UNIVERSITY OF EXETER

Computer Science Department

M.Sci. Mathematics and Computer Science
Computer Science Department

A Comparison of Machine Learning and Statistical Techniques for Crossdating

Candidate

Ursula Mennear

Student ID 257534

Supervisor

Dr. Johan Wahlström

University of Exeter

Co-supervisor

Dr. David Reynolds

University of Exeter

Academic Year
2022/2023

Abstract

This project investigates different techniques for crossdating. Crossdating is a method used in dendrochronology (the study of tree rings) and involves matching tree ring measurements between samples to determine if they lived at the same time. It has been used to establish multi-millennial histories of past ecosystems and climate dynamics. These records have been fundamental in developing our understanding of modern climate change. This project aims to compare a machine learning-based technique and a statistical-based technique to see which can facilitate crossdating by identifying consistent patterns between tree rings. Crossdating processes have, primarily, been performed manually and are, therefore, vastly time-consuming. However, some valuable programs have been developed to automate parts of the crossdating process. The statistically based method uses pairwise lead-lag analysis, a popular method used in previous programs. The machine learning method uses multilayer perceptron classifiers because they have been successful at similar problems in other fields. A Graphical user interface was constructed in order to run both methods efficiently. Both methods were analysed against success criteria and where possible compared with a state-of-the-art crossdating program. From this, the statistical method was the most successful of the two created. This was due to its accuracy, speed and ability for the user to customise parameters used in the method. Despite being less successful the machine learning method still produced good results which could be improved upon with further research. Additionally, areas for improvement were identified including testing data containing errors to determine how robust the methods are.

	Yes	No
I certify that all material in this dissertation which is not my own work has been identified.	☑	☐
I give the permission to the Department of Computer Science of the University of Exeter to include this manuscript in the institutional repository, exclusively for academic purposes.	☑	☐

Contents

List of Figures

List of Tables

Glossary

Crossdating A procedure which matches the pattern of annual growth rings of an organism to another of the same species alive during overlapping time periods

Sample A set of ring widths measurements from one organism.

Chronology A collection of individual samples which have been correctly dated and overlap in time with each other.

Master Chronology A collection of samples forming a chronology that has been averaged using a robust mean to form one signal.

Dendrochronology The study of tree growth rings to ascertain when each ring was formed.

Sclerochronology The study of calcified structures of bivalves, fish and corals to ascertain when each ring was formed [1].

CSV Comma Separated Values

MLP Multilayer Perceptron

GUI Graphical User Interface

1 Introduction

Crossdating is a procedure which matches the pattern of growth rings [2]. Organisms such as trees and bivalves grow annually with a new layer or ring forming every year of the organisms' life [3][4]. Samples of the same species grown in similar locations are compared to form a precise time series, identifying the corroborating years through similarities in the pattern of their ring sizes [5]. Growth patterns reflect the environment; hence, species grown in a similar location will have a similar pattern. The objective of crossdating is to create a chronology, which is a collection of crossdated organisms that were alive at the same time and each of their growth rings are labelled with the correct year. The collection of samples forming a chronology can then be averaged using a robust mean to form a master chronology with a stable signal to date more samples [6]. The study of Dendrochronology uses the tree's growth rings to determine characteristics of the past [5]. Crossdating is used to create chronologies from living trees, which can live for hundreds of years [7], and can be extended using preserved dead trees [3]. Dendrochronology is used in multiple fields such as climatology and dating wooden panels in art history [8]. It is particularly important for climatology because modern climate observations only date back to the late nineteenth century, during which industrialisation highly impacted the climate [9][10]. Crossdating gathers crucial information on how the climate operated before human interaction to predict how it may behave in the future [11].

Chronologies have allowed scientists to discern unique insights, such as changes in atmospheric conditions (cold periods), fluctuations in the water table (flash floods and droughts) and forest fires [12][13][14]. For this to be accurate, chronologies should be deemed to have zero errors, known as "absolutely dated" [15]. This means that every ring must be assigned to the correct year, and all erroneous data, such as missing or added rings, have been found and corrected. An "absolutely dated" chronology is necessary in order to fully utilise the climate data hidden in the tree rings, such as interesting patterns or trends [16]. In order for crossdating to occur the growth rings must be preprocessed and standardised so that correlation methods can be applied. As crossdating techniques are used to determine environmental trends, it is essential to first eliminate growth variability that is not caused by environmental factors in order to examine useful variations in growth ring width [17]. These factors include growth trends which are age-related. Detrending facilitates the removal of the long-term growth slowdown associated with these age-related growth trends while maintaining growth that is driven by the environment [18].

The process of dating detrended samples is labour-intensive, and, as a result, the creation of an "absolutely dated" time series is very expensive and time-consuming. Especially since historically, crossdating was a manual process using a graph called a skeleton plot [19]. However, since the 1980s, many successful statistical-based programs have been developed to help automate parts of the process [20]. Current statistical methods predominantly rely on the same statistical procedure called pairwise lead-lag analysis. Figure 1.1 depicts a timeline of the development of popular dendrochronology programs throughout history. More recent programs have implemented methods to locate possible errors and have created graphs to support manual error identification.

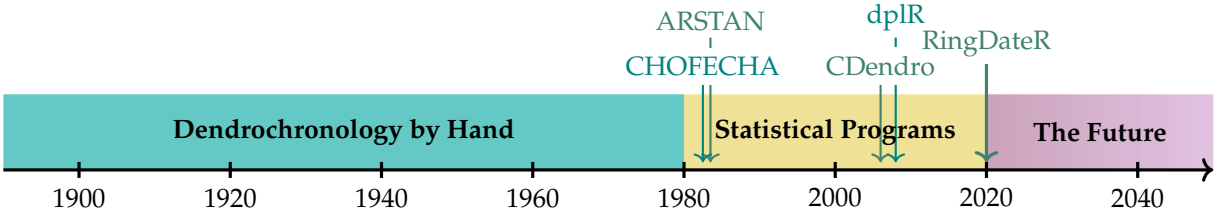

Figure 1.1: Timeline of Development of Popular Crossdating Programs.

Pairwise lead-lag analysis is a statistical method for the analysis of two sets of data over a time series; it is currently used in many of the dendrochronology programs available, such as CDendro [21]

and RingDateR [1] and has shown to be reliable and robust. The data is split up into segments of equal length; the sections of the master chronology will overlap for a given number of years. Figure 1.2 shows how a sample would be divided with maximal overlap.

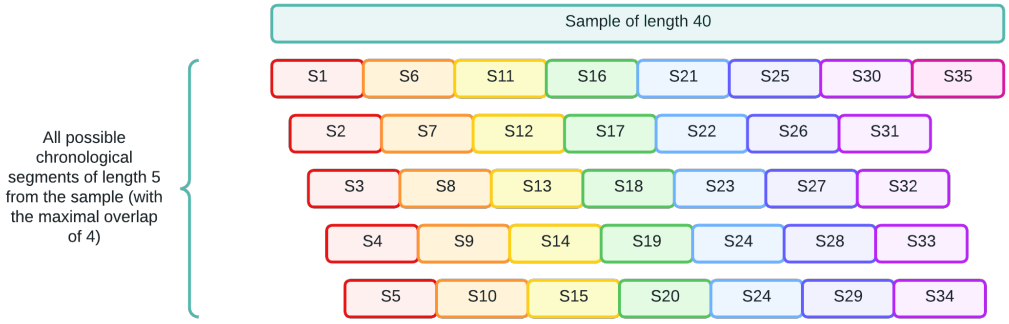

Figure 1.2: Example of a chronology of length 40 being split into length 5 segments with the maximum overlap of 4.

Each segment of the master chronology will be paired with the unknown chronology and then a t value will be calculated. The t values determine the strength of the correlation whilst taking into account the overlap in the data. This can be done by using this formula $\frac{r\sqrt{n-2}}{\sqrt{1-r^2}}$ where n is the number of years over which the correlation is calculated and r is the Pearson correlation coefficient [22]. The Pearson correlation coefficient is defined by $r = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i-\bar{y})^2}}$ where x_i is the i^{th} value of the x variable in a sample, $\bar{x}$ is the mean of the values of the x-variable, y_i is the i^{th} value of the y-variable in a sample and $\bar{y}$ is the mean of the values of the y-variable. In this example the x variable would be the master chronology segment and the y variable would be the sample chronology segment. Master and sample chronology pairs with a high absolute t value have a high correlation and are likely in the correct position relative to each other. So, in order to find the probable positions for both chronologies, all the t-values are compared to each other to find an outlier which is significantly larger than other t values [1]. Once an outlying t value is found, it is checked using cross-correlation and plotted as a line graph [23]. The lines for the master chronology and the new sample should follow similar trends.

It is important to note that there still exists variations in detrended samples and, therefore, matches may not be perfect, and any method must match the slopes of the lines formed by multiple data points rather than the values of each individual data point. Figure 1.3 is an example of a sample against a master chronology to illustrate this.

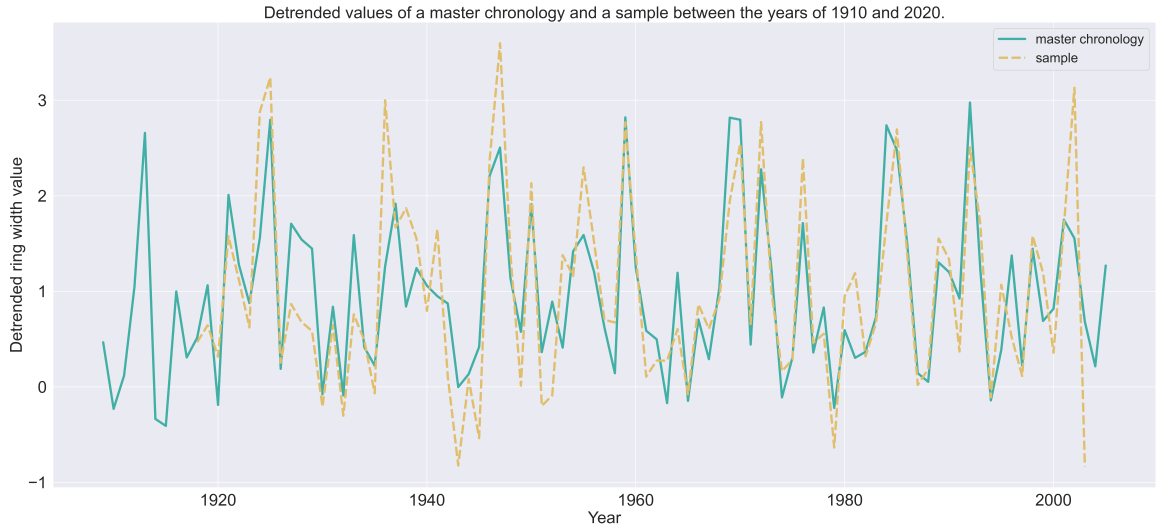

Figure 1.3: Line graph of a detrended master chronology (blue) against a sample chronology (yellow) between the years 1910 and 2020 from the 2013 Butler et al. marine molluscs dataset [24].

2 Project Specification

2.1 Project Specification

This project uses Python to experiment and explore both a statistical and machine learning approach to crossdating. Crossdating programs are predominantly written in R because it is a language tailored to statistical methods and graph visualisations. Implementing the tried and tested pairwise lead-lag analysis will allow the method to be experimented with to determine the best parameters to use as defaults and to suggest helpful insights to the user. Machine learning, in particular multi-layer perceptrons (MLP), has been used in various forms of pattern-matching experiments [25]. The second method will use MLP classifiers to crossdate, a classifier predicts a binary output of either 1 or 0 [26].

Errors such as ring insertion and deletion can severely affect the ability of statistical methods to crossdate, so new methods to improve the ability to handle errors in crossdating need to be explored. Thus, by designing two different approaches to the same problem and comparing them, their strengths and weaknesses can be properly assessed. Both methods are implemented and a user interface is constructed so the methods can be effectively run. The methods are tested with entirely correct master chronologies and samples with the possibility to extend to samples with errors (ring insertions or deletions).

2.2 Success Criteria

In order for this project to be successful both methods require robust methodologies with any design decision explained and testing to substantiate each claim. The success of each method will be compared using the success criteria in Chapter 4.

A successful method will:

1. Run in under 10 minutes on a standard laptop.

   A successful method should run in under 10 minutes on a standard laptop because it would be unpractical for users if it took any longer.

2. Work on multiple datasets.

   It must be robust and work correctly on multiple datasets so that it is effective as a crossdating program.

3. Return the correct results for each input.

4. Be easily explained.

   The explainability of the method is important because users will need to know how the method reached its solution.

5. Be easy to use with a simple user interface.

   A user interface allows the method to be used quickly with minimal difficulty.

6. Allow the user to customise the method.

   A customisable method increases the flexibility of the method allowing users to tailor the method to get the best results on any given sample.

7. Return extra helpful information in an easy-to-read format.

   Other state-of-the-art crossdating programs such as RingDateR's heatmap for t values, return helpful extra information which allows the users to gain more insights into their data and improve their confidence in the method [1].

3 Methodology

3.1 Dataset

This project used two datasets from the National Centers for Environmental Information Tree Ring database [27]. The first was a Dendrochronology dataset produced by Adams et al. and is a chronology containing 58 samples of tree rings from the Huachuca Mountains in North America formed between the years 1630 to 1995 [28]. The second was a Sclerochronology dataset produced by Butler et al and is a chronology containing 33 samples of marine molluscs from the North Icelandic Shelf formed between the years 649 to 2005 [24]. These datasets were chosen as they are from well-regarded researchers, were "absolutely dated" and the ring width values were already detrended. Various master chronologies were created from the datasets by taking different sets of 20 samples from each and averaging the years using Biweight Tukey Robust Mean, an industry-standard method that has proved more reliable than the arithmetic mean because it minimises the effect of outliers [29]. This was done using a method called `BIWEGHT` from the Real Statistics Functions package in Excel [30].

3.2 Statistical Method

The basis of the method is pairwise lead-lag analysis detailed in Chapter 1. In order to give dendrochronologists flexibility to optimise the technique for different species, experiments have been conducted on possible parameters and user input is required with a suggested default. The method splits up the master chronology and sample chronology into as many equal-length segments as possible, it creates a list of every combination of the sample segments to each of the master chronology segments. The t values are calculated for every element in the list and elements are identified as outliers if the t value is more than an inputted number of standard deviations away from the mean. The method then deviates from the traditional method of choosing the highest t values and calculating the start year. A filter is applied to remove pairs which do not have an outlying pair directly afterwards for a specified number of consecutive segments and then, a list of these pairs is returned. The filter is applied to reduce the number of outliers for the next step. For each of these pairs, a start year for where the segments line up is calculated and the occurrence of each start year is counted, the highest number of counts is given as an output.

3.2.1 Implementation

The first version of this method simply implemented pairwise lead-lag analysis, then picked the highest t value and found the position the sample chronology is in compared to the master chronology. The statistical crossdating process can be split up into 7 steps:

1. Add the master chronology and the year index to a data frame.
2. Divide the sample and the master chronology into equal-length segments defined by the user so that segments have the maximum possible overlap.
3. Create pairs of every sample segment with every master chronology segment.
4. Calculate the t values for all of the pairs.
5. Find the highest t value.
6. Calculate the start year of the sample from the index of the master chronology segment and sample chronology segment from the highest t value.

7. Return the start year for the largest t value and add the sample chronology to the data frame in the correct place.

To begin the program asks for a CSV containing 3 columns, the years that index the master chronology, the master chronology and a final column containing the sample chronology to crossdate with the master chronology with padding at the end so all columns are the same length. The CSV file is converted into a `pandas DataFrame` [31]. This is because it helps to create easy `matplotlib` graphs [32] and the final crossdated position can be added back into the data frame with the correct padding to be converted into a CSV file. Next, the columns in the data frame were stripped of their padding and stored in 2 lists. Using list comprehension the lists are split into segments of length given by the user. The segments of shorter length are removed and all the values are added to a dictionary. Each of the master chronology segments and the sample chronology segments is then paired together using list comprehensions so that every sample segment is compared to the first segment of the master chronology and so forth. The t values for each of the pairs are calculated using Equation 1.1 and stored as the key to a dictionary where the value is a list containing both of the segments. The highest absolute t value is located and using the dictionary the master chronology and sample chronology segments are identified. The start year of the sample segment can be calculated using the location of the first element of each of the segments in relation to their respective chronologies.

The method failed for 2 of the 33 samples it was tested on because the highest absolute t value was an anomaly where that pair of segments had a high correlation but the rest of the chronology had a low correlation at that position. This is a known challenge of pairwise-lead lag analysis called multiplicity and in order to make the method more accurate it had to be adapted to reduce the impact of this error [2].

