Veridian Internship Report: Automation of Article Segmentation in Newspapers

Summer 2017-2018

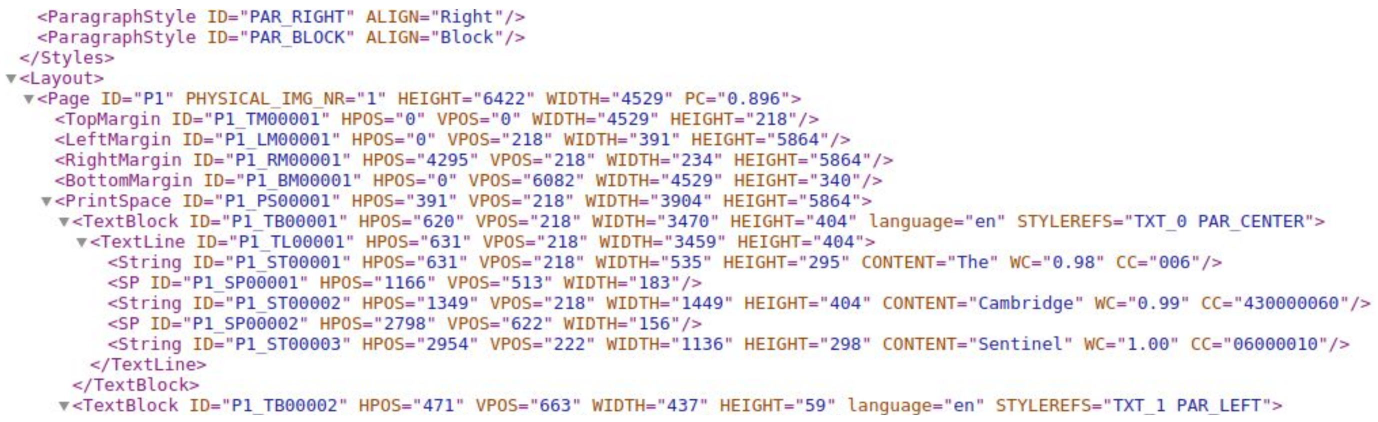
Nicole Chan
nc83@students.waikato.ac.nz

Acknowledgements:
Many thanks to the various people who have given me their time, advice, and support me throughout this project. Special thanks goes to my supervisor, **David Bainbridge**. To **Jeffrey Ke** and **Stefan Bodie** at Veridian Software, and **Alvin Yeo** for setting the wheels into motion. Thanks also goes to the The Lunchtime Club for keeping me sane.

Introduction:

To the human eye, identifying sections of a newspaper is an easy task that we do almost automatically. We learn to associate these sections, for instance, an article heading, and its corresponding article content, through visual clues. Perhaps through the breaks between columns and paragraphs, the position of the text, the size of the text, a change in topic or a combination of all these things and more. This task to identify sections correctly though quick and easy for humans takes only a few seconds across a page, but across a newspaper or millions of pages, it is taxing of time and energy to do. Hence, the purpose of my project was to explore the automation of this process through machine learning.

