The Extremes of Good and Evil

Bachelor Thesis

presented by
Larissa Strauch
Matriculation Number 1518629

submitted to the
Data and Web Science Group
Prof. Dr. Ponzetto
University of Mannheim

August 2014

Contents

List of Algorithms

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem Statement

1.2 Contribution

1.3 Related Work

Chapter 2

Data

2.1 Data Description

The given data, on which we want to build our model on, was provided by zenodo.org as part of SemEvals Task 4 and consists of 2 independent datasets, which in turn have been divided into a GroundTruth-, Training- and Validation set.
The first dataset, recognizable by the term 'byPublisher', reflects the publisher's general bias set forth by BuzzFeed journalists or MediaBiasFastCheck.com beforehand. It consists of a total of 750,000 items, of which 600,000 belong to the Training- and 150,000 to the Validation set.
In return, the second dataset, recognizable by the term 'byArticle', was scrapped by crowdsourcing at hand and therefore consists of only 645 items without a Validation set.

The GroundTruth dataset was provided as an XML File and consists of the features 'article url', 'labeled-by', 'id' and 'hyperpartisan'. In addition, the GroundTruth dataset scrapped 'byPublisher' contains the feature 'bias'.

- Article-url: Contains the article's url.
- Labeled-by: Reflects whether the respective article was labeled 'byPublisher' or 'byArticle'.
- Id: Allocates each article a unique id.
- Hyperpartisan: Displays whether the particular article has been labeled as hyperpartisan or not.
- Bias: Divides the publisher's bias into 'left', 'left-center', 'least', 'right-center' and 'right'.

The Training dataset was as well provided as an XML file and contains the contents of the website of the respective article. In addition, it consists of the features 'article title' 'published-at' and 'id'.

- Article title: Represents the articles title.
- Published-at: Specifies the published date.
- Id: A unique id, which is the same as the corresponding entry in the GroundTruth dataset.

The given Data has been cleaned in advance, therefore no additional steps were necessary.
The main focus of the datasets is on the Hyperpartisan feature, on which we want to classify the articles as this thesis progresses.

2.1.1 Dataset labelled by Publisher

As mentioned above, this dataset consists of a total of 750,000 articles and is divided into a training record consisting of 600,000 articles and a validation set consisting of 150,000 articles. Summarizing these two sets of data, a total of 375,000 were labelled as 'Hyperpartisan' and 375,000 were not – which corresponds to a 50:50 distribution. But even individually, this distribution does not change.

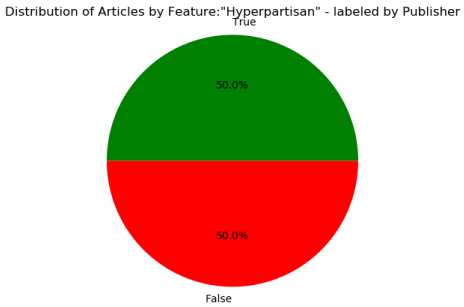
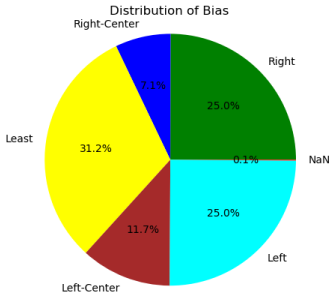


Figure 2.1: Distribution of as Hyperpartisan labelled articles by publisher

This dataset also includes the feature 'bias', which informs you about the general bias of the publisher. All 375,000 Hyperpartisan labelled all are assigned to either the left or right sectors, but none are right-centre, least or left-centre and are again 50:50 distributed.

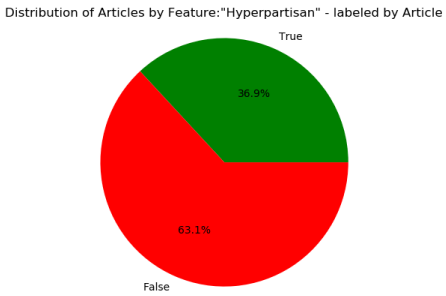
The other 50% are split between the remaining bias, with 'Least' owning the largest share at 37%.
The publicity data is distributed over the years 1964-2018, with most of the data coming from 2012-2018.