The second version of this method was adjusted after the t values were calculated. It found all outlying t values that were a set number of standard deviations away from the mean and calculated the start years of each of the pairs. The start years were then counted up using the `Counter` method from the `collections` package in Python and the start year with the highest count was returned as the correct start year. This method was much more successful and returned the correct result for samples from the Dendrochronology and the Sclerochronology dataset. However, even with an absolutely dated master chronology and samples with no errors, there were still incorrect start years with multiple counts, as such the introduction of more noise or errors would cause the method to return incorrect results. Additionally, even if the segment did not match the master chronology it was still returning an incorrect start year with multiple counts which is misleading. This led to the development of a filter function for the outlying t values so the margin between the correct start year count and the incorrect start years was larger.

The third and final version of the method used a filter function after the t values were calculated so that the number of incorrect pairs going into the counter was reduced. This was done by removing outliers where the next consecutive segment in the master and sample chronology was not an outlier. For example, if the filter was set to 2 consecutive outliers then a sample chronology segment from index 10-19 inclusive and the master chronology segment from index 40-49 inclusive would only be included if a sample chronology segment from index 11-20 and the master chronology segment from index 41-50 was also an outlier. This filter attempts to automate rejecting anomalies and is proved successful on testing data from both datasets and identified that a sample chronology did not match with a master chronology. Throughout the development of the method, several experiments were performed to find and optimise the parameters of the model. From this, the user has the option to adjust three different parameters in order to improve the performance of the method. There is a default given for each of the parameters based on the results of the experiments. The parameters users can adjust are as follows:

1. The length of the segments the master chronology and the sample chronology are separated into.
2. The number of standard deviations away from the mean required for a t value to be an outlier.

3. The number of consecutive outliers required to be considered significant (strength of the filter function).

Through testing, it can be seen in Figure 3.1 that a segment of length 10 is the point at which the method begins to return correct results.

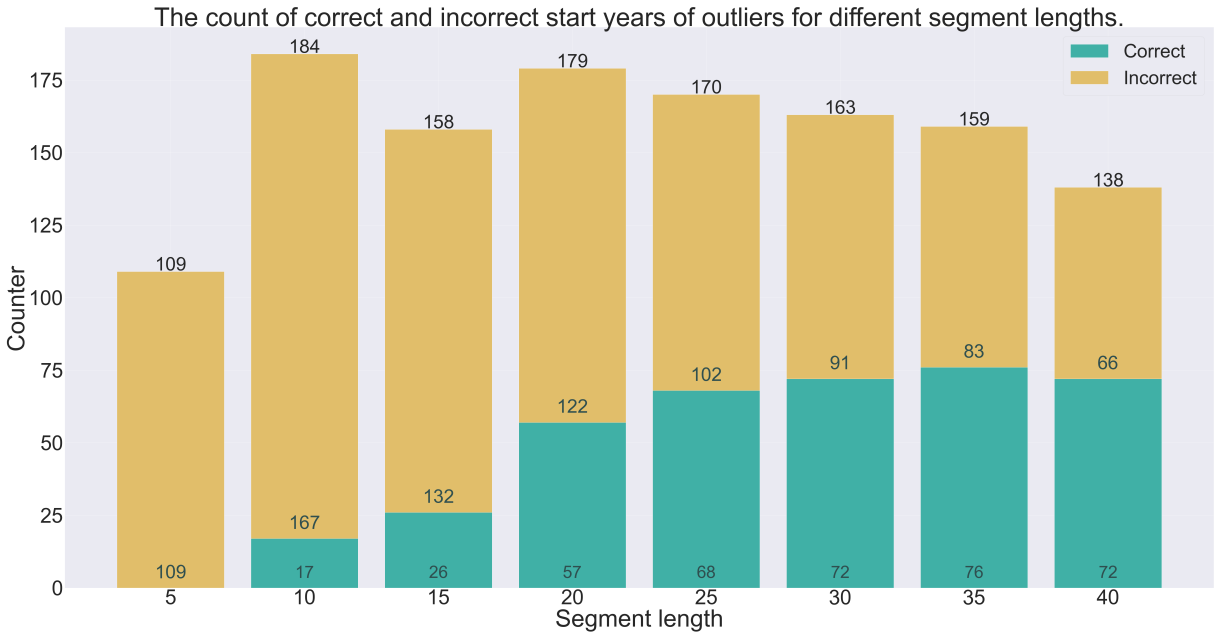

Figure 3.1: Graph depicting the proportion for which the outlying values (over 3 standard deviations from the mean) predict the correct start year for the chronology with no filter.

Figure 3.2 uses the same sample and master chronology with a filter of 8. Additional experiments with different strength filters are located in the Appendix 8.

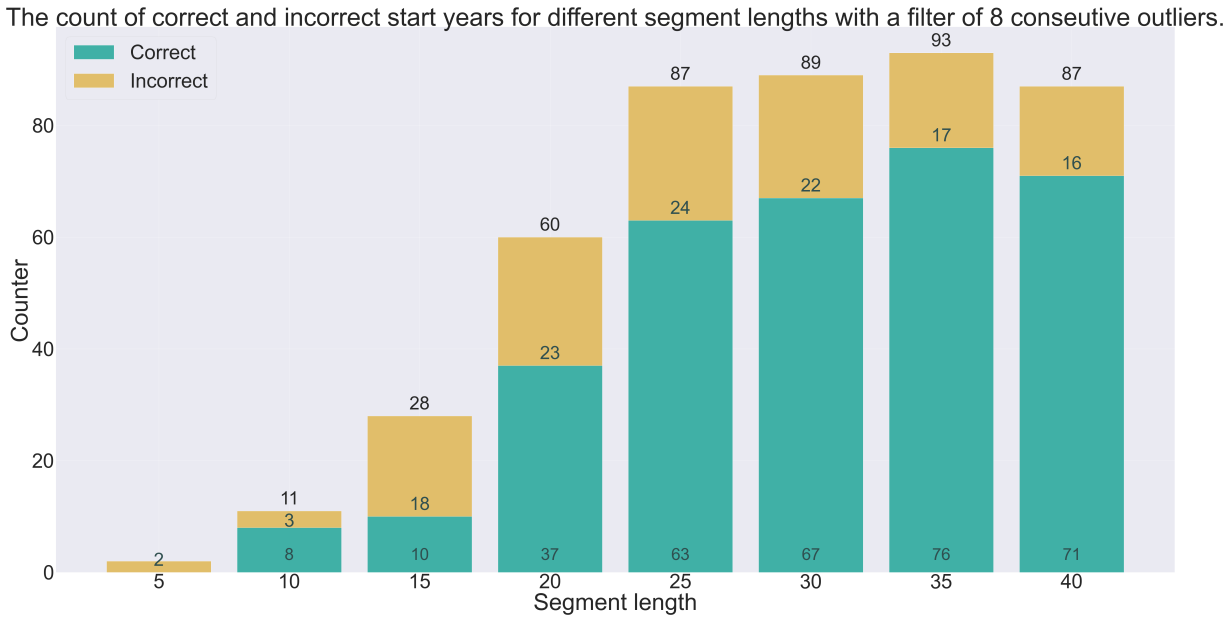

Figure 3.2: Graph depicting the proportion for which the outlying values (over 3 standard deviations from the mean) predict the correct start year for the chronology with a filter of length 8 for different segment lengths.

Figure 3.1 and Figure 3.2 show that the strength of the filter has a large impact on the proportion of

correct start years produced from different segment lengths. Due to this, it is impossible to consider the parameters independently of each other and instead, they must be considered holistically as a multi-variable optimisation problem. By conducting these experiments it is possible to suggest a default set of parameters which produce correct results for a variety of datasets.

Figure 3.2 shows that a segment of length 10 has the highest proportion of correct start years to incorrect start years and the correct start year is obtained for other filter values so it will be suggested as the default for this parameter. This is concurrent with observations of additional experiments performed, the graphs produced can be found in the Appendix 8.

It is worth noting that the incorrect answers encapsulate all incorrect start years, so even if the proportion looks high it is because there are many different incorrect start years that only have a low or single count. Table 3.1 illustrates that despite the high number of incorrect answers (yellow), the correct answer (blue) still has the highest individual count of any of the start years and thus would be the output of the method. Whilst this produced the correct start year for samples with little noise or errors, it cannot be guaranteed that the correct start year will have the highest count if this is not the case which is why the filter was added to the method.

Segment length (years)	First most common Start year	First most common Start year count	Second most common Start year count	Second most common Start year count	Third most common Start year	Third most common Start year count
5	1759	4	1766	4	1827	4
10	1886	17	1799	13	1581	9
15	1886	26	1581	14	1668	11
20	1886	57	1581	20	1627	13
25	1886	68	1581	20	1627	14
30	1886	71	1581	21	1627	11
35	1886	73	1581	18	1627	12
40	1886	68	1581	17	1627	13

Table 3.1: The start years and their corresponding counts for the first 3 most common values for different segment lengths, the correct start year and count cells are coloured blue and the incorrect start years and counts are coloured yellow for dataset 4 with no filter as shown in Figure 3.1.

Next, experiments were performed to find the number of standard deviations away from the mean required to categorise the t value as an outlier. The best standard deviation is heavily related to the size of the segments and so for these experiments, the suggested default of segment length 10 was used. From Figure 3.3 a standard deviation of 3 has the highest proportion of correct start years to incorrect start years. This is the same result as other researchers experimenting with the standard deviation in computer-based pairwise lead-lag analysis programs [33]. As the filter becomes stronger the number of standard deviations required to obtain the high proportion of correct start years decreases. For the filter of 8, there are no outliers that fit the criteria beyond 3 standard deviations and the last count returns the correct start year as the highest result. However, with a filter of 3 or below a standard deviation above 3 for which there were still outliers returned the incorrect start year such as the case for a filter of 3 for 4 and 5 standard deviations. From this, it can be said that 3 standard deviations is a good default and increasing it further can decrease performance.

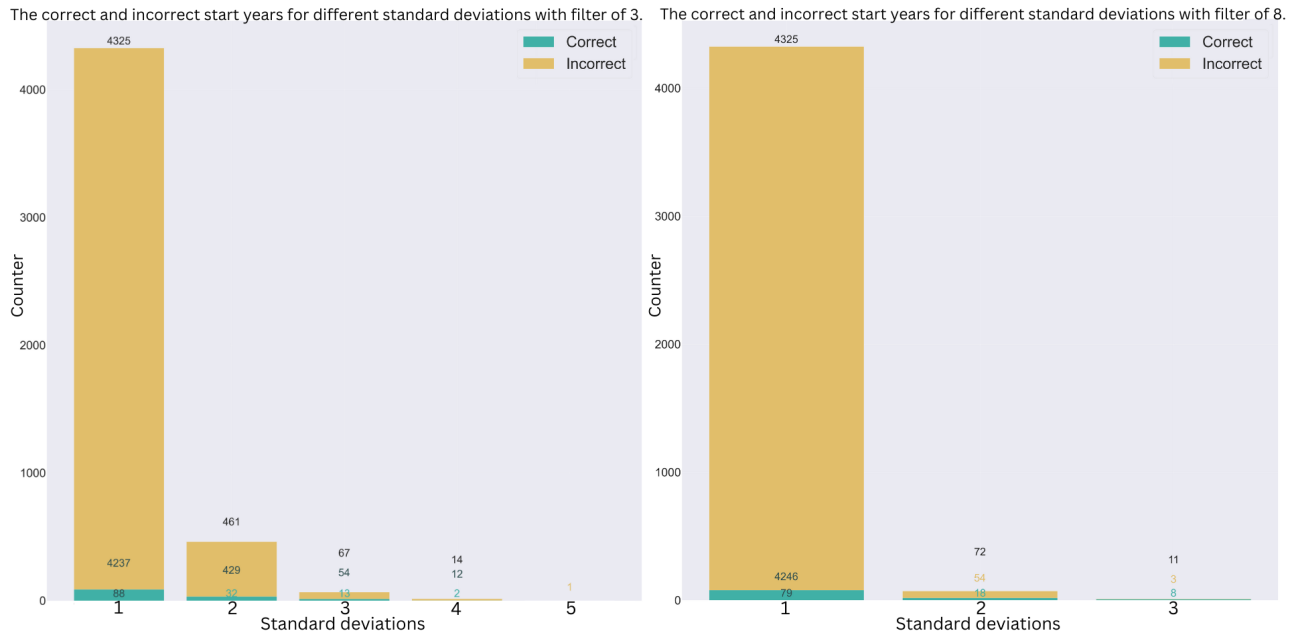

Figure 3.3: Graphs depicting the proportion for which outlying values at different standard deviations from the mean predict the correct start year for the chronology with a filter of 3 or 8 consecutive outliers.

The third parameter is the number of consecutive outliers required to be considered significant. As this parameter increases the number of significant outliers decreases, which allows the counting method to work more effectively. A filter of 0 would output the same result as if the filter was not applied. For the experiments, a segment of length 10 and a standard deviation of 3 were used as they have been chosen as the default values and performed well in every test previously conducted. Figure 3.4 shows the effect of varying the strength of the filter on the count of correct and incorrect start years. The same experiment was performed on several different master chronology and sample pairs which reinforce these findings, the results can be seen in the Appendix (8).

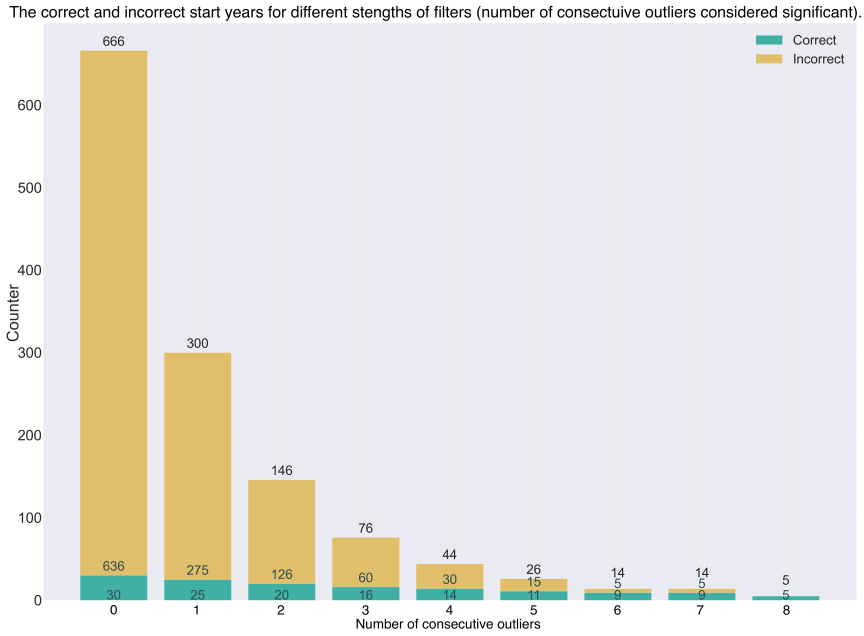

Figure 3.4: Graph depicting the proportion of correct start year (blue) to incorrect start years (yellow) for the t values of segments for length 10 where outliers are 3 standard deviations away from the mean.

The filter has a big impact on reducing the number of incorrect start year counts whilst maintaining most of the correct start year counts. For Figure 3.4 the ratio of correct start years to incorrect start years is 5 : 106 whereas the ratio for the filter of 8 is 5 : 0, where only the correct start year remained after the filtering. This is not always the case, sometimes there are incorrect start years at the end of the filter but their number has significantly reduced and there is always less in total than the correct start year class. The loss of some of the correct start year counts is due to the fact there were outliers near the end of the sample and so it was impossible for there to be enough consecutive outliers to pass the filter, and it is small in comparison to the number of incorrect samples removed by the filter. Another benefit of using the filter is that if there is no match between the master and the sample chronology the filter will reduce the significant outliers to 0 so that it does not return the incorrect result as shown in Figure 3.5.

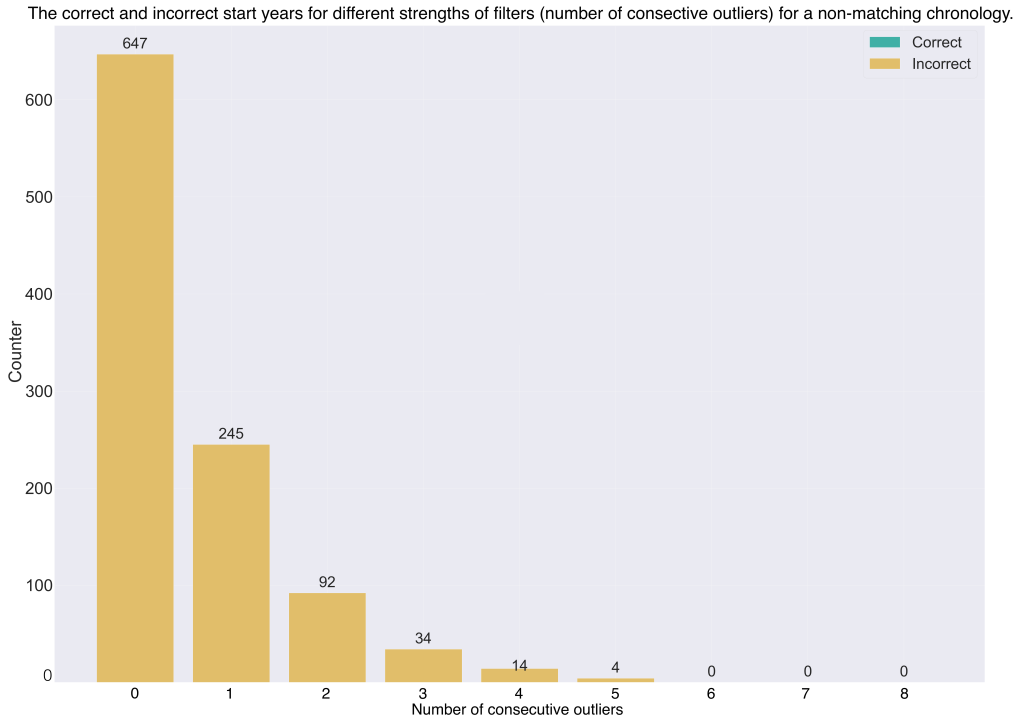

Figure 3.5: Graph depicting the number of incorrect start years (yellow) for the t values of segments for length 10 where outliers are 3 standard deviations from the mean.