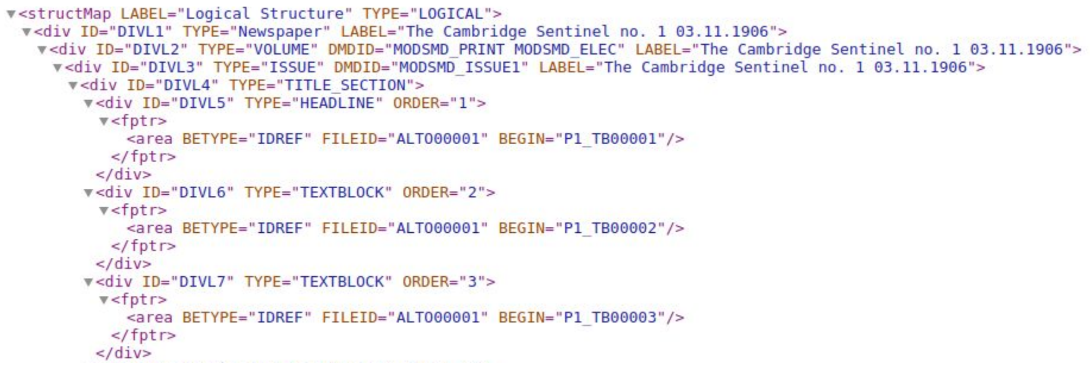
The corpus of data given by Veridian Software contained their specialised ALTO format of digitised newspaper pages, and the corresponding METS format of the page. The ALTO .xml files contained a single page of data, where the raw values of the page were contained as they were read in by an Optical Character Recognition (OCR). This was a structured format in that, loosely, a page was composed of nested structures of Page>Print Space>Text Block>Text Line>String. Where within a page and print space, many Text Blocks could occur, and within those, many Text Lines, and multiple Strings within the Text Lines.



```
    <ParagraphStyle ID="PAR_RIGHT" ALIGN="Right"/>
    <ParagraphStyle ID="PAR_BLOCK" ALIGN="Block"/>
  </Styles>
▼<Layout>
 ▼<Page ID="P1" PHYSICAL_IMG_NR="1" HEIGHT="6422" WIDTH="4529" PC="0.896">
     <TopMargin ID="P1_TM00001" HPOS="0" VPOS="0" WIDTH="4529" HEIGHT="218"/>
     <LeftMargin ID="P1_LM00001" HPOS="0" VPOS="218" WIDTH="391" HEIGHT="5864"/>
     <RightMargin ID="P1_RM00001" HPOS="4295" VPOS="218" WIDTH="234" HEIGHT="5864"/>
     <BottomMargin ID="P1_BM00001" HPOS="0" VPOS="6082" WIDTH="4529" HEIGHT="340"/>
   ▼<PrintSpace ID="P1_PS00001" HPOS="391" VPOS="218" WIDTH="3904" HEIGHT="5864">
     ▼<TextBlock ID="P1_TB00001" HPOS="620" VPOS="218" WIDTH="3470" HEIGHT="404" language="en" STYLEREFS="TXT_0 PAR_CENTER">
       ▼<TextLine ID="P1_TL00001" HPOS="631" VPOS="218" WIDTH="3459" HEIGHT="404">
           <String ID="P1_ST00001" HPOS="631" VPOS="218" WIDTH="535" HEIGHT="295" CONTENT="The" WC="0.98" CC="006"/>
           <SP ID="P1_SP00001" HPOS="1166" VPOS="513" WIDTH="183"/>
           <String ID="P1_ST00002" HPOS="1349" VPOS="218" WIDTH="1449" HEIGHT="404" CONTENT="Cambridge" WC="0.99" CC="430000060"/>
           <SP ID="P1_SP00002" HPOS="2798" VPOS="622" WIDTH="156"/>
           <String ID="P1_ST00003" HPOS="2954" VPOS="222" WIDTH="1136" HEIGHT="298" CONTENT="Sentinel" WC="1.00" CC="06000010"/>
         </TextLine>
       </TextBlock>
     ▼<TextBlock ID="P1_TB00002" HPOS="471" VPOS="663" WIDTH="437" HEIGHT="59" language="en" STYLEREFS="TXT_1 PAR_LEFT">
```

Image 1: An example of Sentinel_19061103_0001.xml in its ALTO format.