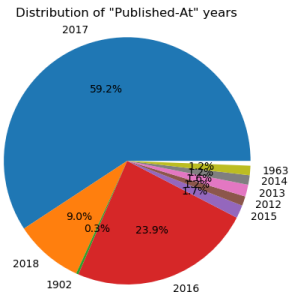
(a) Distribution of Bias

2.1.2 Dataset labelled by Article

The dataset labelled by Article is a little different to the larger one labelled by publisher. Here the articles were individually labelled by hand. Accordingly, the distributions of this dataset are completely different. This becomes quiet obvious if we look at how the distribution of the Hyperpartisan labelled articles is here. Here



(b) Distribution of Hyperpartisan labelled articles



(c) Distribution of publishing years

we can see that there is no 50:50 distribution left. Only 36.9% were defined here as Hyperpartisan as shown in figure 2.4.
Moreover, in this dataset, the distribution of publication data is not mainly from the

years 2012-2018, but in 2016-2018, with the largest number of articles dating back to 2017 at just under 60% as we can see in figure 2.5. Altogether all 645 articles date from the years 1902-2018.

2.2 Data Analysis

Hyperpartisan means ”extremely partisan; extremely biased in favor of a political party.” [definition]. This often materializes in relation to significant political events. As a result, in the following chapter, I will discuss in detail the direct link between the various features, especially the correlations between publication dates and label, as well as the connections between publisher and label.

The two figures e and f list the publisher who produced the highest amount of hyperpartisan articles in the respective dataset. To keep track, only those publishers who have publisher more than 7 hyperpartisan articles in the dataset 'labelled by article' and who have published more than 10.000 hyperpartisan articles in the dataset 'labelled by publisher' are included in the tables. However, a problem here is the dataset 'Labelled by Publisher', since this record, as previously described, has been labelled by the overall bias of the publisher. Meaning, for the further development, we can not include the publishers, since each article of a publishing house has the same label. What I would like to discuss later, however, is the connection between whether or not one of the publishers listed in Table x published more articles in important political years.

2.3 Data Preparation

In order to be able to work with the existing data in the further course of this project, several preprocessing steps are necessary. In the preprocessation phase of my bachelor's thesis, the data therefore went through the following steps:

1. Read the XML files.
2. Filter out the important information.
3. Merge the Groundtruth- and Training datasets into a csv file.
4. Remove special characters and stop words.
5. Tokenization and stemming of the datasets.

As a result, in the following section, I will go further into detail how I preprocessed my data.

2.3.1 Read the XML files

Since it is difficult to work with the given data in an XML format, it is necessary to bring them into a format with which it'll be easier to work with. The particular challenge hereby is the size of the dataset labelled by publisher. A standard algorithm for reading XML files is provided by pythons DOM library ElementTree, called "ElementTree.parse". This method returns an ElemenTree type, which "is a flexible container object, designed to store hierarchical data structures in memory" [http://effbot.org/zone/element.htm]. Meaning, that this library forms the entire model in the memory which can pose a problem with very large files, such as ours. As a substitute,I therefore use the method "ElemenTree.iterparse", which can process XML documents in a streaming fashion, retaining in memory only the most recently examined parts of the tree.

2.3.2 Filter out the important information

As mentioned in Chapter 2.1 Data Description, the XML files include varoius fetatures, which will play an important role in the further course of this work. It is now necessary to filter these features out of the XML format so the can be better worked with later. In order to be able to do this, we pass an algorithm the, by the iterparse method read content, which then passes thorugh this algorithm in a double for-loop and checks for each item of an element in the content which "key" is currently processed. I then save this in an array, so that the features which have already been parsed, can be used later.

```
def parse_features(content, publisher):
  for event, elem in content:
    for key, value, in elem.items():
      if publisher:
        if key == 'id':
          id_array_publisher.append(str(value))
        elif key == 'published-at':
          published_at_array_publisher.append(value)
        elif key == 'title':
          title_array_publisher.append(value)
      else:
        if key == 'id':
          id_array_article.append(str(value))
        elif key == 'published-at':
          published_at_array_article.append(value)
        elif key == 'title':
          title_array_article.append(value)
      elem.clear()
```