Overall, the filter is very successful at making the counting method viable, for data with more noise or possible errors the filter can be reduced and the counter can provide helpful information including narrowing down all the likely start years which can be checked manually. It must be adjustable because short samples are unlikely to have outliers for 8 consecutive segments due to the variability between signals described in the introduction.

Additionally, the user has the ability to return multiple possible likely start years. The counter ranks the start years and returns the highest of however many the user chooses. This is so the user can see different possible options and the method offers flexibility for datasets with noise or possible errors.

3.3 Machine Learning Method

The Machine Learning Method used Binary MLP classifiers from the `sklearn` package [34] to determine if segments of the master and sample chronology overlapped or matched. Training data, made up of 17 different master chronologies and sample pairs, was created using data from the Dendrochronology dataset. The models were then tested first using the data from splitting the dataset

between training and testing and then using a separate master chronology and sample chronology so the method could be tested all the way through. After the method had been developed the testing was expanded to include the Sclerochronology dataset.

3.3.1 Implementation

The first method divided the master chronology and sample chronology into segments of length 10 and made a list of every possible master chronology segment paired with every possible sample segment. The segment pairs that were an exact match were classified as 1 and the segment pairs that did not align with each other were classified as 0. The target of the model was to predict where the master chronology and sample chronology align and to use the segment pairs predicted as 1 and the counting method for all the given start years to determine the most likely one. An MLP binary classifier was used because of its ability to accurately pattern match in other fields [35]. The dataset was split up into 60% training and 40% testing. The parameters of the MLP such as hidden layer size, activation, solver, alpha, learning rate, max iterations and early stopping were tested using `GridSearchCV`. The method tests all possible combinations of an inputted list of comparisons and returns the combination with the highest accuracy, the results are then printed as a report found in the Appendix (8). The `GridSearchCV` method also allowed the parameters to be cross-validated with the industry standard of 10 k-folds [36]. All the results are stored in a data frame with the master chronology and sample chronology segments on one row and then the result of the classifier on the other. The pairs which returned a 1 are extracted from the data frame and the master chronology segment is split from the sample segment. The same counting method as the statistical method is employed where all the start year from all the extracted pairs is calculated and then the occurrence of each is counted and ordered using the `Counter` method. The results of this method were poor, to begin with, the model only predicted 0. This was due to the high class imbalance between 1's and 0's as only 0.07% of the training data was class 1. Class imbalance is a weakness of MLP's and classifiers in general, there are a few methods to reduce the effect on the output of the model [37]. One method to combat this is undersampling, where the number of 0 class inputs is reduced to the size of the 1 class. The best parameters after testing were:

```
Best parameters found:
{'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes':
(150,), 'learning_rate': 'constant', 'max_iter': 100000, 'solver': 'adam'}
```

The metrics for each class of this model are shown in Figure 3.6.

```
Results on the test set:
              precision    recall  f1-score   support

           0       1.00      0.89      0.94    189528
           1       0.01      0.93      0.01       136

    accuracy                           0.89    189664
   macro avg       0.50      0.91      0.48    189664
weighted avg       1.00      0.89      0.94    189664
```

Figure 3.6: Accuracy metrics for both classes for the best model returned by the GridSearchCV method.

Figure 3.6 shows the class 1 output had a precision of 0.01. This means that the proportion of true positives to predicted positives was low. So, while undersampling did increase the accuracy of the model then produced a large proportion of false positives shown in Figure 3.7 which meant the counting method never returned the correct results.

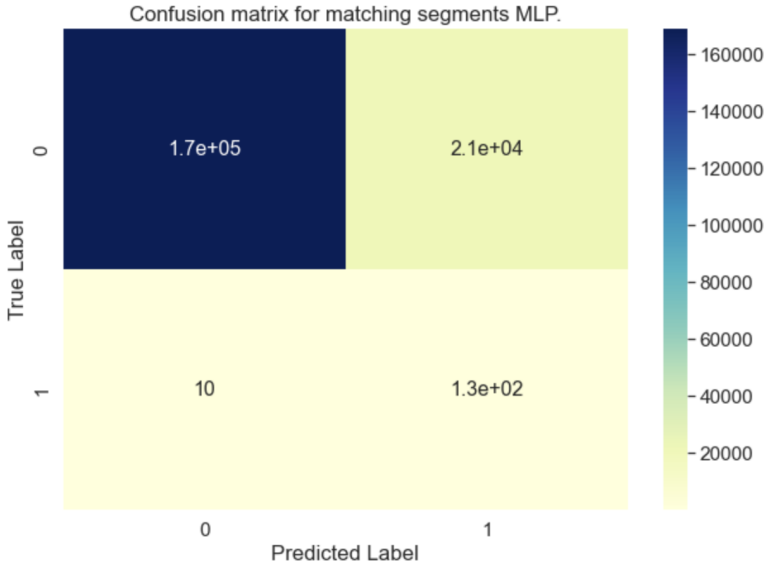

Figure 3.7: Confusion matrix for the test class of the first version of the machine learning method.

In order to reduce the class imbalance another MLP classifier was added to the model which fed into the model previously described. This model took the same input of 10 length segments of the master and sample chronology but instead predicted whether the master chronology and the sample chronology overlapped by at least 5 years. The `GridSearchCV` was implemented again using a variety of different parameters, the results can be found in the Appendix (8). While this decreased the class imbalance, it was still very skewed because a typical sample chronology is between 50 and 200 years long whereas the master chronologies are thousands of years long. Due to this undersampling was still required on the new MLP, and this resulted in an even higher proportion of false positives.

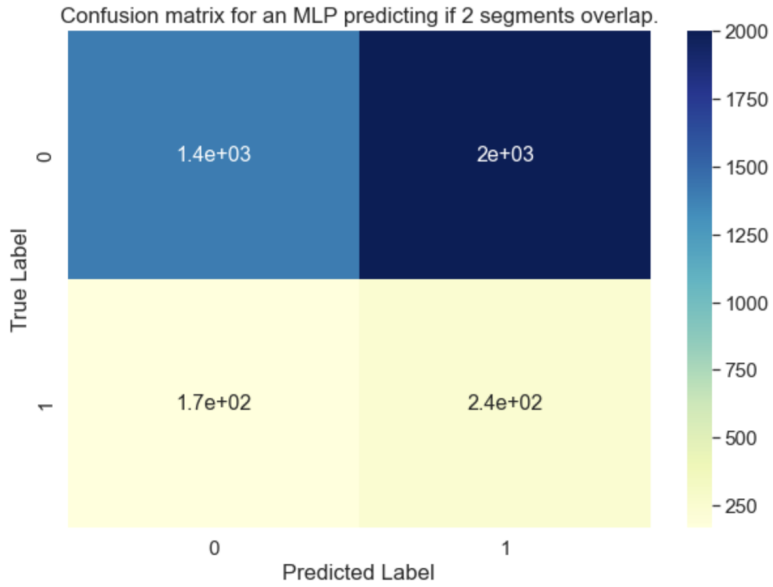

Figure 3.8: Confusion matrix for the test class of the first MLP for the second version of the machine learning method.

This affected the ability of the second MLP because the false positives diluted the count used to determine the correct start year and the false negatives reduced the proportion of class 1 values in the next model. The next MLP often could not predict any true positives as none existed within the data filtered through by the first MLP. This version of the method also failed to crossdate every sample

tested on it.

The final version of this method added a third MLP in front of the previous two. The reason both of the previous methods failed was due to class imbalance because the master chronology is 2 magnitudes larger than the sample chronology. In order to reduce the size of the class 0 inputs in the first method, the master chronology needed to be reduced to just larger than the sample chronology. To do this the third binary classifier classified whether a segment of length 10 matched within a segment of length 100 of the master chronology. This meant a typical master chronology pair reduced the master chronology from several thousand years to between 100-300 where the sample chronology is most likely to start. The best parameters of this model after testing using a variety of combinations, which can be found in the Appendix (8), are given below.

```
Best parameters found:
{'activation': 'relu', 'alpha': 0.001, 'early_stopping': False, 'hidden_layer_sizes':
(300, 100), 'learning_rate': 'constant', 'max_iter': 7000, 'solver': 'adam'}
```

This method had an accuracy of 0.819 and consistently outputted the a high number of true positive results without undersampling, successfully reducing the segment of the master chronology being considered. Figure 3.9 gives the individual metrics for each class.

```
Results on the test set:
              precision    recall  f1-score   support

           0       0.87      0.90      0.89      4472
           1       0.66      0.60      0.63      1456

    accuracy                           0.83      5928
   macro avg       0.77      0.75      0.76      5928
weighted avg       0.82      0.83      0.82      5928
```

Figure 3.9: Accuracy metrics for both classes for the best model for the first MLP in the third version of the method.

Each class has a high precision score unlike the previous versions, this means it is a good first model and will allow the next two models to have higher accuracy. Figure 3.10 shows the confusion matrix for the first MLP in the third version of the method.

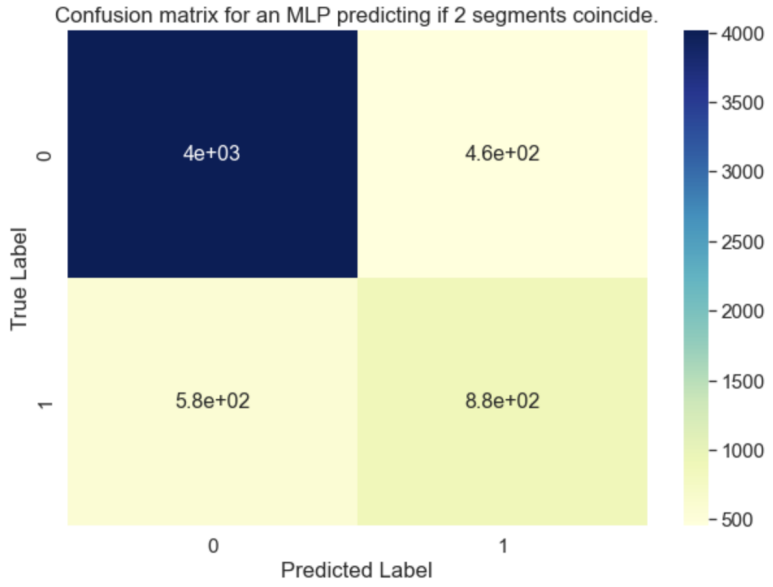

Figure 3.10: Confusion matrix for the test class of the first MLP for the third version of the machine learning method.

The class results were then taken and the segments of the master chronology were split into as many segments of length 10 as possible. They were then paired with the sample segment pairs and used as the input for the second MLP which determines if the master and sample segment have an overlap of at least 5 years (both segments had at least 5 years where they were both alive). The first MLP worked well to filter the master chronology and as such the performance of the second method increased. Undersampling was still required to get the best results but the precision was higher than in previous versions using this MLP. The best parameters for this MLP

```
Best parameters found:
{'activation': 'relu', 'alpha': 0.001, 'early_stopping': False, 'hidden_layer_sizes':
(300, 100), 'learning_rate': 'constant', 'max_iter': 7000, 'solver': 'adam'}
```

Figure 3.11 shows the individual metrics for each class for the second MLP. The accuracy of this model was 0.779 (+/-0.016) which is a good accuracy with a low standard deviation in comparison to the first use of the model in version 2 where the accuracy was 0.750 (+/-0.522) which is lower and significantly less consistent.

```
Results on the test set:
              precision    recall  f1-score   support

           0       1.00      0.78      0.87   1665862
           1       0.04      0.78      0.08     19913

    accuracy                           0.78   1685775
   macro avg       0.52      0.78      0.48   1685775
weighted avg       0.99      0.78      0.87   1685775
```

Figure 3.11: Accuracy metrics for both classes for the best model for the second MLP in the third version of the method.

It is important to note that despite the precision not being much higher in the first method the recall is high for both classes as there were fewer false negatives which is good as the method is designed to filter out negatives whilst retaining the positives. This point is illustrated by Figure 3.12.

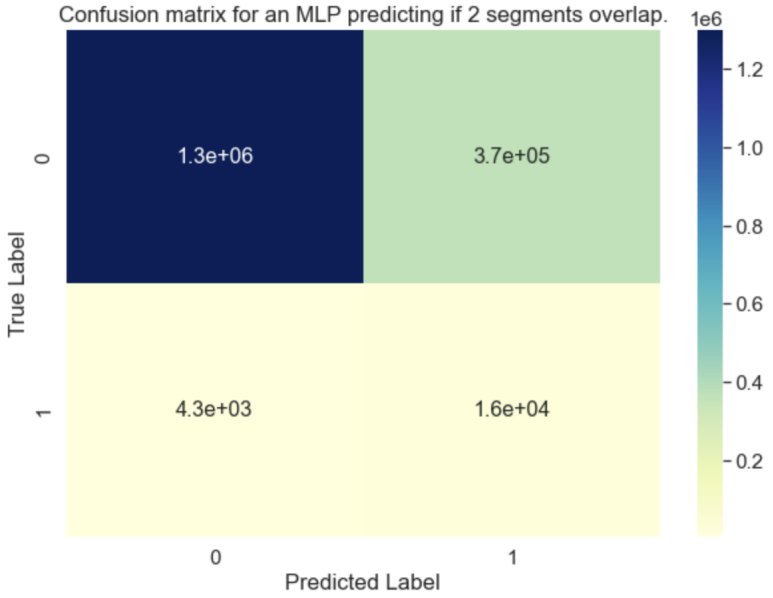

Figure 3.12: Confusion matrix for the test class of the second MLP for the third version of the machine learning method.

The class 1 output of the second MLP model was used as input for the final MLP to find an exact match between master and sample chronology segments. It required undersampling as there was still a big class imbalance. The best parameters of the model after testing multiple parameters are given below.

```
Best parameters found:
{'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes
':(525,), 'learning_rate': 'invscaling', 'max_iter': 600, 'solver': 'adam'}
```

The metrics for both classes of the model are shown in Figure 3.13

```
Results on the test set:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    190887
           1       0.19      0.13      0.15       727

    accuracy                           0.99    191614
   macro avg       0.59      0.56      0.57    191614
weighted avg       0.99      0.99      0.99    191614
```

Figure 3.13: Accuracy metrics for both classes for the best model for the third MLP in the third version of the method.

This was a considerable improvement from the first and second versions allowing the method to correctly predict the start year for multiple different sets of test data. Despite the false positives, the number of correct start years was significantly larger than any single incorrect start years count. Figure 3.14 is a pie chart of the start years for one of the test datasets, the correct start year is 1812.

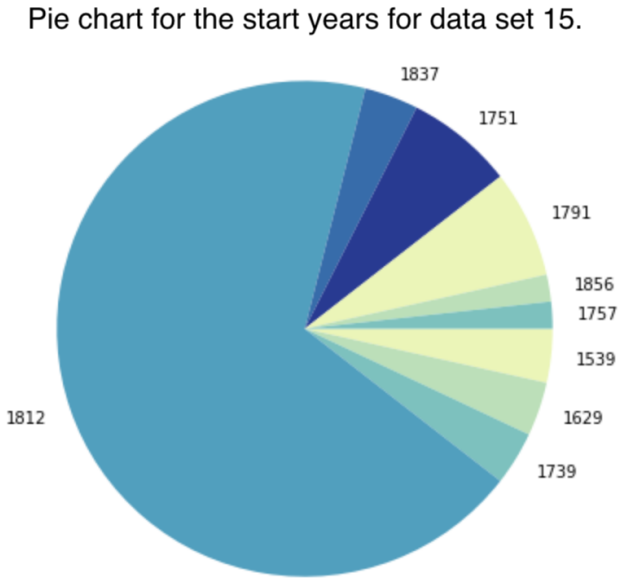

Figure 3.14: Pie chart for the start years for testing dataset 15 outputted by the third version of the machine learning method.