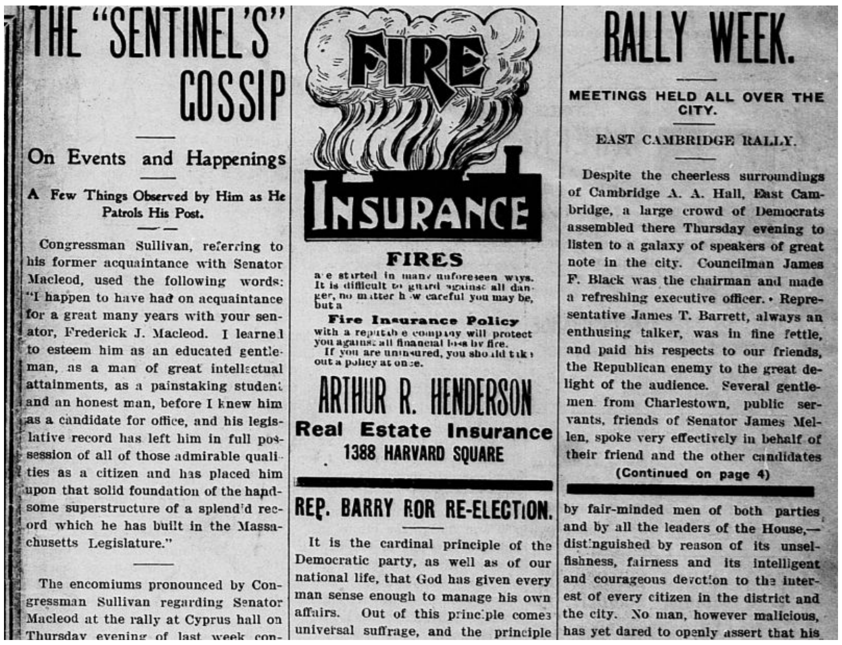
The METS .xml files contained similar nested structures, however there was only one file per newspaper issue. The METS file contained 5 sections, detailing the structure of the newspaper and the relevant files it contained data on (i.e. the pages for the issue). The fifth section was the Logical structure of the newspaper, where, within nested tags, each valid Text Block was listed. For example, if at the end of a “branch” of text blocks, the containing div types were TITLE_SECTION, then HEADLINE, the resulting tag of the text block would be TITLE_SECTION_HEADLINE.



```
▾<structMap LABEL="Logical Structure" TYPE="LOGICAL">
 ▾<div ID="DIVL1" TYPE="Newspaper" LABEL="The Cambridge Sentinel no. 1 03.11.1906">
  ▾<div ID="DIVL2" TYPE="VOLUME" DMDID="MODSMD_PRINT MODSMD_ELEC" LABEL="The Cambridge Sentinel no. 1 03.11.1906">
   ▾<div ID="DIVL3" TYPE="ISSUE" DMDID="MODSMD_ISSUE1" LABEL="The Cambridge Sentinel no. 1 03.11.1906">
    ▾<div ID="DIVL4" TYPE="TITLE_SECTION">
     ▾<div ID="DIVL5" TYPE="HEADLINE" ORDER="1">
      ▾<fptr>
         <area BETYPE="IDREF" FILEID="ALTO00001" BEGIN="P1_TB00001"/>
       </fptr>
      </div>
     ▾<div ID="DIVL6" TYPE="TEXTBLOCK" ORDER="2">
      ▾<fptr>
         <area BETYPE="IDREF" FILEID="ALTO00001" BEGIN="P1_TB00002"/>
       </fptr>
      </div>
     ▾<div ID="DIVL7" TYPE="TEXTBLOCK" ORDER="3">
      ▾<fptr>
         <area BETYPE="IDREF" FILEID="ALTO00001" BEGIN="P1_TB00003"/>
       </fptr>
      </div>
```

Image 2: An example of Sentinel_19061103_METS.xml in its METS format.

The initial approach to this problem was through a Natural Language Processing lens. Considering potential applications of sentence and text segmentation, semantic annotators, passage retrieval for the given corpus. However, it soon became clear that to correctly identify which sections of text belonged to which article was largely evident through visual formatting clues. As opposed to reading the text to figure out which sections are separate through topic shifts and semantics. For example, adjacent articles where one is on may be identified through topic shifts, however a gossip column would have multiple topic shifts, but be part of the same “article”. Though formatting clues may be different across different newspapers publishers, and even within the same publication, there are general similarities and indicators of these sections which hold true.



THE “SENTINEL’S” GOSSIP

On Events and Happenings

A Few Things Observed by Him as He Patrols His Post.

Congressman Sullivan, referring to his former acquaintance with Senator Macleod, used the following words: “I happen to have had on acquaintance for a great many years with your senator, Frederick J. Macleod. I learned to esteem him as an educated gentleman, as a man of great intellectual attainments, as a painstaking student and an honest man, before I knew him as a candidate for office, and his legislative record has left him in full possession of all of those admirable qualities as a citizen and has placed him upon that solid foundation of the handsome superstructure of a splendid record which he has built in the Massachusetts Legislature.”

The encomiums pronounced by Congressman Sullivan regarding Senator Macleod at the rally at Cyprus hall on Thursday evening of last week con-

FIRE INSURANCE

FIRES

are started in many unforeseen ways. It is difficult to guard against all danger, no matter how careful you may be, but a

Fire Insurance Policy

with a reputable company will protect you against all financial loss by fire.

If you are uninsured, you should take out a policy at once.

ARTHUR R. HENDERSON

Real Estate Insurance

1388 HARVARD SQUARE

REP. BARRY ROR RE-ELECTION.

It is the cardinal principle of the Democratic party, as well as of our national life, that God has given every man sense enough to manage his own affairs. Out of this principle comes universal suffrage, and the principle

RALLY WEEK.

MEETINGS HELD ALL OVER THE CITY.

EAST CAMBRIDGE RALLY.

Despite the cheerless surroundings of Cambridge A. A. Hall, East Cambridge, a large crowd of Democrats assembled there Thursday evening to listen to a galaxy of speakers of great note in the city. Councilman James F. Black was the chairman and made a refreshing executive officer. Representative James T. Barrett, always an enthusing talker, was in fine fettle, and paid his respects to our friends, the Republican enemy to the great delight of the audience. Several gentlemen from Charlestown, public servants, friends of Senator James Mellen, spoke very effectively in behalf of their friend and the other candidates

(Continued on page 4)

by fair-minded men of both parties and by all the leaders of the House,—distinguished by reason of its unselfishness, fairness and its intelligent and courageous devotion to the interest of every citizen in the district and the city. No man, however malicious, has yet dared to openly assert that his

Image 3: An example of Sentinel_19061103_0001 in its image format. <The “Sentinel’s” Gossip> is depicts a topic shift within the same article.