2.3.3 Merge the Groundtruth- and Training datasets into a csv file

Since both, the Groundtruth- and Trainingdata contain important information, it is necessary to merge them into one file. I decided to use Python's library pandas to write the two files into one csv file. This, and espacially Pandas, allows us to read the file more quickly as well as to access individual rows and columns of the merged file in a targeted manner.

```
feature_extraction.parse_features(xml_training, publisher)
groundtruth_parser.parse_groundtruth(xml_gt, publisher)
content = content_parser.parse_content(content_training)

a_id = feature_extraction.get_id_array(publisher)
published = feature_extraction.get_published_at_array(publisher)
title = feature_extraction.get_title_array(publisher)
bias = groundtruth_parser.get_bias_array(publisher)
hyperpartisan = groundtruth_parser.get_hyperpartisan_array(publisher)

columns = {"ArticleID": a_id,
           "PublishedAt": published,
           "Title": title,
           "Bias": bias,
           "Content": content,
           "Hyperpartisan": hyperpartisan}

tf = Pd.DataFrame(columns)
tf = tf[['ArticleID','Published', Title','Bias','Content','Hyperpartisan'
    ]]
tf.to_csv(titlecsv,encoding='utf-8',index=False)
```

2.3.4 Remove special characters and stop words

Now that we have merged both files and written them into a csv file, we now need to remove special characters and stop words. Especially the removal of stopwords is necessary to shrink our corpus and due to the fact that words such as "the," "a," etc. do not influence the classification of our content in the further course.

```
stop = stopwords.words('english')

df['Content'] = df['Content'].apply(lambda x: ''.join([item for item in x.
    split() if item not in stop]))
df['Title'] = df['Title'].apply(lambda x: ''.join([item for item in x.
    split() if item not in stop]))
df['Content'] = df['Content'].map(lambda x: re.sub(r"[^a-zA-Z0-9]+", '', x
    ))
df['Title'] = df['Title'].map(lambda x: re.sub(r"[^a-zA-Z0-9]+", '', x))
```

Here it becomes obvious that using pandas was a good choice, as we can now specifically access the 'Content' and 'Title' columns in order to perform this step of preprocessing on only these two and not the whole file.

2.3.5 Tokenization and Stemming of the datasets

After cleaning the dataset, the words are tokenized in order to convert them into numerical vectors so that a classifier is able to work with them. Tokenization is definied as ”The process of demarcating and possibly classifying sections of a string of input characters. The resulting tokens are then passed on to some other form of processing. The process can be considered a sub-task of parsing input”.
Stemming is the procedure of reducing the word to its grammatical (morpho-syntatctic) root. The result is not necessarily a valid word of the language. For example, ”recognized” would be converted to ”recogniz”. Still, the basic word almost always contains the very meaning of the word. Stemming is advantageous in that the algorithm used later now only has to fall back on a few different words and not many, all of which have the same meaning.

In order to implement Stemming and Tokenization, only 2 lines of python code are necessary, due to the Pandas dataframe.

```
df["Content"] = df["Content"].apply(nltk.word_tokenize)
df['Content'] = df['Content'].apply(lambda x: [stemmer.stem(y) for y in x
    ])
```

This will transform our output file in the following way:

```
ArticleID,PublishedAt,Title,Bias,Content,Hyperpartisan
0000000,2017-09-10,Kucinich: Reclaiming the money power,/,"From flickr.com: Money
0000001,2017-10-12,Trump Just Woke Up & Viciously Attacked Puerto Ricans On Twitte
0000002,2017-10-11,"Liberals wailing about gun control, but what about abortion?",
```

(d) Csv file before Stemming and Tokenization

```
ArticleID,PublishedAt,Title,Bias,Content,Hyperpartisan
0,2017-09-10,"[kucinich, reclaiming, money, power]",/,"[flickr, com, money, mid, 1
1,2017-10-12,"[trump, woke, viciously, attacked, puerto, ricans, twitter, like, cr
2,2017-10-11,"[liberals, wailing, gun, control, abortion]",/,"[photo, justin, sull
```

(e) Csv file after Stemming and Tokenization

Ehrenwörtliche Erklärung

Ich versichere, dass ich die beiliegende Master-/Bachelorarbeit ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen. Ich bin mir bewusst, dass eine falsche Er- klärung rechtliche Folgen haben wird.

Mannheim, den 31.08.2014 Unterschrift