3.4 Graphical User Interface (GUI)

A GUI was created using the package `pysimpleGUI` in order to make it easier to use the methods and view graphical outputs. `PysimpleGUI` was chosen as it can integrate `matplotlib` graphs, take

user input and is easy to implement. Additionally, a GUI makes the application more accessible by introducing a dyslexia-friendly theme, larger text and options to enlarge graphical outputs. Features such as File browsers, progress bars and interactive graphs were integrated into the program to improve the user experience. All of the pages are resizable to allow for different screens and monitors. There is only one disadvantage of this package, it has a known bug where the navigation bar for a `matplotlib` graph can be embedded in a window and is functional but does not display correctly on macOS. It displays correctly on Windows and Linux machines. Figure 3.15 is the flowchart between all of the pages found in the application. Screenshots of all of the pages can be found in the Appendix Subsection 8.6.

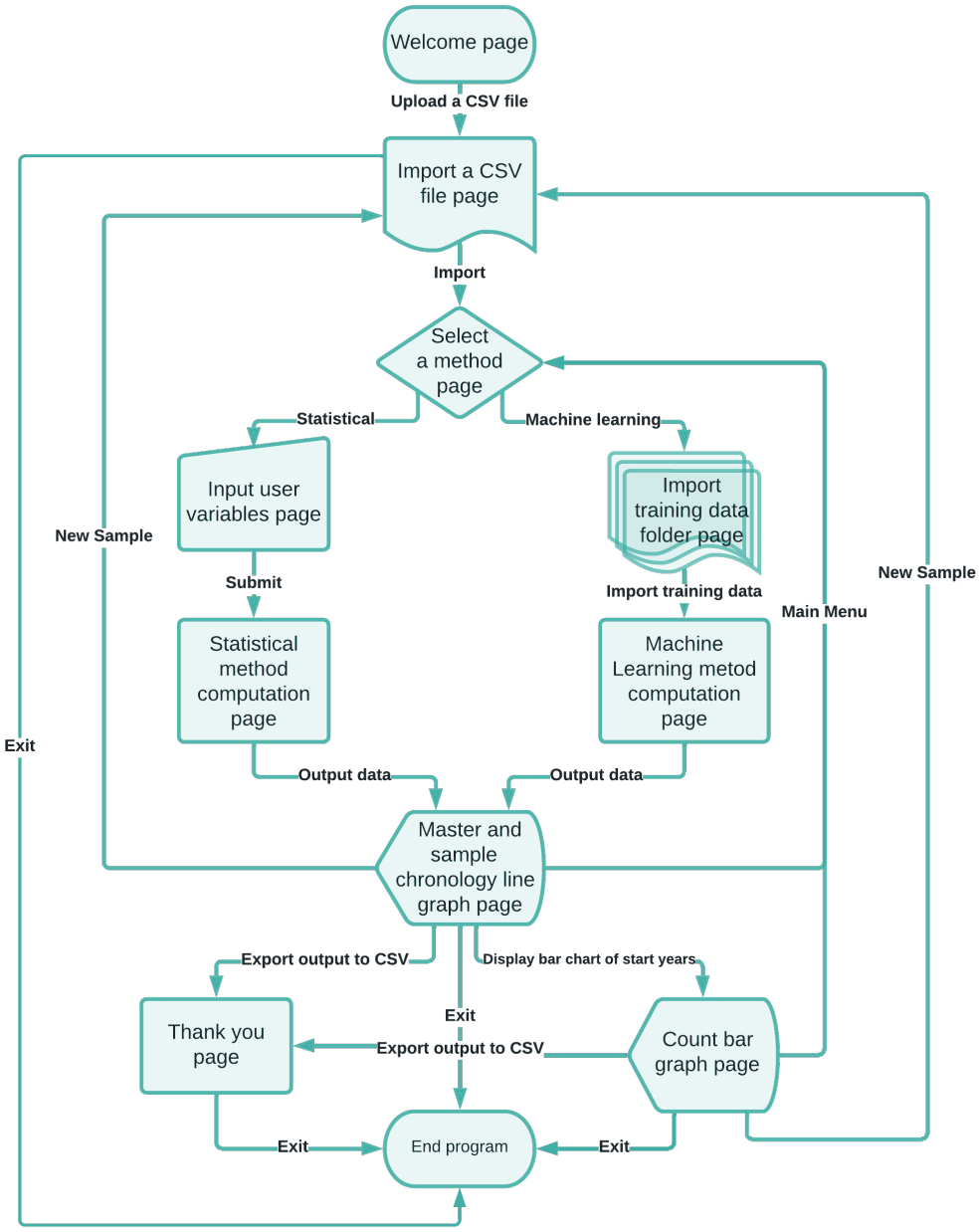

Figure 3.15: Flowchart of the pages in the Crossdating Application where all the boxes are pages and the arrows are buttons available to press on each of the pages.

4 Results

In order to conclusively determine the best method both were analysed and evaluated in this section to identify strengths and weaknesses.

4.1 Analysis

The methods will be compared using the 7 characteristics defined in the project specification to determine their overall success, they will be compared to each other and, where possible, a current accepted industry standard package called RingDateR to determine if improvements have been made [1]. The speed of the methods was compared using the same test data on the same computer. To compare the speeds of the methods a built in python package called `cProfile` [38] was used as it returns the time taken to run a method and breaks down the time taken for all the steps of the method. The statistical method returned the following overview (the full return statement can be found in the Appendix 8.24):

```
7131937 function calls (6382152 primitive calls) in 8.643 seconds
```

In comparison, the online RingDateR package version 1.4 applied to the same data took 1.68 seconds to complete, so, despite the results being slower than an industry-standard package they are still within the same magnitude [39]. The Machine learning method was much slower taking 11.20 minutes as shown in the following overview (the full return statement can be found in the Appendix 8.25):

```
22676196 function calls (22559776 primitive calls) in 672.294 seconds
```

This is significantly slower than the statistical method and took almost 700 times as long as RingDateR. One deciding factor on the runtime is the amount of training data used to train the model, in this case, 16 datasets were used to train the model. The model returned the correct start year given this training date with these parameters, reducing the training dataset will reduce the accuracy of the model and the ability of the model to correctly crossdate the test data.

The statistical method produced the correct results for the 33 samples it was tested on 16 samples from the Sclerochronology dataset and 17 samples from the Dendrochronology dataset. Out of these datasets, 31 produced the correct results with the default values given. The other two (Sclerochronology samples 8 and 9) required the strength of the filter to be reduced to 6 consecutive segments as the samples in the datasets were both less than 80 years in length so outlying samples of 8 sets of consecutive segments of length 10 is not possible. This shows that the statistical method is robust and produces correct results from multiple datasets. The machine learning method produced the correct results for 31 of the 33 datasets it was tested on. It failed to identify the correct start year, or even have the correct start year appear in the top 20 start year counts for Sclerochronology samples 15 and 16. Both of these samples were between 200 and 300 years in length and had start years and end years in the middle of the chronology. The output for each of the samples was around 200 years later than the real start date. This may suggest the method can only identify samples' start years if the sample ends within the last 100 years of the end of the master chronology. However, sample 10 in the Dendrochronology dataset has the same characteristics but it was crossdated correctly by the method so this may be a more complicated issue. Producing incorrect results means the method is not viable for crossdating. It is dependent on the training data added into it and because most of the samples end within a few years of the master chronology ending it is possibly overfitting. This is where it too closely models the training data so cannot accurately perform on unseen data, defeating the point of the model [40]. An overview of these results can be seen in Table 4.1.

	Dataset	Sclerochronology															
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Method	Statistical	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Machine Learning	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	x	x

	Dataset	Dendrochronology															
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Method	Statistical	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Machine Learning	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Table 4.1: Table showing if the outcome of the model was correct for each particular sample for both datasets. A blue box with a tick signifies the method produced the correct start year and a yellow box with a cross signifies the method produced the incorrect start year for the sample.

The statistical method can be easily explained to a Dendrochronologist as the majority of the process is the industry-standard method of pairwise lead-lag analysis. Each design decision is explained and the functions that make up the method are documented in the code. This is important as users of the package need to understand the progress in order to make useful adjustments using the variables in order to get the most robust results. Understanding the decisions made by a machine learning model can be difficult as they are often beyond human comprehension, this is known as the black box problem [41]. This is particularly a problem because the method is returning incorrect results for certain samples but other samples with similar characteristics are returning correct results so the reason for these results cannot be determined and mitigated easily. It is difficult to convince potential users to use the method if the reason why the method works cannot be explained.

The GUI allows users to try both methods in the same method. Once running it is simple to use with clear instructions on each page. Unlike RingDateR which can be downloaded or used as a website, the app must be run locally, requiring the user to run the main.py file. The program requires a short set-up progress, python must be downloaded and 4 additional dependencies, listed in the ReadMe, must be installed using the command line. Additionally, the input data requires a specific format of a CSV file with 3 columns, the first a list of years which indexes the master chronology, the second a master chronology and the third a sample chronology with appropriate padding. This is a reasonable format as CSV files are a standard file format for data. The statistical method can sometimes take experimentation with the variables in order to obtain correct answers, the filter strength in particular can affect whether the correct result is returned for short samples. One method to determine if the correct result is returned is to check the disparity between the first and second most common start years. For incorrect results, the gap is usually minimal whereas for correct results it is often double the next most common start date. The machine learning method requires several dated samples and master chronologies in order to work as such it cannot be used without other data first being crossdated correctly. The statistical method can work on individual samples without the need for extra data so is more flexible in this regard. The machine learning method's count graph is not as helpful as the statistical method as it cannot be explained why the model has returned those start years.

The ability to customise multiple variables of the statistical method is a particular strength of the method. It allows the program to apply to multiple different datasets producing accurate results. This is similar to RingDateR which allows the user to customise 6 different variables including the method for detrending data. The machine learning method allows the user to customise the training dataset inputted in the method. This is good if you have a set of samples that have already been dated from the same area and there are additional samples to crossdate. However, if the user does not have crossdated samples readily available they would have to gather similar samples from the National Centers for Environmental Information Tree Ring database and put them in the right format which would be time-consuming [27].

Finally, both methods return the same information, including a line graph of master and sample chronologies at the most likely start year and a bar graph of the up to the 15 highest likely start years with their respective counts. Figure 4.1 shows the Bar graph page produced by both of the methods as an example. Both graphs can be adjusted, zoomed in and saved using the toolbar embedded into the

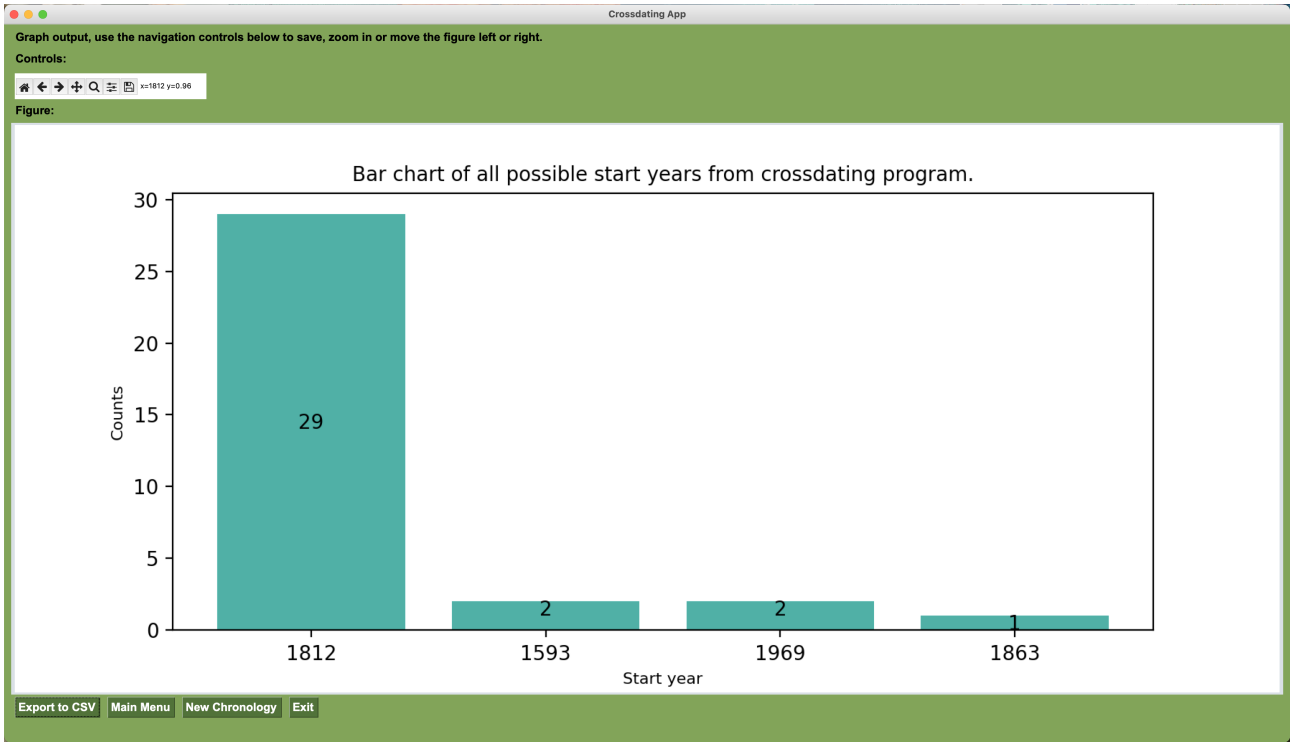

Figure 4.1: Bar chart of the possible start dates with their respective counts.

window. Additionally, the results are exported to a CSV with the original 3 columns from the input CSV plus the sample aligned in the correct position. For the statistical method, multiple possible start years can be exported by adjusting the number of start years the program outputs variable above the default of 1. The bar charts produced by both methods identify possible start dates, this can be helpful to check how likely the highest start year is the correct start year and possibly identify alternative positions to check, narrowing down the alignment of the chronology. This is an attempt to emulate the innovative heatmap used in RingdDateR which identifies positions with high correlations and can show where there are possible errors. It is useful for the methods to utilise the computations they have performed to generate helpful graphs which can allow users to gain more insights.

4.2 Evaluation

The Statistical method performed better overall, its key benefits were that it was explainable and easily adjustable, performed well on different datasets and, by using the filter samples that did not coincide with the master chronology returned a null result. It was successful on all the success criteria objectives and augmented the original pairwise lead-lag analysis method to produce results that would not be affected by anomalously high t values. The variable analysis allowed test lead defaults to be implemented for each of the variables with the option to edit them.

The machine learning method has two major flaws; it took 11 minutes to run each sample and it returned incorrect results for two of the test data samples. Therefore, it can be concluded that the machine learning method is not reliable or robust and so was less successful.

5 Discussion

5.1 Critical Reflection

Despite successes in the project, there were areas that could be improved upon. The machine learning method was fundamentally flawed due to how the binary classifiers were used. The final method led to a large class imbalance in the second and third MLP. This meant that the training data required heavy undersampling which significantly reduced the size of the training data and caused a high proportion of false positives. Additionally, since a larger dataset was required to offset the reduction in training data, the time the method took to run was increased. The false positives produced from the final MLP often outnumbered the number of true positives predicted, it was only by implementing the counting function developed for the statistical method that the correct result was returned for 31 of the 33 test data samples. Research is being conducted into reducing the effect of class imbalance in machine learning models so the method could be improved if a breakthrough is made in that area [42]. Alternatively, a different approach using a neural network or an MLP regressor may have been more effective because it has had success in other fields [43][44].

The methods were only tested on a simplified version of average data where an accurate master chronology has already been created. This reduces the difficulty of crossdating new samples because the master chronology has a strong signal to compare with. The introduction of errors in the data would solidify if the method can effectively crossdate all datasets.

5.2 Future Directions

The project forms a strong basis to continue developing a new crossdating method. The statistical method showed promise and a similar method has not been published within the field. A number of potential improvements to make have been identified. The first is implementing a method that scales the default parameters suggested by the statistical method based on the characteristics of the data inputted. This is particularly useful for the filter because if it is set too high the correct result cannot be returned. This would reduce the need for user experimentation with the parameters whilst still allowing the flexibility to change them. Further testing into how the statistical method performs when the sample or master chronology has errors could be conducted. Adjusting the parameters and analysing the output of the method could provide additional information and will likely be more tolerant of errors at the start and end of the chronology. This differs from current crossdating programs, such as RingDateR, which is good at identifying possible errors in the middle of the sample using a heatmap so some combinations of the program's methods could be more error resistant [1]. Secondly, the machine learning method could be refined or reapproached with a different machine learning model. Despite its failings, it did crossdate 94% of test data accurately which shows some promise for machine learning based crossdating approaches in the future. One potential approach to improve the method would be to implement the filter used in the statistical method for the machine learning techniques, this would be possible as the methods both count start years. It would require additional testing to suggest a good level for the filter. Additionally, time could be spent on creating a larger training database to allow future projects to work with a wide variety of data in the same format and file type. This may make the machine learning method more accurate and could be used as a default training dataset if one could not be provided by the user. This would encourage the development of machine learning methods because sourcing reputable data that does not require intensive preprocessing is a challenge of machine learning [45].