Thus, I decided to use text formatting as my parameter values. I explored other forms of Machine Learning and settled on using WEKA as my tool. This meant that I could send in a readable format of the newspaper, choose an algorithm, and then easily access results with the accuracy, and various levels of detail in results that would be automatically generated.

Method:

The first phase of the project was to design a test bed where, given an input and an algorithm, the results (accuracy and time) could be output and then compared.

I analysed the newspaper articles and came up with 8 different parameters I would use to inform a decision on what sort of section of newspaper I was identifying. The parameters generated for each line were: horizontal position, vertical position, width, height, space above the current line to the previous line, number of strings in the line, number of characters in the line, the text block number. Lastly, based on processing the METS file prior to the processing of the newspaper pages, the associated text block tag was also included as a parameter to compare against.

- E.g. The first two lines of Sentinel_19061103_0001.xml in raw form are below:

HPOS	**VPOS**	**width**	**height**	**space**	**strings**	**chars**	**textBlockNum**	**tag**
631	218	3459	404	404	3	20	1	TITLE_SECTION_HEADLINE
472	665	432	57	100	4	9	2	TITLE_SECTION_TEXTBLOCK

Additionally, due to the nature of the project – generalising the input data to accommodate for different formatting due to different newspaper publishers, the same 9 values were generated, but were re-calculated to be “normalised”. In that the horizontal position of the line is divided by the width mode for the newspaper, the vertical position is divided by the height of the text block it is contained within, the width is divided by the width mode, the height is divided by the height mode, the space is calculated and capped to a maximum and minimum of 10 and -10, the number of strings is divided by the string mode, and the number of characters is divided by the character mode.

- E.g. The first two lines of Sentinel_19061103_0001.xml in normalised form are below:

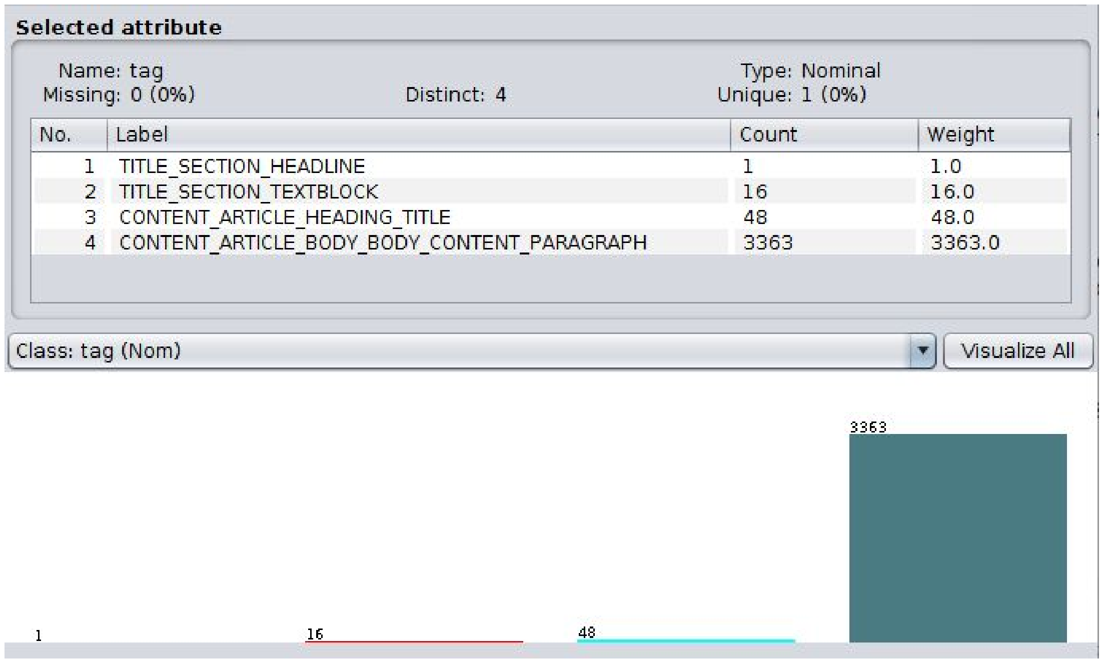
HPOS	**VPOS**	**width**	**height**	**space**	**strings**	**chars**	**textBlockNum**	**tag**
1.01	0.54	5.55	13.03	9.4	0.43	0.67	1	TITLE_SECTION_HEADLINE
0.76	11.27	0.69	1.84	2.33	0.57	0.3	2	TITLE_SECTION_TEXTBLOCK

The Visualiser.java program reads in a directory, or several directories, and generates the relevant .csv files with the 9 values per line (8 parameters, and the tag). That is, it generates a raw values page per issue of newspapers it finds, then generates the normalised values per issue, and then a file with all the concatenation of all the files given as the input.

