**Predictive Analysis of Dissolved Oxygen Levels**

Evan Moore

101185201

2023-11-27

**Introduction**

The Department of Fisheries and Oceans (DFO) mandate focuses on sustainably managing fisheries and protecting aquatic ecosystems. DFO’s motivation for studying dissolved oxygen levels in estuaries is due to the potential negative impact of nutrient loading on aquatic eco-systems, including impacts on fish and aquaculture through the destruction of habitat and/or the development of hypoxic or anoxic zones. With a dataset that uses dissolved oxygen level as a proxy, the tools of data science can help us better understand the distribution and intensity of eutrophication in estuaries, and possibly predict its future occurrence.

**Project Objective and Methodology**

This project is examining how to effectively classify future (i.e., for the next day) dissolved oxygen levels within four different estuaries. Specifically, given that we are looking for significant changes in dissolved oxygen levels, we will be classifying future data as low, medium, and high by considering anything below the 25th quantile as low, anything above the 75th quantile as high, and anything in between as medium.

The data science methods that I have chosen for the project are preprocessing techniques and supervised learning. To begin the preprocessing, I examined the standard deviations for each time-series to understand their variability. This will help determine how easy or difficult it will be to classify future values in the time-series. Based on this preprocessing, I chose the two time-series with the lowest standard deviations and two with the highest standard deviations. I did not check for missing values since the data provided was already cleaned. I then smoothed these four columns using a low pass filter so that it would remove the influence of tidal fluctuations.

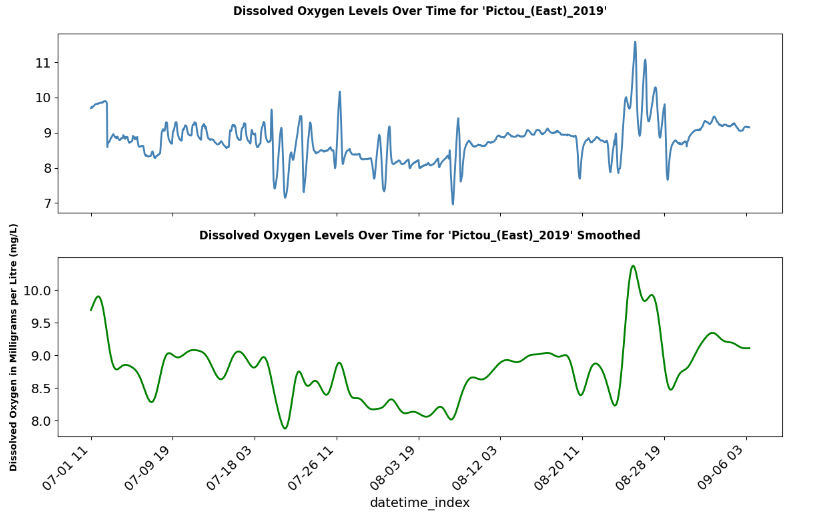


Figure : An example of a low pass filter applied on a time series with a low standard deviation.

A graph of oxygen levels

Description automatically generated

Figure : An example of a low pass filter applied on a time series with a high standard deviation.

Following this, I created a time-series cross-validator using an expanding window size which will allow us to train the data over multiple training and testing sets. We then started to loop through a grid search of hyper-parameters. These are defined hyper-parameters based on the current time-series under analysis. Inside this grid search, the data was split into training and testing sets. The data was then rearranged to fixed size windows such that the features are now the recent windows which will be used to predict the target, (i.e., future values we aim to predict). The target was then converted from numerical values to the categorical values of low, medium, and high as described above. The model was then fit, and the results were recorded.

The three different supervised learning techniques that were tested were: k-nearest neighbours (KNN), support vector machine (SVM), and random forest classifier (RFM). Inside the grid search, this analysis examined various hyper-parameters for each of these classification models. The hyper-parameters were initially randomly chosen at the begin which are the values discussed over the next three paragraphs. The hyper-parameters were then tuned further for each time series to get the best results.

For KNN, this analysis first varied the number of input data points - 48, 72, 96, and 120 to determine the best input size for the model (this was performed for all three classification models). Additionally, I considered whether standardizing the data with the standard scaler function impacted the model's performance (this was performed for all three classification models). Another hyperparameter which was tuned was the count of nearest neighbors, with values of 5, 7, 9, and 11 being evaluated. Finally, I explored different metrics for calculating the distance between instances, specifically the Euclidean Distance and Dynamic Time Warping.

For SVM, the first parameter tuned was the kernel type, which is crucial as it transforms the input data for a decision boundary to be found. The values tested were both the linear and radial basis function (RBF) kernels to see which would perform better. The second hyperparameter was the C parameter, which controls the degree to which the classifier penalizes misclassified data points.