6 Conclusion

This project investigated two new approaches to crossdating, one statistical and one machine learning. Each method was refined through experimentation in order to optimise them. There were three versions of each method. After each version, the strengths and weaknesses of the methods were assessed and improvements were made.

The statistical method created was based on pairwise lead-lag analysis. It split up the master and sample chronology into segments of length 10 and paired all the segments of the master chronology with all the segments of the sample chronology. Then the t values for all the pairs are calculated and pairs with t values more than 3 standard deviations away from the mean are classified as outliers. A filter was applied to remove outliers where the next eight consecutive segments in the master and sample chronology were not outliers. This reduces the number of remaining pairs. The start years for the sample segments are calculated using the index of each of the master chronologies and the sample chronology and the occurrence of each start year are counted up. The highest occurrence is outputted as the most likely possible start year. The statistical method required extensive experimentation in order to develop a set of default parameters used in the description of the method that would perform well for most datasets. The variable parameters are the length of the segments, the number of standard deviations away from the mean that is required to be an outlier and the strength of the filter. The default parameters for the method had to be considered holistically as each affected the performance of the other.

The Machine learning method used three MLP binary classifiers where the output of one is used as the input of the next. The first MLP predicted the sections of length 100 of the master chronology that the sample aligns with. This is to reduce the size of the master chronology which does not align with the sample in order to reduce the large class imbalance in the input for the next MLP. The class 1 segments of length 100 are split into overlapping segments of length 10 and paired with every possible sample segment. The second MLP predicts if a segment of length 10 of the master chronology and a segment of length 10 of the sample chronology overlap by at least 5 years. The pairs with class 1 outputs of the second MLP are then inputted straight into the third MLP which predicts if the pairs are perfectly aligned. The start years for the pairs with class 1 predictions are calculated and the same counting function used in the statistical method is applied to find the best possible alignment. This method only began producing the correct answer in the final version, the class imbalance meant the training data had to be undersampled which increased the amount of false positives. This affected the output of the method because the counting method relies on the correct start year having the highest count, which is not always the case with a high number of unpredictable false positives. Due to this, the method failed 2 of the 33 test cases.

Once both methods were complete, they were added to an app with a basic GUI which runs the program and produced helpful graphs such as a bar graph of the top 15 start years and a line graph of the master chronology and the sample chronology in the best alignment. Through testing it can be concluded that the statistical method was more successful. It passed all of the test cases, took just 1% of the time that the machine learning method did and was flexible with the ability to edit parameters. As such, this method could be integrated into current programs in the field and the counting function can help reduce the effect of errors on the crossdating process. The project can be extended by testing the statistical method on datasets that contain errors to determine how the method performs. Further experimentation by testing the method with varying parameters, especially by varying the filter, would increase the ability to set dynamic default parameters which would most benefit the user. Additionally, this project suggests that with some refinement machine learning can be used to crossdate and more research in this area may produce effective programs. Overall, this project has been successful in exploring alternative methods of crossdating.

References

[1] D. J. Reynolds, D. C. Edge, and B. A. Black, "Ringdater: A statistical and graphical tool for crossdating," *Dendrochronologia*, vol. 65, 2021, ISSN: 1125-7865. DOI: https://doi.org/10.1016/j.dendro.2020.125797.

[2] T. Wigley, P. Jones, and K. Briffa, "Cross-dating methods in dendrochronology," *Journal of Archaeological Science*, vol. 14, no. 1, pp. 51–64, 1987, ISSN: 0305-4403. DOI: https://doi.org/10.1016/S0305-4403(87)80005-5.

[3] J. G. A. Lageard, "Dendrochronology," *Encyclopedia of Geoarchaeology*, A. S. Gilbert, Ed., pp. 180–197, 2017. DOI: 10.1007/978-1-4020-4409-0_41.

[4] D. J. Reynolds, V. R. von Biela, K. H. Dunton, D. C. Douglas, and B. A. Black, "Sclerochronological records of environmental variability and bivalve growth in the pacific arctic," *Progress in Oceanography*, vol. 206, 2022.

[5] M. Kaennel and F. H. Schweingruber, "Multilingual glossary of dendrochronology," *Swiss Federal Institute for Forest, Snow and Landscape Research. Berne, Stuttgart, Vienna, Haupt*, vol. 133, pp. 162–184, 1995.

[6] R. Laxton and C. Litton, "Construction of a kent master dendrochronological sequence for oak, ad 1158 to 1540," *Medieval archaeology*, vol. 33, no. 1, pp. 90–98, 1989.

[7] J. J. Camarero, M. Colangelo, A. Gracia-Balaga, M. A. Ortega-Martínez, and U. Büntgen, "Demystifying the age of old olive trees," *Dendrochronologia*, vol. 65, 2021, ISSN: 1125-7865. DOI: https://doi.org/10.1016/j.dendro.2020.125802.

[8] M. Baillie, "Some thoughts on art-historical dendrochronology," *Journal of Archaeological Science*, vol. 11, no. 5, pp. 371–393, 1984, ISSN: 0305-4403. DOI: https://doi.org/10.1016/0305-4403(84)90019-0.

[9] R. Allan and T. Ansell, "A new globally complete monthly historical gridded mean sea level pressure dataset (hadslp2): 1850–2004," *Journal of Climate*, vol. 19, no. 22, pp. 5816–5842, 2006.

[10] Q. He and B. R. Silliman, "Climate change, human impacts, and coastal ecosystems in the anthropocene," *Current Biology*, vol. 29, no. 19, R1021–R1035, 2019, ISSN: 0960-9822. DOI: https://doi.org/10.1016/j.cub.2019.08.042.

[11] J. Adams and F. Woodward, "The past as a key to the future: The use of palaeoenvironmental understanding to predict the effects of man on the biosphere," in *The Ecological Consequences of Global Climate Change*, ser. Advances in Ecological Research, M. Begon, A. Fitter, and A. Macfadyen, Eds., vol. 22, Academic Press, 1992, pp. 257–314. DOI: https://doi.org/10.1016/S0065-2504(08)60138-5.

[12] F. H. Schweingruber, "Tree rings: Basics and applications of dendrochronology," in Springer Netherlands, 1988, pp. 1–4, ISBN: 978-94-009-1273-1. DOI: 10.1007/978-94-009-1273-1_1.

[13] M. Baillie, "Seven thousand years of alternative history: The tree-ring story.," *Irish Forestry*, vol. 56, 1999.

[14] E.-B. Choi, Y.-J. Kim, J.-H. Park, C.-R. Park, and J.-W. Seo, "Reconstruction of resin collection history of pine forests in korea from tree-ring dating," *Sustainability*, vol. 12, Nov. 2020. DOI: 10.3390/su12219118.

[15] B. Gmińska-Nowak, A. D'Agostino, Y. Özarslan, *et al.*, "Dendrochronological analysis and radiocarbon dating of charcoal remains from the multi-period site of uşaklı höyük, yozgat, turkey," *Journal of Archaeological Science: Reports*, vol. 38, p. 103 078, 2021, ISSN: 2352-409X. DOI: https://doi.org/10.1016/j.jasrep.2021.103078.

[16] E. Cook and N. Pederson, "Uncertainty, emergence, and statistics in dendrochronology," *Dendroclimatology: Progress and Prospects*, vol. 11, Sep. 2011. DOI: 10.1007/978-1-4020-5725-0_4.

[17] S. A. Johnson, S. I. Gass, and M. C. Fu, "Splines," in *Encyclopedia of Operations Research and Management Science*, Springer US, 2013, pp. 1443–1446, ISBN: 978-1-4419-1153-7. DOI: 10.1007/978-1-4419-1153-7_982.

[18] D. Frank, K. Fang, and P. Fonti, "Dendrochronology: Fundamentals and innovations," in *Stable Isotopes in Tree Rings: Inferring Physiological, Climatic and Environmental Responses*, R. T. W. Siegwolf, J. R. Brooks, J. Roden, and M. Saurer, Eds. Springer International Publishing, 2022, pp. 21–59. DOI: 10.1007/978-3-030-92698-4_2.

[19] P. R. Sheppard, *Making skeleton plots*, (Accessed: 14th of September 2022). [Online]. Available: https://www.ltrr.arizona.edu/skeletonplot/plotting.htm.

[20] H. D. Grissino-Mayer, "Evaluating crossdating accuracy: A manual and tutorial for the computer program cofecha," *Tree-Ring Research*, vol. 57, no. 2, 2001.

[21] P. O. Larsson and L.-Å. Larsson, "Match a sample towards a reference curve," *Cybis.se: Technical writing, software development, Dendrochronology*, Jun. 2015. [Online]. Available: https://cdendro.se/forfun/dendro/index.htm.

[22] S. Bennett, M. Cucuringu, and G. Reinert, "Lead-lag detection and network clustering for multivariate time series with an application to the us equity market," pp. 1–42, 2022. DOI: https://doi.org/10.1007/s10994-022-06250-4.

[23] T. Derrick and J. Thomas, "Time series analysis: The cross-correlation function," *Innovative analyses of human movement*, pp. 189–205, 2004.

[24] P. G. Butler, A. D. Wanamaker, J. D. Scourse, C. A. Richardson, and D. J. Reynolds, "Variability of marine climate on the north icelandic shelf in a 1357-year proxy archive based on growth increments in the bivalve arctica islandica," *Palaeogeography, Palaeoclimatology, Palaeoecology*, vol. 373, pp. 141–151, 2013, ISSN: 0031-0182. DOI: https://doi.org/10.1016/j.palaeo.2012.01.016.

[25] S. Dash and A. Dash, "A correlation based multilayer perceptron algorithm for cancer classification with gene-expression dataset," *2014 14th International Conference on Hybrid Intelligent Systems*, pp. 158–163, 2014. DOI: 10.1109/HIS.2014.7086190.

[26] T. Windeatt, "Accuracy/diversity and ensemble mlp classifier design," *IEEE Transactions on Neural Networks*, vol. 17, no. 5, pp. 1194–1211, 2006.

[27] *Tree ring database*, (Accessed: 26th September 2022). [Online]. Available: https://www.ncei.noaa.gov/products/paleoclimatology/tree-ring.

[28] R. Adams, K. Baird, L. Baxter, L. Pedicino, K. Scotti, and A. Tissescu, "NOAA/WDS Paleoclimatology - Scheelite Canyon Huachuca Mountains - PSME - ITRDB AZ556," *NOAA National Centers for Environmental Information*, 2002. DOI: https://doi.org/10.25921/tff4-e451.

[29] A. G. Bunn, "A dendrochronology program library in r (dplr)," *Dendrochronologia*, vol. 26, no. 2, pp. 115–124, 2008. DOI: https://doi.org/10.1016/j.dendro.2008.01.002.

[30] *Real statistics functions*, (Accessed: 26th January 2022). [Online]. Available: https://real-statistics.com/descriptive-statistics/m-estimators/.

[31] W. McKinney *et al.*, *Pandas dataframe*, Pandas package Version: 2.0.1 (Accessed: 28th March 2022). [Online]. Available: https://pandas.pydata.org/docs/index.html.

[32] M. D. John D. Hunter *et al.*, *Matplotlib*, Matplotlib package Version: 3.7.1 (Accessed: 10th March 2022). [Online]. Available: https://matplotlib.org/stable/index.html.

[33] R. L. Holmes, "Computer-assisted quality control in tree-ring dating and measurement," *Tree-Ring Bulletin*, vol. 43, pp. 51–67, 1983.

[34] D. Cournapeau *et al.*, *Scikit-learn*, scikit-learn package Version: 1.2.2 (Accessed: 5th April 2022). [Online]. Available: https://scikit-learn.org/stable/.

[35] F. Murtagh, "Multilayer perceptrons for classification and regression," *Neurocomputing*, vol. 2, no. 5, pp. 183–197, 1991, ISSN: 0925-2312. DOI: https://doi.org/10.1016/0925-2312(91)90023-5.

[36] R. Kohavi *et al.*, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Ijcai*, Montreal, Canada, vol. 14, 1995, pp. 1137–1145.

[37] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intelligent data analysis*, vol. 6, no. 5, pp. 429–449, 2002.

[38] D. Cournapeau *et al.*, *Cprofile*, The Python Standard Library 3.11.3: cProfile (Accessed: 20th April 2022). [Online]. Available: https://docs.python.org/3/library/profile.html.

[39] *Ringdater version 1.4*, (Accessed: 24th April 2022). [Online]. Available: https://ringdater.shinyapps.io/ringdater_beta_v1_4/.

[40] T. Dietterich, "Overfitting and undercomputing in machine learning," *ACM computing surveys (CSUR)*, vol. 27, no. 3, pp. 326–327, 1995.

[41] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature machine intelligence*, vol. 1, no. 5, pp. 206–215, 2019.

[42] H. Kaur, H. S. Pannu, and A. K. Malhi, "A systematic review on imbalanced data challenges in machine learning: Applications and solutions," *ACM Computing Surveys (CSUR)*, vol. 52, no. 4, pp. 1–36, 2019.

[43] D. M. Skapura, *Building neural networks*. Addison-Wesley Professional, 1996.

[44] S. Sridevi, S. Parthasarathy, and S. Rajaram, "An effective prediction system for time series data using pattern matching algorithms.," *International Journal of Industrial Engineering*, vol. 25, no. 2, 2018.

[45] L. Zhou, S. Pan, J. Wang, and A. V. Vasilakos, "Machine learning on big data: Opportunities and challenges," *Neurocomputing*, vol. 237, pp. 350–361, 2017.

8 Appendix

8.1 Pairwise Lead Lag Analysis Diagram

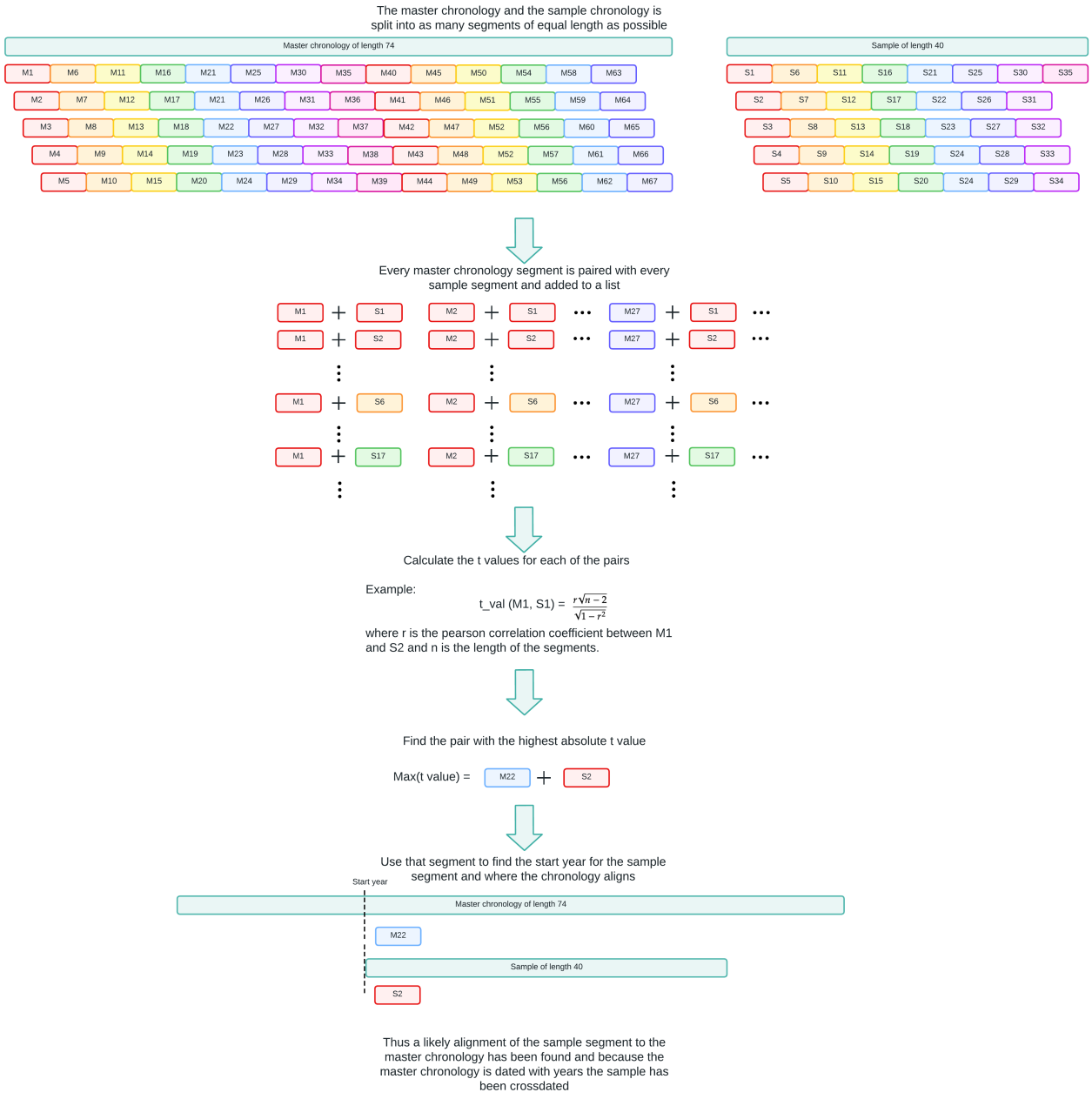

Figure 8.1: Diagram overviewing a simplified version of the pairwise lead-lag analysis approach