WEKA

Due to the data being unbalanced, the SMOTE and randomize filters were applied to the training data, but not the test data for each iteration.

- **Unbalanced data:** there are many instances/lines where the tag "BODY_PARAGRAPH" has been applied, but only a few where "TITLE_SECTION_HEADLINE" has been applied, thus there are more examples of a BODY_PARAGRAPH which then skews the prediction and makes it more likely to be predicted). This meant there would likely be overfitting of the training model specific tags which would not produce reliable or more accurate results.



Selected attribute

Name: tag Type: Nominal
Missing: 0 (0%) Distinct: 4 Unique: 1 (0%)

No.	Label	Count	Weight
1	TITLE_SECTION_HEADLINE	1	1.0
2	TITLE_SECTION_TEXTBLOCK	16	16.0
3	CONTENT_ARTICLE_HEADING_TITLE	48	48.0
4	CONTENT_ARTICLE_BODY_BODY_CONTENT_PARAGRAPH	3363	3363.0

Class: tag (Nom)

Image 4: In Sentinel_19061103_0001.xml, the data is unbalanced with a 1:16:48:3363 ratio between the four tags found.

- The **SMOTE** filter allows different weights to be put on different tags, thus a higher weight on the tags where there are only a few instances would mitigate some of that imbalance.

Image 5: In Sentinel_19061103_0001.xml, a Class 3 filter of weight 100.0 was applied, which increased the weight of Class 3 - CONTENT_ARTICLE_HEADING_TITLE.

- The **randomize** filter is then applied so that the order the training data is used is shuffled. This is because when the SMOTE filter is applied, sometimes the data is grouped together so the instances of the tags will be together for some tests of the data, which will cause overfitting when the cross-validation is applied.

The process to finding a good algorithm involved guess and checking and reading about the various algorithms available, to find ones that were suitable for the dataset. Using the Auto-WEKA tool (Kotthoff, Thornton & Hutter, 2017), I found an algorithm that produced the highest accuracy with the dataset I was given, was the RandomForest, where the specific hyperparameters differed depending on the dataset, but were a good starting point.

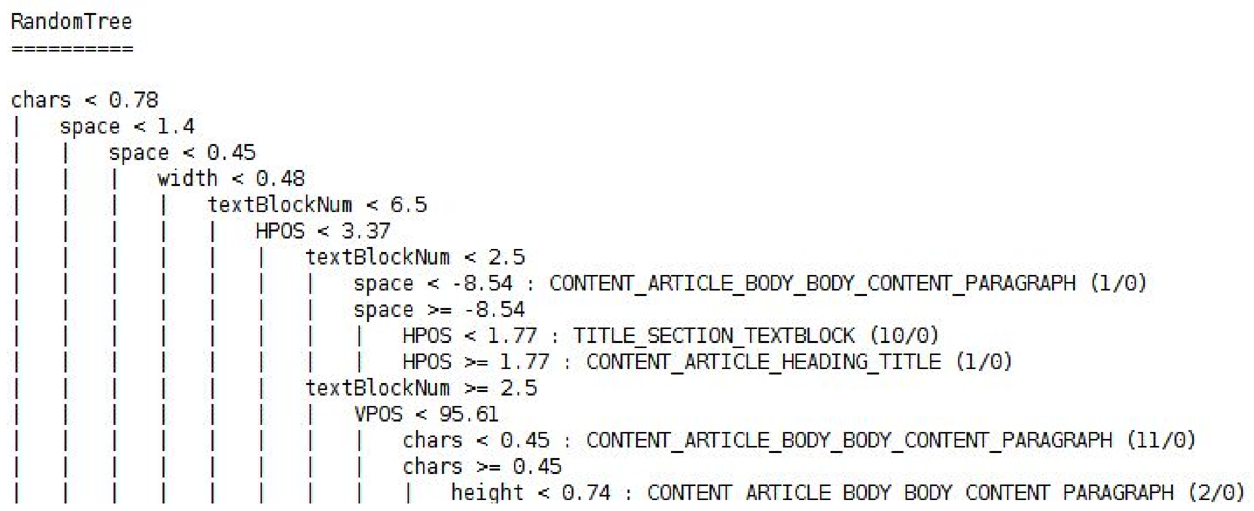
- Initially, **10-fold cross validation** was used to test the data. This is where WEKA takes the classified data i.e. every line of data represents a line of text with its various attributes, and the last column has its "tag" which is taken from the METS file.
- It then "hides" 10% of the data (due to it being a 10-fold cross-validation)
    - For the hidden 10%: it will have access to all the attributes of the lines, but not the tag column
    - For the other 90%: it will have access to all the attributes of the lines, including the tag column
- It applies the filters to the other 90%, then creates a tree with how it got to its classification, and predicts the remaining 10%.
- This process is repeated 10 times with a different section each time, hence why the randomization filter was applied prior to testing.

