out yourself with the Docker container or check out the code in the GitHub repository.

I would like to thank my mentor, Someet, for his advice and guidance throughout this internship. His support has been instrumental in making this experience not only educational but also highly enjoyable. I am truly grateful for the opportunity to learn and grow under his mentorship.