8.2 Statistical method Diagram

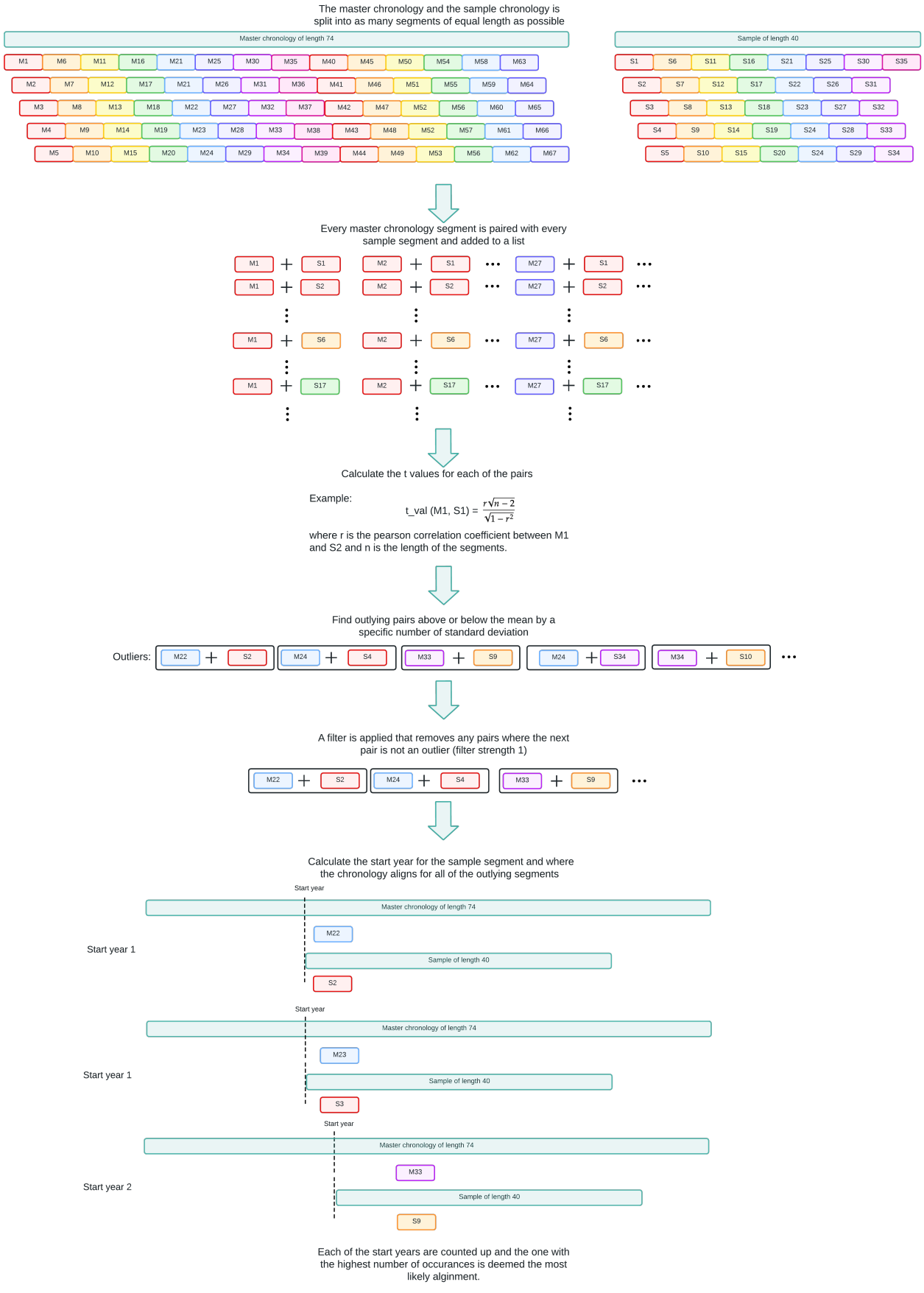

Figure 8.2: Diagram overviewing a simplified version of the statistical method developed in this project.

8.3 Segment Length Graphs

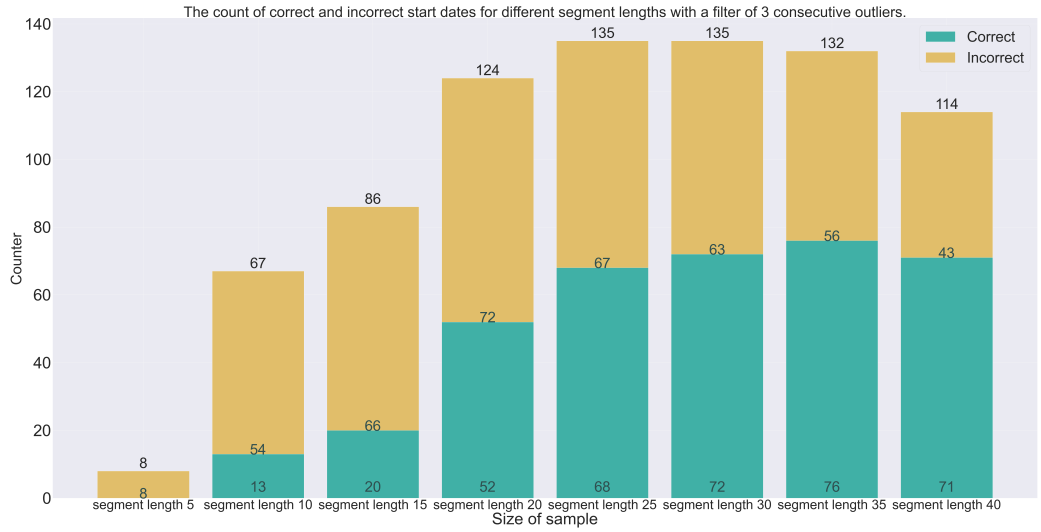

Figure 8.3: Graph depicting the proportion for which the outlying values (over 3 standard deviations from the mean) predict the correct start year for the chronology with a filter of length 3 for different segment lengths.

8.4 Filter Strength Graphs

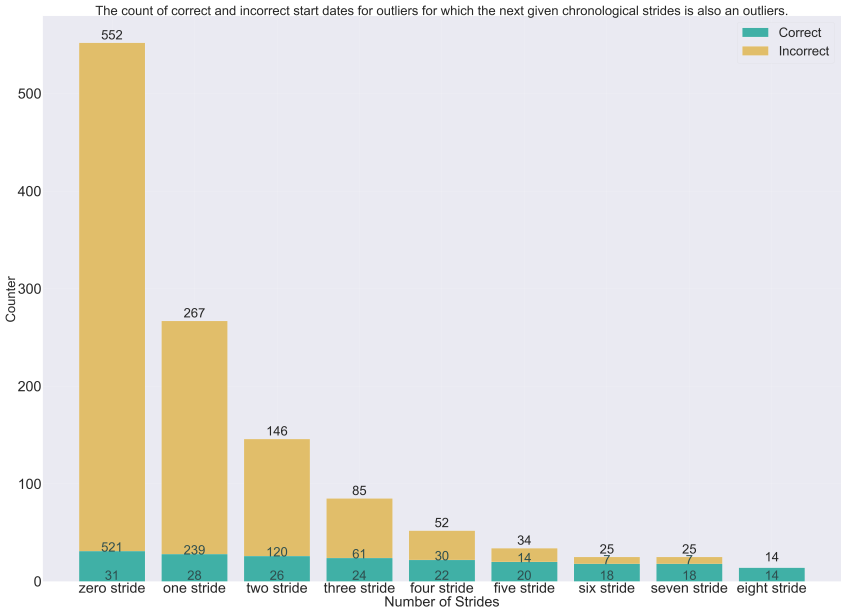

Figure 8.4: Graph depicting the proportion for which the outlying values (over 3 standard deviations from the mean) predict the correct start year for the chronology with different filter strengths.

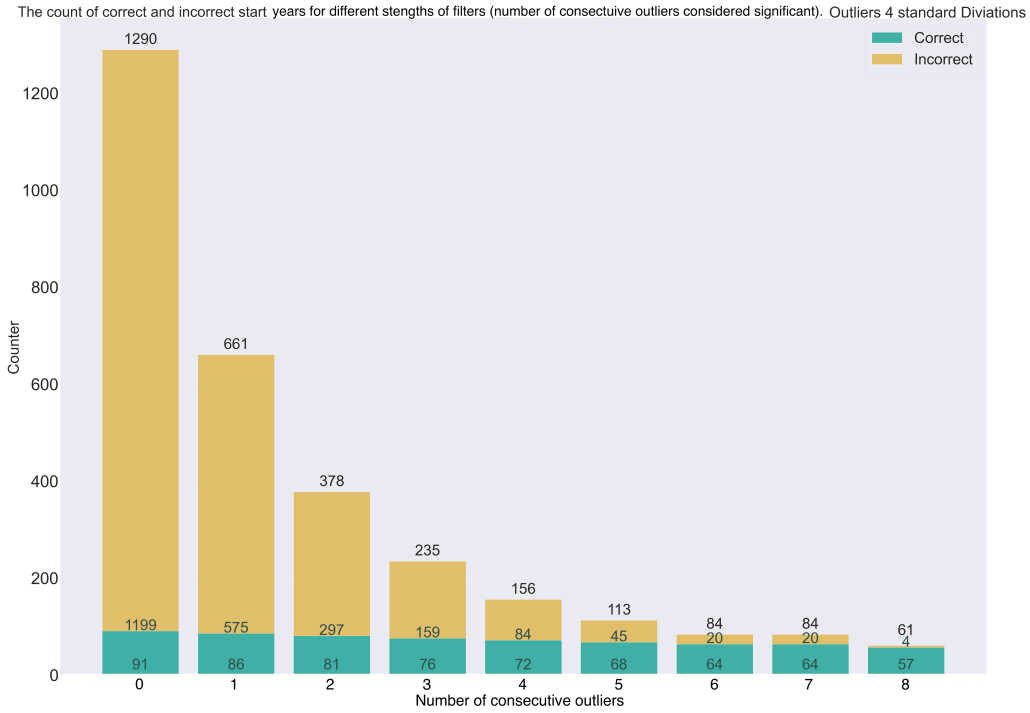

Figure 8.5: Graph depicting the proportion for which the outlying values (over 4 standard deviations from the mean) predict the correct start year for the chronology with different filter strengths.

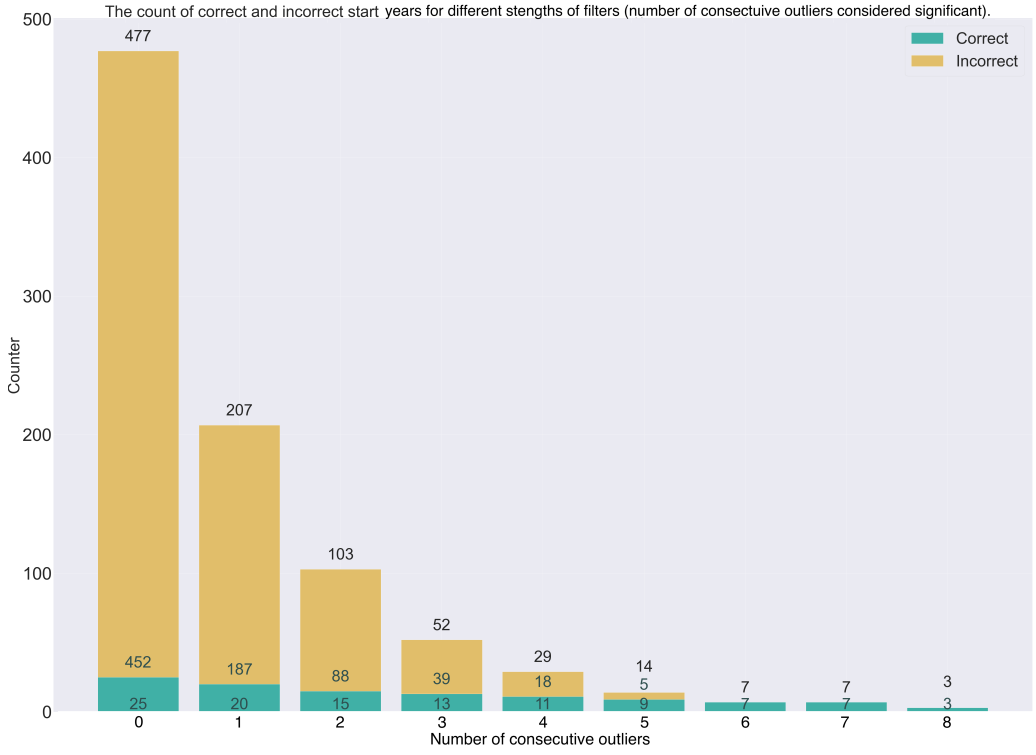

Figure 8.6: Graph depicting the proportion for which the outlying values (over 3 standard deviations from the mean) predict the correct start year for the chronology with different filter strengths.

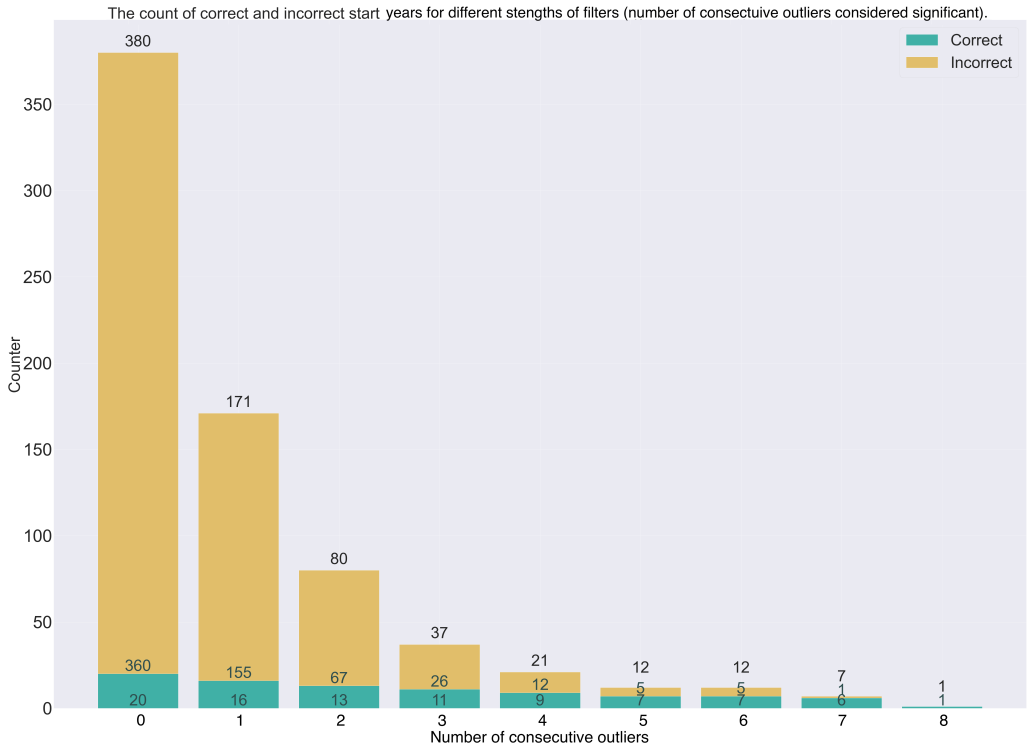

Figure 8.7: Graph depicting the proportion for which the outlying values (over 3 standard deviations from the mean) predict the correct start year for the chronology with different filter strengths.