Results*:*

In the next section, the results are from 5 issues from five different newspapers, Cambridge Sentinel from the Cambridge Public Library, and Daily Kent Stater, The Kent Stater, The Summer Kent Stater, and The Searchlight from Kent State University. The years and issues were chosen at random, within the corpus of data provided.

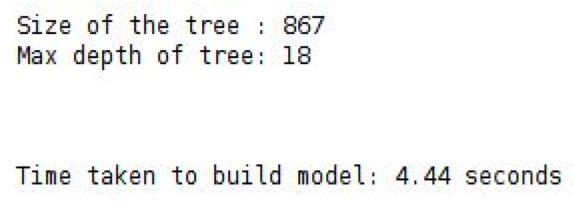
The scheme applied to the data was a 10-fold cross validation of a Filtered Classifier:
- Filters applied in order:
    - SMOTE Filter: class 1, weight 100
    - Randomize Filter
- Classifier:
    - Random Forest: depth 18, iterations 53

The first item WEKA outputs is the tree generated for the learning model, then the size, depth, and the time taken to build.



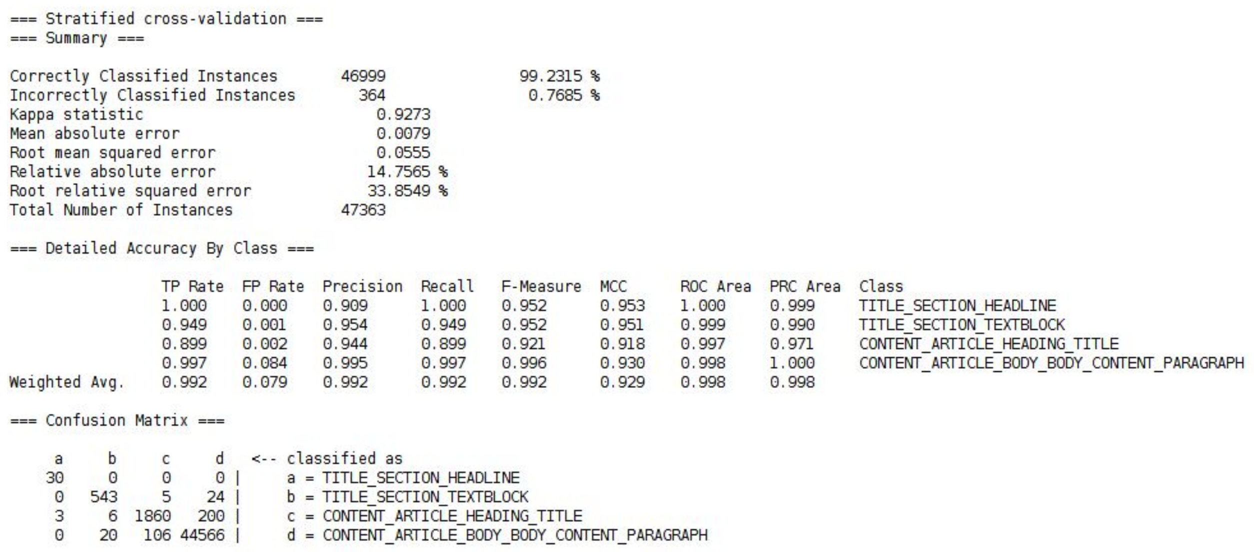
```
RandomTree
==========

chars < 0.78
|   space < 1.4
|   |   space < 0.45
|   |   |   width < 0.48
|   |   |   |   textBlockNum < 6.5
|   |   |   |   |   HPOS < 3.37
|   |   |   |   |   |   textBlockNum < 2.5
|   |   |   |   |   |   |   space < -8.54 : CONTENT_ARTICLE_BODY_BODY_CONTENT_PARAGRAPH (1/0)
|   |   |   |   |   |   |   space >= -8.54
|   |   |   |   |   |   |   |   HPOS < 1.77 : TITLE_SECTION_TEXTBLOCK (10/0)
|   |   |   |   |   |   |   |   HPOS >= 1.77 : CONTENT_ARTICLE_HEADING_TITLE (1/0)
|   |   |   |   |   |   textBlockNum >= 2.5
|   |   |   |   |   |   |   VPOS < 95.61
|   |   |   |   |   |   |   |   chars < 0.45 : CONTENT_ARTICLE_BODY_BODY_CONTENT_PARAGRAPH (11/0)
|   |   |   |   |   |   |   |   chars >= 0.45
|   |   |   |   |   |   |   |   |   height < 0.74 : CONTENT ARTICLE BODY BODY CONTENT PARAGRAPH (2/0)
```