For Random Forest Classifier, the first parameter I varied was the number of estimators, which dictates the number of trees in the forest, testing values of 100, 200, and 300. Secondly, I explored the maximum depth parameter, which sets the maximum depth for each tree in the forest. By trying out values of none, 10, 20, and 30.

Lastly, inside this grid search, we calculated balanced accuracy score in each iteration to determine which models performed the best. Also a confusion matrix was made for each classification to know how well each classifier does at predicting each class (low, medium, or high). Risks for the project include data collection errors due to faulty equipment and faulty analysis due to limited data. Consulting with experts can help mitigate these risks by ensuring consistency and accuracy in results.

**Project Timeline:**The following table outlines the program plan for this project. All tasks were completed within the timeframe specified.

| **Date** | **Description** |
| --- | --- |
| November 23rd | 1. Choose the non-stationary and stationary columns to be analyzed. 2. Preprocess both data sets so that forecasting techniques can be easily applied to them. 3. Create a column for each feature we are forecasting which classifies the data into high, medium, or low. 4. Create a training set and testing set of the data which will be used later to test the effectiveness of the forecast. 5. Create a basic forecast and calculate its effectiveness by comparing the forecasted values to the actual values in the testing set. 6. Use this as the baseline for future forecasts to see if they do better or worse. |
| December 6th | 1. Implement supervised learning techniques on the stationary and non-stationary features. 2. Graph the forecast and the actual values. 3. Create a PowerPoint to present the findings on forecasts that were implemented and which forecasting technique performed the best for each feature. |
| December 15th | 1. Try tuning the hyperparameters even more to try and get the best mean balanced accuracy score. 2. Document the code in Collab such that it can be understood by anyone. 3. Prepare a report which discusses the business motivation and objective for the project. It will also explain the data science techniques that were used on the data and the results. Lastly the paper will propose future work which can be done. |

**Results:**

After performing the grid search for each of the four time-series and looking at their mean balanced accuracy score and their standard deviation, these are the best classification models for each time-series respectively.

For the Pictou East 2019 time-series, which is the time-series with the lowest standard deviation, the best classification model was SVM. This model produced a balanced accuracy score of 0.656 and a standard deviation of 0.248. The parameters which this model used was an input size of 48, a linear kernel, and a C value of 300. Using a confusion matrix, this model was accurate at predicting the low class with an accuracy of 71%, then medium with an accuracy of 66% and lastly high with an accuracy of 62%.

A blue squares with numbers and text

Description automatically generated

For the Pugwash 2019 time-series, which is the time-series with the second lowest standard deviation, the best classification model was also SVM. This model produced a balanced accuracy score of 0.853 and a standard deviation of 0.174. The parameters which this model used was an input size of 96, a linear kernel, and a C value of 200. Using a confusion matrix, this model was the most accurate at predicting the low category with an accuracy of 88%, then medium with an accuracy of 65% and lastly high with an accuracy of 56%.

A blue squares with white text

Description automatically generated

For the Kildare 2013 time-series, which is the time-series with the second highest standard deviation, the best classification model was SVM. This model produced a balanced accuracy score of 0.662 and a standard deviation of 0.383. The parameters which this model used was an input size of 48, a linear kernel, and a C value of 50. Using a confusion matrix, this model was the most accurate at predicting the low category with an accuracy of 93%, then medium with an accuracy of 73% and lastly high with an accuracy of 21%.

A blue squares with white text

Description automatically generated

For the Mill 2019 time-series which is the time-series with the highest standard deviation the best classification model was also SVM. This model produced a balanced accuracy score of 0.673 and a standard deviation of 0.264. The parameters which this model used was an input size of 48, a linear kernel, and a C value of 300. Using a confusion matrix, this model was the most accurate at predicting the low category with an accuracy of 88%, then medium with an accuracy of 65% and lastly high with an accuracy of 56%.

A blue squares with white text

Description automatically generated

**Challenges:**

Challenges that were faced, were primarily the limited data set which makes it very difficult to predict a whole day ahead. When trying to classify future values, the more data that the model can work with the better the model could perform as it has a better picture of how the data behaves.

**Summary**:

As seen in the results, we were able to build four different classifier models for various time-series which were sampled from both stationary and non-stationary time-series. These models for the most part did not produce the most accurate results, but as explained in the challenges section this can be explained more so by the lack of data than the actual models themselves. It would be interesting to test these with the longer time series to see how the models would perform with this additional data.

Some future work which can be examined are firstly, combining multiple time-series from the same estuary and then perform these classification models. Once this has been done this could also open opportunities to implement deep learning algorithms to classify future values. The reason this was avoided in the original analysis was because deep learning algorithms shown not to perform as well when working with a small amount of data as in the current case.