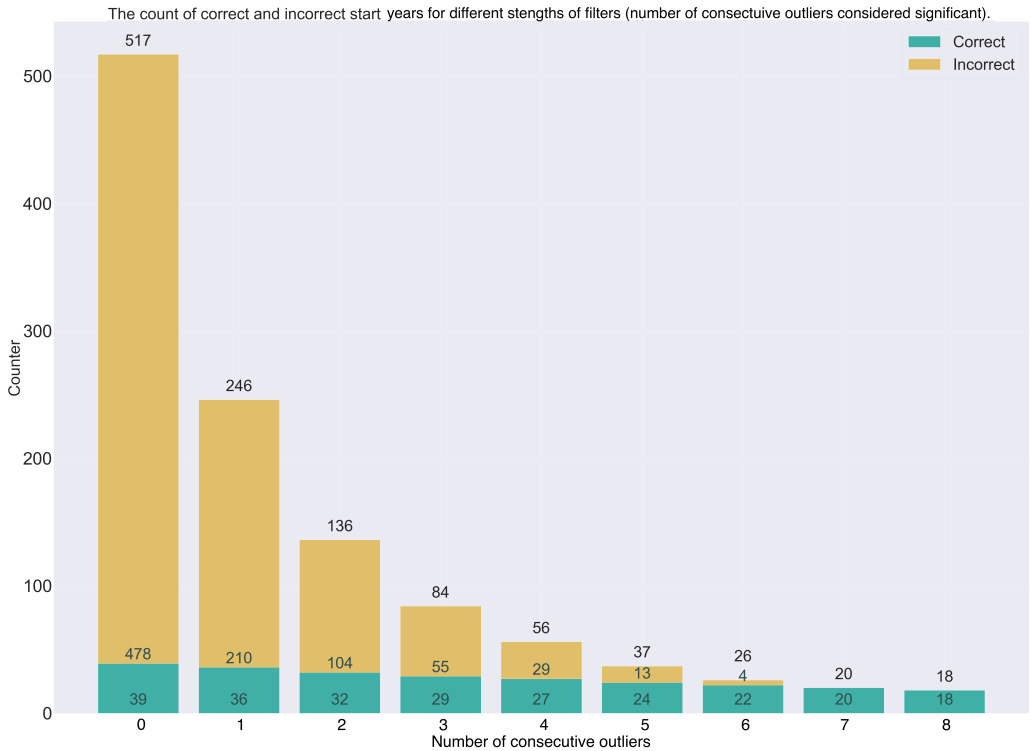

Figure 8.8: Graph depicting the proportion for which the outlying values (over 3 standard deviations from the mean) predict the correct start year for the chronology with different filter strengths.

8.5 MLP Results

```
0.883 (+/-0.030) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (130,), 'le
arning_rate': 'constant', 'max_iter': 10000, 'solver': 'adam'}
0.883 (+/-0.061) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (130,), 'le
arning_rate': 'constant', 'max_iter': 50000, 'solver': 'adam'}
0.875 (+/-0.049) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (130,), 'le
arning_rate': 'constant', 'max_iter': 100000, 'solver': 'adam'}
0.890 (+/-0.037) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (130,), 'le
arning_rate': 'constant', 'max_iter': 250000, 'solver': 'adam'}
0.868 (+/-0.057) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (150,), 'le
arning_rate': 'constant', 'max_iter': 10000, 'solver': 'adam'}
0.889 (+/-0.043) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (150,), 'le
arning_rate': 'constant', 'max_iter': 50000, 'solver': 'adam'}
0.893 (+/-0.046) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (150,), 'le
arning_rate': 'constant', 'max_iter': 100000, 'solver': 'adam'}
0.880 (+/-0.066) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (150,), 'le
arning_rate': 'constant', 'max_iter': 250000, 'solver': 'adam'}
0.892 (+/-0.051) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (200,), 'le
arning_rate': 'constant', 'max_iter': 10000, 'solver': 'adam'}
0.880 (+/-0.085) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (200,), 'le
arning_rate': 'constant', 'max_iter': 50000, 'solver': 'adam'}
0.893 (+/-0.044) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (200,), 'le
arning_rate': 'constant', 'max_iter': 100000, 'solver': 'adam'}
0.884 (+/-0.070) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (200,), 'le
arning_rate': 'constant', 'max_iter': 250000, 'solver': 'adam'}
0.887 (+/-0.059) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (300,), 'le
arning_rate': 'constant', 'max_iter': 10000, 'solver': 'adam'}
0.890 (+/-0.051) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (300,), 'le
arning_rate': 'constant', 'max_iter': 50000, 'solver': 'adam'}
0.887 (+/-0.054) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (300,), 'le
arning_rate': 'constant', 'max_iter': 100000, 'solver': 'adam'}
0.890 (+/-0.056) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (300,), 'le
arning_rate': 'constant', 'max_iter': 250000, 'solver': 'adam'}
```

Figure 8.9: Accuracy with standard deviation report for every combination of the parameters in the Parameters Grid for the first version of the Machine Learning Method.

```
0.700 (+/-0.512) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (150,),
'learning_rate': 'constant', 'max_iter': 2000, 'solver': 'adam'}
0.650 (+/-0.657) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (150,),
'learning_rate': 'constant', 'max_iter': 2300, 'solver': 'adam'}
0.650 (+/-0.657) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (150,),
'learning_rate': 'constant', 'max_iter': 2600, 'solver': 'adam'}
0.650 (+/-0.657) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (150,),
'learning_rate': 'constant', 'max_iter': 2500, 'solver': 'adam'}
0.650 (+/-0.657) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (150,),
'learning_rate': 'constant', 'max_iter': 2750, 'solver': 'adam'}
0.650 (+/-0.657) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (150,),
'learning_rate': 'constant', 'max_iter': 5000, 'solver': 'adam'}
0.650 (+/-0.795) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (150, 15
0), 'learning_rate': 'constant', 'max_iter': 2000, 'solver': 'adam'}
0.667 (+/-0.775) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (150, 15
0), 'learning_rate': 'constant', 'max_iter': 2300, 'solver': 'adam'}
0.650 (+/-0.657) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (150, 15
0), 'learning_rate': 'constant', 'max_iter': 2600, 'solver': 'adam'}
0.650 (+/-0.482) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (150, 15
0), 'learning_rate': 'constant', 'max_iter': 2500, 'solver': 'adam'}
0.750 (+/-0.522) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (150, 15
0), 'learning_rate': 'constant', 'max_iter': 2750, 'solver': 'adam'}
0.650 (+/-0.657) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (150, 15
0), 'learning_rate': 'constant', 'max_iter': 5000, 'solver': 'adam'}
0.650 (+/-0.657) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (120, 13
0), 'learning_rate': 'constant', 'max_iter': 2000, 'solver': 'adam'}
0.600 (+/-0.618) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (120, 13
0), 'learning_rate': 'constant', 'max_iter': 2300, 'solver': 'adam'}
0.650 (+/-0.657) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (120, 13
0), 'learning_rate': 'constant', 'max_iter': 2600, 'solver': 'adam'}
0.550 (+/-0.716) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (120, 13
0), 'learning_rate': 'constant', 'max_iter': 2500, 'solver': 'adam'}
0.650 (+/-0.657) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (120, 13
0), 'learning_rate': 'constant', 'max_iter': 2750, 'solver': 'adam'}
0.667 (+/-0.775) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (120, 13
0), 'learning_rate': 'constant', 'max_iter': 5000, 'solver': 'adam'}
0.650 (+/-0.657) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (140,),
'learning_rate': 'constant', 'max_iter': 2000, 'solver': 'adam'}
0.650 (+/-0.657) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (140,),
'learning_rate': 'constant', 'max_iter': 2300, 'solver': 'adam'}
0.617 (+/-0.684) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (140,),
'learning_rate': 'constant', 'max_iter': 2600, 'solver': 'adam'}
0.650 (+/-0.657) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (140,),
'learning_rate': 'constant', 'max_iter': 2500, 'solver': 'adam'}
0.650 (+/-0.657) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (140,),
'learning_rate': 'constant', 'max_iter': 2750, 'solver': 'adam'}
0.567 (+/-0.636) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (140,),
'learning_rate': 'constant', 'max_iter': 5000, 'solver': 'adam'}
0.650 (+/-0.657) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (160, 15
0), 'learning_rate': 'constant', 'max_iter': 2000, 'solver': 'adam'}
0.700 (+/-0.512) for {'activation': 'relu', 'alpha': 0.0001, 'early_stopping': False, 'hidden_layer_sizes': (160, 15
0), 'learning_rate': 'constant', 'max_iter': 2300, 'solver': 'adam'}
```

Figure 8.10: Accuracy with standard deviation report for every combination of the parameters in the Parameters Grid for the first MLP in the second version of the Machine Learning Method.

```
0.988 (+/-0.009) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (100,), 'le
arning_rate': 'constant', 'max_iter': 100, 'solver': 'adam'}
0.988 (+/-0.009) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (100,), 'le
arning_rate': 'constant', 'max_iter': 500, 'solver': 'adam'}
0.986 (+/-0.012) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (100,), 'le
arning_rate': 'constant', 'max_iter': 1000, 'solver': 'adam'}
0.987 (+/-0.011) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (100,), 'le
arning_rate': 'constant', 'max_iter': 2000, 'solver': 'adam'}
0.987 (+/-0.010) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (150,), 'le
arning_rate': 'constant', 'max_iter': 100, 'solver': 'adam'}
0.987 (+/-0.010) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (150,), 'le
arning_rate': 'constant', 'max_iter': 500, 'solver': 'adam'}
0.986 (+/-0.011) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (150,), 'le
arning_rate': 'constant', 'max_iter': 1000, 'solver': 'adam'}
0.987 (+/-0.009) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (150,), 'le
arning_rate': 'constant', 'max_iter': 2000, 'solver': 'adam'}
0.987 (+/-0.009) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (200,), 'le
arning_rate': 'constant', 'max_iter': 100, 'solver': 'adam'}
0.987 (+/-0.009) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (200,), 'le
arning_rate': 'constant', 'max_iter': 500, 'solver': 'adam'}
0.987 (+/-0.010) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (200,), 'le
arning_rate': 'constant', 'max_iter': 1000, 'solver': 'adam'}
0.986 (+/-0.010) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (200,), 'le
arning_rate': 'constant', 'max_iter': 2000, 'solver': 'adam'}
0.987 (+/-0.009) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (300,), 'le
arning_rate': 'constant', 'max_iter': 100, 'solver': 'adam'}
0.987 (+/-0.009) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (300,), 'le
arning_rate': 'constant', 'max_iter': 500, 'solver': 'adam'}
0.987 (+/-0.009) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (300,), 'le
arning_rate': 'constant', 'max_iter': 1000, 'solver': 'adam'}
0.986 (+/-0.010) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (300,), 'le
arning_rate': 'constant', 'max_iter': 2000, 'solver': 'adam'}
```

Figure 8.11: Accuracy with standard deviation report for every combination of the parameters in the Parameters Grid for the first MLP in the third version of the Machine Learning Method.

```
0.517 (+/-0.504) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (120, 130),
'learning_rate': 'constant', 'max_iter': 2750, 'solver': 'adam'}
0.483 (+/-0.504) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (120, 130),
'learning_rate': 'constant', 'max_iter': 5000, 'solver': 'adam'}
0.433 (+/-0.581) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (120, 130),
'learning_rate': 'constant', 'max_iter': 50694, 'solver': 'adam'}
0.400 (+/-0.562) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (120, 130),
'learning_rate': 'constant', 'max_iter': 60000, 'solver': 'adam'}
0.483 (+/-0.504) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (140,), 'le
arning_rate': 'constant', 'max_iter': 2000, 'solver': 'adam'}
0.517 (+/-0.504) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (140,), 'le
arning_rate': 'constant', 'max_iter': 2300, 'solver': 'adam'}
0.483 (+/-0.504) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (140,), 'le
arning_rate': 'constant', 'max_iter': 2600, 'solver': 'adam'}
0.450 (+/-0.597) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (140,), 'le
arning_rate': 'constant', 'max_iter': 2500, 'solver': 'adam'}
0.517 (+/-0.504) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (140,), 'le
arning_rate': 'constant', 'max_iter': 2750, 'solver': 'adam'}
0.400 (+/-0.562) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (140,), 'le
arning_rate': 'constant', 'max_iter': 5000, 'solver': 'adam'}
0.483 (+/-0.407) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (140,), 'le
arning_rate': 'constant', 'max_iter': 50694, 'solver': 'adam'}
0.400 (+/-0.476) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (140,), 'le
arning_rate': 'constant', 'max_iter': 60000, 'solver': 'adam'}
0.533 (+/-0.593) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (160, 150),
'learning_rate': 'constant', 'max_iter': 2000, 'solver': 'adam'}
0.567 (+/-0.371) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (160, 150),
'learning_rate': 'constant', 'max_iter': 2300, 'solver': 'adam'}
0.533 (+/-0.389) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (160, 150),
'learning_rate': 'constant', 'max_iter': 2600, 'solver': 'adam'}
0.550 (+/-0.496) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (160, 150),
'learning_rate': 'constant', 'max_iter': 2500, 'solver': 'adam'}
0.533 (+/-0.512) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (160, 150),
'learning_rate': 'constant', 'max_iter': 2750, 'solver': 'adam'}
0.450 (+/-0.396) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (160, 150),
'learning_rate': 'constant', 'max_iter': 5000, 'solver': 'adam'}
0.483 (+/-0.504) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (160, 150),
'learning_rate': 'constant', 'max_iter': 50694, 'solver': 'adam'}
0.483 (+/-0.504) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (160, 150),
'learning_rate': 'constant', 'max_iter': 60000, 'solver': 'adam'}
0.450 (+/-0.396) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (164, 160,
120), 'learning_rate': 'constant', 'max_iter': 2000, 'solver': 'adam'}
0.483 (+/-0.504) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (164, 160,
120), 'learning_rate': 'constant', 'max_iter': 2300, 'solver': 'adam'}
0.567 (+/-0.371) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (164, 160,
120), 'learning_rate': 'constant', 'max_iter': 2600, 'solver': 'adam'}
0.500 (+/-0.516) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (164, 160,
120), 'learning_rate': 'constant', 'max_iter': 2500, 'solver': 'adam'}
0.517 (+/-0.674) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (164, 160,
120), 'learning_rate': 'constant', 'max_iter': 2750, 'solver': 'adam'}
0.433 (+/-0.581) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': False, 'hidden_layer_sizes': (164, 160,
120), 'learning_rate': 'constant', 'max_iter': 5000, 'solver': 'adam'}
```

Figure 8.12: Accuracy with standard deviation report for every combination of the parameters in the Parameters Grid for the second MLP in the third version of the Machine Learning Method.

```
0.507 (+/-0.232) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500,), 'lea
rning_rate': 'adaptive', 'max_iter': 1000, 'solver': 'adam'}
0.500 (+/-0.400) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500,), 'lea
rning_rate': 'adaptive', 'max_iter': 2000, 'solver': 'adam'}
0.500 (+/-0.400) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500,), 'lea
rning_rate': 'adaptive', 'max_iter': 5000, 'solver': 'adam'}
0.527 (+/-0.427) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500, 150),
'learning_rate': 'constant', 'max_iter': 100, 'solver': 'adam'}
0.507 (+/-0.232) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500, 150),
'learning_rate': 'constant', 'max_iter': 200, 'solver': 'adam'}
0.460 (+/-0.392) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500, 150),
'learning_rate': 'constant', 'max_iter': 500, 'solver': 'adam'}
0.553 (+/-0.487) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500, 150),
'learning_rate': 'constant', 'max_iter': 1000, 'solver': 'adam'}
0.540 (+/-0.160) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500, 150),
'learning_rate': 'constant', 'max_iter': 2000, 'solver': 'adam'}
0.427 (+/-0.311) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500, 150),
'learning_rate': 'constant', 'max_iter': 5000, 'solver': 'adam'}
0.340 (+/-0.431) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500, 150),
'learning_rate': 'invscaling', 'max_iter': 100, 'solver': 'adam'}
0.347 (+/-0.155) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500, 150),
'learning_rate': 'invscaling', 'max_iter': 200, 'solver': 'adam'}
0.540 (+/-0.299) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500, 150),
'learning_rate': 'invscaling', 'max_iter': 500, 'solver': 'adam'}
0.267 (+/-0.304) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500, 150),
'learning_rate': 'invscaling', 'max_iter': 1000, 'solver': 'adam'}
0.420 (+/-0.265) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500, 150),
'learning_rate': 'invscaling', 'max_iter': 2000, 'solver': 'adam'}
0.307 (+/-0.401) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500, 150),
'learning_rate': 'invscaling', 'max_iter': 5000, 'solver': 'adam'}
0.533 (+/-0.422) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500, 150),
'learning_rate': 'adaptive', 'max_iter': 100, 'solver': 'adam'}
0.340 (+/-0.240) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500, 150),
'learning_rate': 'adaptive', 'max_iter': 200, 'solver': 'adam'}
0.413 (+/-0.390) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500, 150),
'learning_rate': 'adaptive', 'max_iter': 500, 'solver': 'adam'}
0.353 (+/-0.314) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500, 150),
'learning_rate': 'adaptive', 'max_iter': 1000, 'solver': 'adam'}
0.380 (+/-0.196) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500, 150),
'learning_rate': 'adaptive', 'max_iter': 2000, 'solver': 'adam'}
0.580 (+/-0.265) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (500, 150),
'learning_rate': 'adaptive', 'max_iter': 5000, 'solver': 'adam'}
0.540 (+/-0.160) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (550,), 'lea
rning_rate': 'constant', 'max_iter': 100, 'solver': 'adam'}
0.607 (+/-0.373) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (550,), 'lea
rning_rate': 'constant', 'max_iter': 200, 'solver': 'adam'}
0.540 (+/-0.299) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (550,), 'lea
rning_rate': 'constant', 'max_iter': 500, 'solver': 'adam'}
0.313 (+/-0.487) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (550,), 'lea
rning_rate': 'constant', 'max_iter': 1000, 'solver': 'adam'}
0.453 (+/-0.549) for {'activation': 'relu', 'alpha': 0.05, 'early_stopping': True, 'hidden_layer_sizes': (550,), 'lea
rning_rate': 'constant', 'max_iter': 2000, 'solver': 'adam'}
```

Figure 8.13: Accuracy with standard deviation report for every combination of the parameters in the Parameters Grid for the third MLP in the third version of the Machine Learning Method.