Image 6: The first few lines of the Random Tree generated from test.csv.



```
Size of the tree : 867
Max depth of tree: 18


Time taken to build model: 4.44 seconds
```

Image 7: The size of the tree generated, the max depth of the tree, and the time taken to build the model for the Random Tree generated from test.csv.

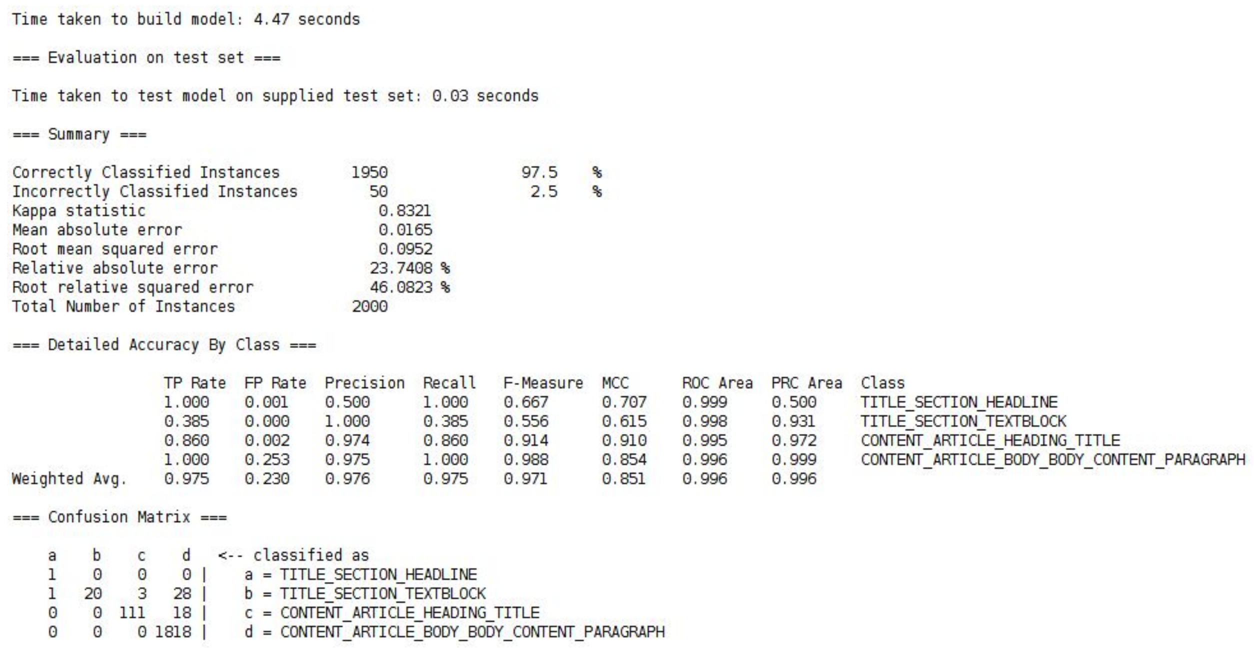


```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances       46999               99.2315 %
Incorrectly Classified Instances       364                0.7685 %
Kappa statistic                          0.9273
Mean absolute error                      0.0079
Root mean squared error                  0.0555
Relative absolute error                 14.7565 %
Root relative squared error             33.8549 %
Total Number of Instances            47363

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    0.909      1.000    0.952      0.953    1.000     0.999     TITLE_SECTION_HEADLINE
                 0.949    0.001    0.954      0.949    0.952      0.951    0.999     0.990     TITLE_SECTION_TEXTBLOCK
                 0.899    0.002    0.944      0.899    0.921      0.918    0.997     0.971     CONTENT_ARTICLE_HEADING_TITLE
                 0.997    0.084    0.995      0.997    0.996      0.930    0.998     1.000     CONTENT_ARTICLE_BODY_BODY_CONTENT_PARAGRAPH
Weighted Avg.    0.992    0.079    0.992      0.992    0.992      0.929    0.998     0.998

=== Confusion Matrix ===

     a     b     c     d   <-- classified as
    30     0     0     0 |     a = TITLE_SECTION_HEADLINE
     0   543     5    24 |     b = TITLE_SECTION_TEXTBLOCK
     3     6  1860   200 |     c = CONTENT_ARTICLE_HEADING_TITLE
     0    20   106 44566 |     d = CONTENT_ARTICLE_BODY_BODY_CONTENT_PARAGRAPH
```

Image 8: The results of the 10-fold cross validation for the Random Tree generated from test.csv.

In Image 8 with the results of the 10-fold cross validation, it is evident that the classifier with the specific hyperparameters performs at 99.2315%, correctly identifying 46999 out of 47363 instances. While 364 incorrect instances seems quite high, given the number of correctly identified instances, this seems reasonable.

In the Confusion matrix, it is evident that many of the incorrectly applied tags for a CONTENT_ARTICLE_BODY_BODY_CONTENT_PARAGRAPH was as a CONTENT_ARTICLE_HEADING_TITLE. For both TITLE_SECTION_TEXTBLOCK, and CONTENT_ARTICLE_HEADING_TITLE, the most common tag applied to incorrectly classified instances was the PARAGRAPH tag. This is likely due to the larger number of instances of the PARAGRAPHS, hence increasing the likelihood of this tag being applied to every single line, regardless of whether it is or not.



```
Time taken to build model: 4.47 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.03 seconds

=== Summary ===

Correctly Classified Instances        1950               97.5    %
Incorrectly Classified Instances        50                2.5    %
Kappa statistic                          0.8321
Mean absolute error                      0.0165
Root mean squared error                  0.0952
Relative absolute error                 23.7408 %
Root relative squared error             46.0823 %
Total Number of Instances             2000

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.001    0.500      1.000    0.667      0.707    0.999     0.500     TITLE_SECTION_HEADLINE
                 0.385    0.000    1.000      0.385    0.556      0.615    0.998     0.931     TITLE_SECTION_TEXTBLOCK
                 0.860    0.002    0.974      0.860    0.914      0.910    0.995     0.972     CONTENT_ARTICLE_HEADING_TITLE
                 1.000    0.253    0.975      1.000    0.988      0.854    0.996     0.999     CONTENT_ARTICLE_BODY_BODY_CONTENT_PARAGRAPH
Weighted Avg.    0.975    0.230    0.976      0.975    0.971      0.851    0.996     0.996

=== Confusion Matrix ===

    a    b    c    d   <-- classified as
    1    0    0    0 |    a = TITLE_SECTION_HEADLINE
    1   20    3   28 |    b = TITLE_SECTION_TEXTBLOCK
    0    0  111   18 |    c = CONTENT_ARTICLE_HEADING_TITLE
    0    0    0 1818 |    d = CONTENT_ARTICLE_BODY_BODY_CONTENT_PARAGRAPH
```

Image 9: The results of the 10-fold cross validation for the test set - (The Kent State) tks-19300821-N.csv, using test.csv as the training set.

As expected, the accuracy dropped when using a different file that was not originally in the input data, however, the accuracy is still quite high, at 97.5%. The problem of the false positive results where the majority of incorrect tags were tagged as “PARAGRAPH” is still evident in this new file.

Discussion:

Due to the small range of input data, both in years, newspapers, and publishers, the likelihood of overfitting to the current corpus is higher. Not only this, but the model generated would handle different formats to a higher degree of accuracy if there was more varied data.

In future study of this topic, more options for comparing data using different algorithms and tools could also be implemented. Presuming the comparison criteria would change depending on the algorithms used, and exploring why they would work better on the data would be interesting. While WEKA is a good tool for finding an implementation of a Machine Learning algorithm that works, the specifics are perhaps more easily tuned with a different method of implementation.

Additionally, the problem of overfitting, while somewhat mitigated in the implementations used in this project with SMOTE and Randomize filters, it is likely there is still a large amount of overfitting. This is natural given the dataset provided, and how newspapers are, but further exploration into how this problem could be reduced would be recommended.

References:

Kotthoff, L., Thornton, C., & Hutter, F. (2017). *Auto-WEKA*. *Cs.ubc.ca*. Retrieved 11 February 2018, from https://www.cs.ubc.ca/labs/beta/Projects/autoweka/#software