8.6 User interface Screenshots

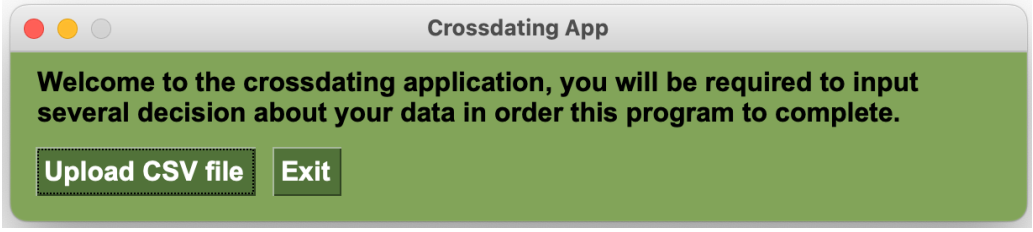

Figure 8.14: Welcome page of the application displayed immediately after starting the program.

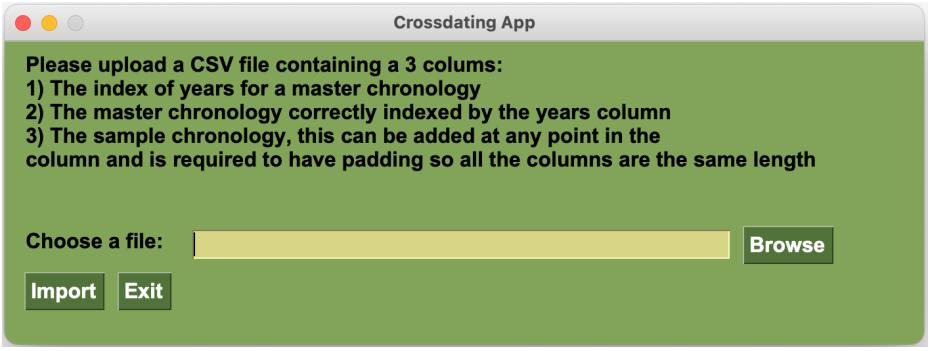

Figure 8.15: File upload page with instructions and the option to browse local files.

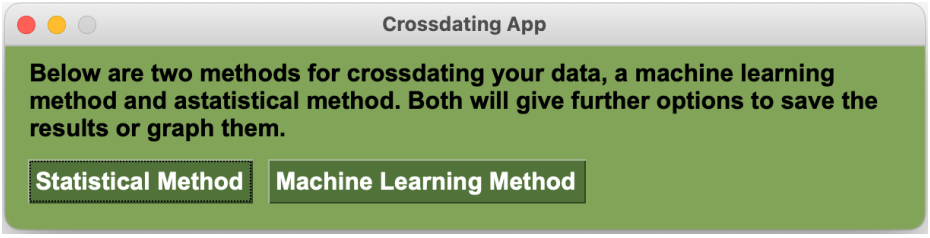

Figure 8.16: Methods page with instructions and the option to pick between methods.

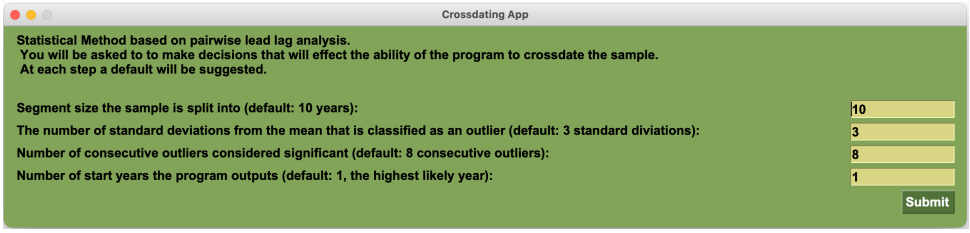

Figure 8.17: Variables page for the user to input the 4 given variables for the statistical methods, with defaults given.

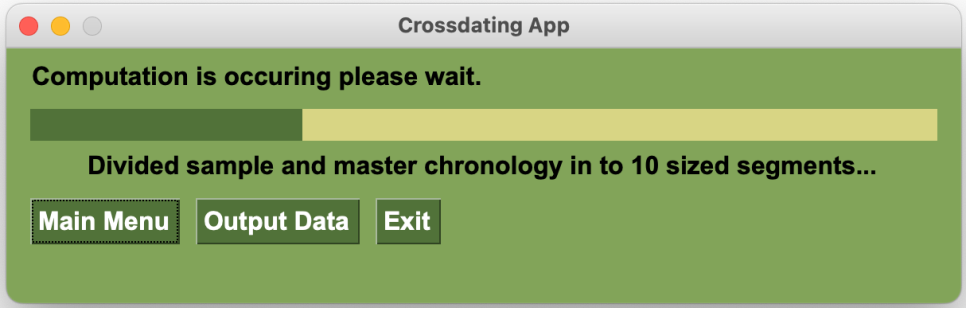

Figure 8.18: Computation page with a progress bar and options for data output.

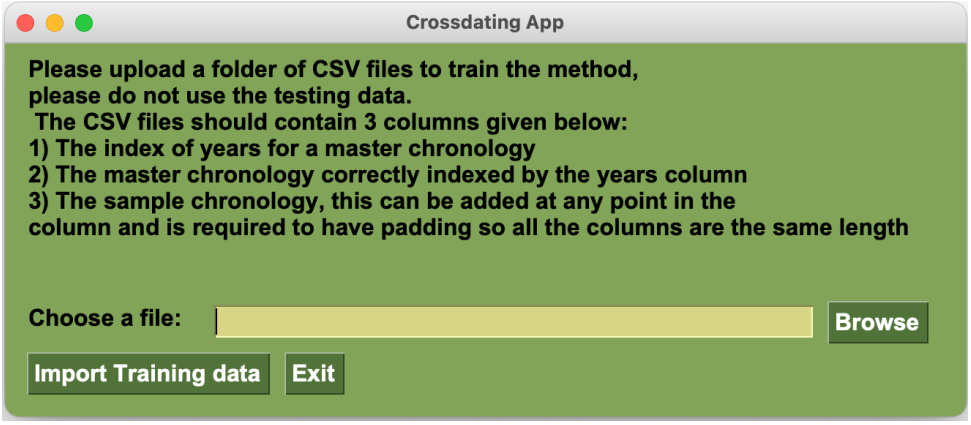

Figure 8.19: Training data input page for the machine learning method.

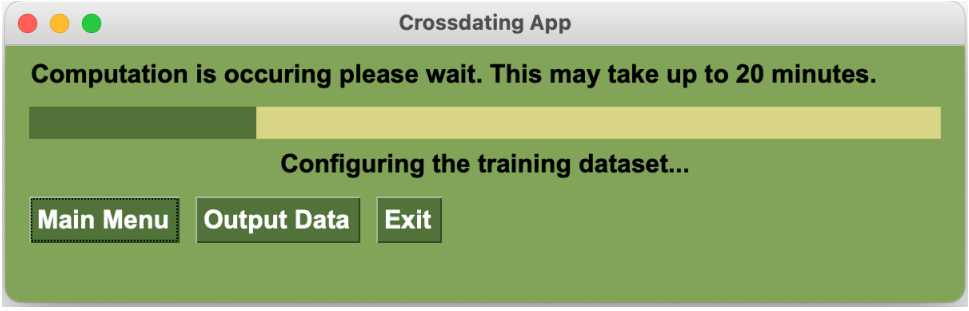

Figure 8.20: Computation page with a progress bar and options for data output.

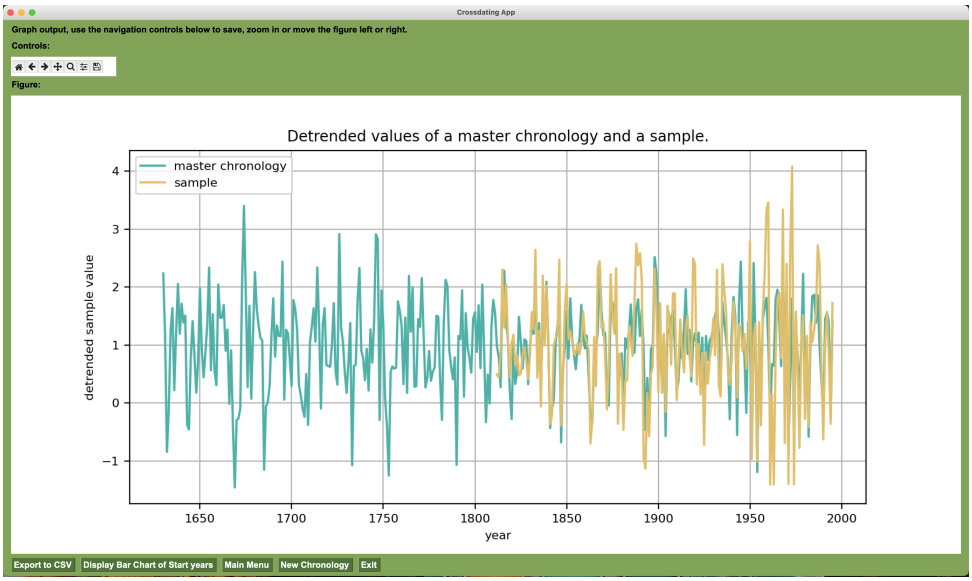

Figure 8.21: Output page for the user to see the master and sample chronology and get further options in regards to the output of the method.

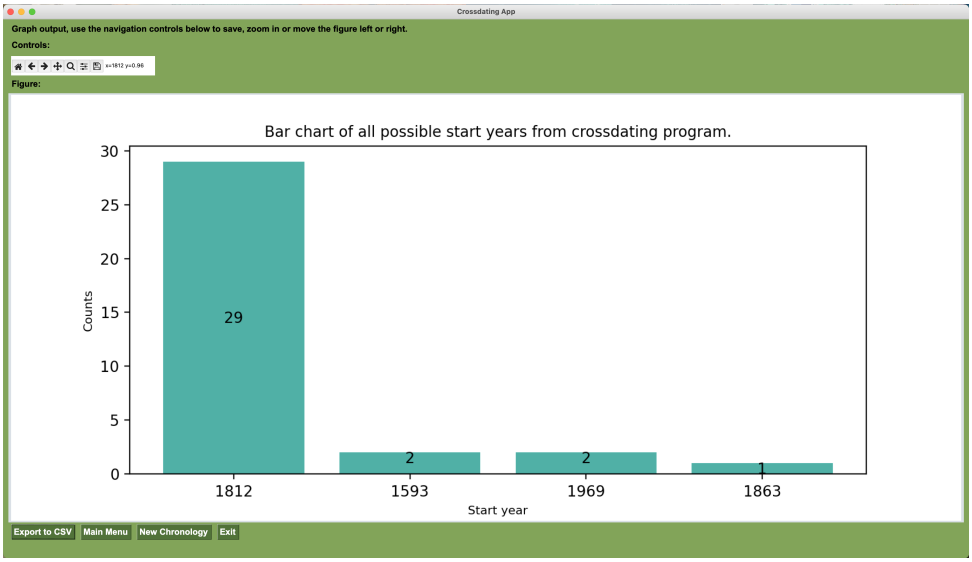

Figure 8.22: Bar chart of the possible start dates with their respective counts.

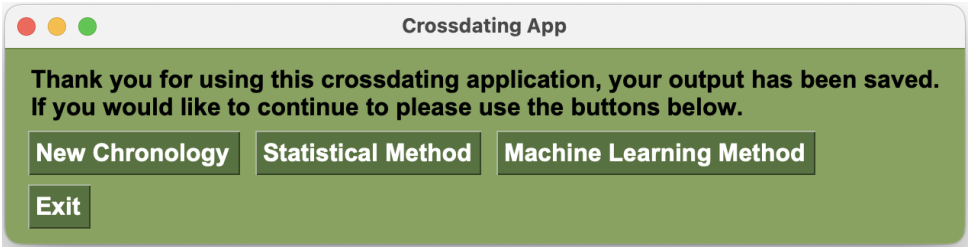

Figure 8.23: Thank you page after the results have been exported to a CSV file

8.7 Results Testing

```
7131937 function calls (6382152 primitive calls) in 8.643 seconds
Ordered by: standard name
ncalls  tottime  percall  cumtime  percall filename:lineno(function)
2    0.000    0.000    0.000    0.000 <__array_function__
    internals>:177(append)
62475    0.056    0.000    2.762    0.000 <__array_function__
    internals>:177(average)
1    0.000    0.000    0.000    0.000 <__array_function__
    internals>:177(bincount)
62475    0.043    0.000    0.833    0.000 <__array_function__
    internals>:177(broadcast_to)
62475    0.063    0.000    1.519    0.000 <__array_function__
    internals>:177(clip)
62477    0.056    0.000    0.224    0.000 <__array_function__
    internals>:177(concatenate)
6    0.000    0.000    0.000    0.000 <__array_function__
        internals>:177(copyto)
62475    0.049    0.000    7.526    0.000 <__array_function__
    internals>:177(corrcoef)
62475    0.050    0.000    4.776    0.000 <__array_function__
    internals>:177(cov)
62475    0.041    0.000    0.444    0.000 <__array_function__
    internals>:177(diag)
62475    0.038    0.000    0.247    0.000 <__array_function__
    internals>:177(diagonal)
62475    0.048    0.000    0.229    0.000 <__array_function__
    internals>:177(dot)
1    0.000    0.000    0.000    0.000 <__array_function__
    internals>:177(insert)
62475    0.035    0.000    0.133    0.000 <__array_function__
    internals>:177(iscomplexobj)
1    0.000    0.000    0.003    0.003 <__array_function__
    internals>:177(mean)
1    0.000    0.000    0.000    0.000 <__array_function__
    internals>:177(moveaxis)
124950    0.068    0.000    0.380    0.000 <__array_function__
   internals>:177(ndim)
2    0.000    0.000    0.000    0.000 <__array_function__
    internals>:177(ravel)
62475    0.041    0.000    0.152    0.000 <__array_function__
    internals>:177(result_type)
        ...
```

Figure 8.24: Output of the cProfile method for the statistical method on dataset 15 showing the time taken for all operations in the method.

```
22676196 function calls (22559776 primitive calls) in 672.294 seconds
Ordered by: standard name
ncalls  tottime  percall  cumtime  percall filename:lineno(function)
 8    0.000    0.000    0.000    0.000 <__array_function__
     internals>:177(amax)
 9    0.000    0.000    0.000    0.000 <__array_function__
     internals>:177(amin)
 8    0.000    0.000    0.000    0.000 <__array_function__
     internals>:177(any)
 12    0.000    0.000    0.000    0.000 <__array_function__
    internals>:177(append)
 1    0.000    0.000    0.001    0.001 <__array_function__
     internals>:177(argsort)
 8    0.000    0.000    0.006    0.001 <__array_function__
     internals>:177(array_equal)
 1    0.000    0.000    0.000    0.000 <__array_function__
     internals>:177(array_split)
 9    0.000    0.000    0.000    0.000 <__array_function__
     internals>:177(atleast_1d)
 7    0.000    0.000    0.000    0.000 <__array_function__
     internals>:177(bincount)
 17    0.000    0.000    0.002    0.000 <__array_function__
    internals>:177(can_cast)
 57363    0.101    0.000    2.375    0.000 <__array_function__
 internals>:177(clip)
 69    0.000    0.000    0.002    0.000 <__array_function__
    internals>:177(concatenate)
 192    0.000    0.000    0.007    0.000 <__array_function__
   internals>:177(copyto)
 6    0.000    0.000    0.001    0.000 <__array_function__
     internals>:177(cumsum)
195824    0.212    0.000    1.849    0.000 <__array_function__
internals>:177(dot)
 56    0.000    0.000    0.000    0.000 <__array_function__
    internals>:177(empty_like)
 4    0.000    0.000    0.000    0.000 <__array_function__
     internals>:177(hstack)
 4    0.000    0.000    0.001    0.000 <__array_function__
     internals>:177(in1d)
 11    0.000    0.000    0.001    0.000 <__array_function__
    internals>:177(insert)
 2    0.000    0.000    0.001    0.000 <__array_function__
     internals>:177(may_share_memory)
195824    0.319    0.000   10.588    0.000 <__array_function__
internals>:177(mean)
 11    0.000    0.000    0.000    0.000 <__array_function__
 internals>:177(moveaxis)
114726    0.105    0.000    0.328    0.000 <__array_function__
internals>:177(ndim)
 ...
```

Figure 8.25: Output of the cProfile method for the machine learning method on dataset 15 showing the time taken for all operations in the method.

Acknowledgments

Thanks to my supervisors, Johan Wahlström and David Reynolds, and Dr. George De Ath for all of their support.