STRAINER: STATE TRANSCRIPT RATING FOR INFORMED NEWS ENTITY RETRIEVAL

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Computer Science

by

Thomas Gerrity

June 2022

© 2022

Thomas Gerrity ALL RIGHTS RESERVED

COMMITTEE MEMBERSHIP

TITLE: STRAINER: State Transcript RAting for Informed News Entity Retrieval

AUTHOR: Thomas Gerrity

DATE SUBMITTED: June 2022

COMMITTEE CHAIR: Alexander Dekhtyar, Ph.D. Professor of Computer Science

COMMITTEE MEMBER: Maria Pantoja, Ph.D. Associate Professor of Computer Science

COMMITTEE MEMBER: Foaad Khosmood, Ph.D. Professor of Computer Science

ABSTRACT

STRAINER: State Transcript RAting for Informed News Entity Retrieval

Thomas Gerrity

Over the past two decades there has been a rapid decline in public oversight of state and local governments. From 2003 to 2014, the number of journalists assigned to cover the proceedings in state houses has declined by more than 30%. During the same time period, non-profit projects such as Digital Democracy sought to collect and store legislative bill and hearing information on behalf of the public. More recently, AI4Reporters, an offshoot of Digital Democracy, seeks to actively summarize interesting legislative data.

This thesis presents STRAINER, a parallel project with AI4Reporters, as an active data retrieval and filtering system for surfacing newsworthy legislative data. Within STRAINER we define and implement a process pipeline by which information regarding legislative bill discussion events can be collected from a variety of sources and aggregated into feature sets suitable for machine learning. Utilizing two independent labeling techniques we trained a variety of SVM and Logistic Regression models to predict the newsworthiness of bill discussions that took place in the California State Legislature during the 2017-2018 session year. We found that our models were able to correctly retrieve more than 80% of newsworthy discussions.

ACKNOWLEDGMENTS

Thanks to:

- My parents for supporting me throughout my educational career.
- Dr. Khosmood, for motivating the AI4Reporters project.
- Parts of this thesis overlaps with work published in "Automatic News Article Generation from Legislative Proceedings: A Phenom-Based Approach" [27]

TABLE OF CONTENTS

					Page
LI	ST O	F TAB	LES		Х
LI	ST O	F FIGU	URES .		xii
Cl	НАРЛ	TER			
1	Intro	oductio	n		1
2	Bacl	kground	1		5
	2.1	Califo	rnia Legis	slature	5
		2.1.1	Legislat	ive Process	5
		2.1.2	Data Re	ecording	7
	2.2	Digita	l Democr	acy	7
		2.2.1	Data St	ored by Digital Democracy	8
			2.2.1.1	Person Data	8
			2.2.1.2	Bill Data	9
			2.2.1.3	Hearing Data	10
	2.3	Machi	ne Learni	ng	11
		2.3.1	The Ma	chine Learning Process	12
			2.3.1.1	Data Acquisition/Data Cleaning	12
			2.3.1.2	Feature Engineering and Feature Selection	12
			2.3.1.3	Model Selection	13
			2.3.1.4	Model Training and Evaluation	13
			2.3.1.5	Hyperparameter Tuning	15
	2.4	Super	vised Clas	ssification Machine Learning Methods	15
		2.4.1	Support	Vector Machines	15

		2.4.2 Logistic Regression	17
	2.5	Natural Language Processing	18
		2.5.1 Named Entity Recognition	18
3	Rela	ted Work	20
	3.1	News Value	20
		3.1.1 News Values and Machine Learning	24
	3.2	AI4Reporters	24
4	Desi	gn	26
	4.1	Data Sources	28
	4.2	Data Aggregators	29
	4.3	Event Data Models	30
	4.4	Feature Extractors	32
	4.5	Event Labelers	33
	4.6	Predictors	33
	4.7	AI4Reporters Integration	34
		4.7.1 EventLabeler to Phenoms	35
		4.7.1.1 Data Source object as API for AI4Reporters	35
		4.7.1.2 Metadata Phenoms in AI4R	36
5	Imp	lementation	37
	5.1	Bill Discussion Data Sources	37
		5.1.1 Newspapers.com Data Source	37
		5.1.2 Digital Democracy Data Source	39
	5.2	Data Models for Bill Discussion Events	42
	5.3	Bill Discussion Event Data Aggregator	46
	5.4	Bill Discussion Feature Extractor	46

	5.5	Bill Di	iscussion Labelers	52
		5.5.1	Rules Based	52
		5.5.2	Article Proximal Association	56
	5.6	Bill Di	iscussion Predictors	57
	5.7	Impler	mentation within AI4Reporters	58
		5.7.1	Metadata extraction	58
		5.7.2	Newsworthiness Filter	59
		5.7.3	Headline and Intro Phenoms	59
6	Eval	uation		60
	6.1	Study	Design	60
		6.1.1	Metrics	61
		6.1.2	Variables	63
	6.2	Datase	et Inspection	63
		6.2.1	Rules Based Labeling Results	64
		6.2.2	Proximal Article Labeling	66
	6.3	Predic	tor Model Configuration and Training	68
		6.3.1	Results Analysis	70
			6.3.1.1 Articles-Based Labeling Results	70
			6.3.1.2 Rules-Based Labeling Results	74
		6.3.2	Analysis	78
7	Con	clusion	and Future Work	80
	7.1	Future	e Work	80
		7.1.1	Training On a Union of Labelers	81
		7.1.2	Improved Data Labeling	81
		7.1.3	STRAINER as a Ranking System	81

	7.1.4 Expanding STRAINER Beyond Bill Discussions		82
	7.1.5	Testing STRAINER Across Multiple Session Years	82
	7.1.6	Testing STRAINER Prediction Using Non-iid Machine Learn- ing Methods	83
BIBLIOGRAPHY 8			84
APPEN	NDICES		
А	A Model Results		92
В	Top 50) Feature Weights from Best Performing Models	100

LIST OF TABLES

Table]	Page
3.1	News Values Consideration for STRAINER	23
5.1	Excerpt From Article Labeling Initial Dataset	39
5.2	DDDatabase Clase Methods	40
5.3	Bill Discussion Data Model Class Attributes	43
5.4	Bill Data Model Class Attributes	45
5.5	Bill Version Model Class Attributes	45
5.6	Methods of Bill Discussion Feature Extractor	47
5.7	Top Level Extracted Features	48
5.8	Spacy Named Entity Categories [11]	51
5.9	BusinessRulesEventExtractor Class Methods	54
6.1	Feature Abbreviations	70
6.2	Articles-Labeled Results	71
6.3	Highest Weighted Features From the Logistic Regression Model with the Highest Recall on Articles- Based Labeled Data	72
6.4	Rules-Labeled Results	74
6.5	Highest Weighted Features From SVM Model With the Highest Recall on Rules-Based Labeled Data	75
6.6	Per-Rule Recall Abbreviations	77
6.7	Rules-Labeled Per-Rule Recall Results	78
A.1	Articles Labeled Results	92
A.2	Rules Labeled Results	94

A.3	Rules Labeled Models, Recall Results Per Rule	97
B.1	Top 50 Features by Weight for Regression Model With the Highest Recall on Articles-Based Labeled Data	100
B.2	Top 50 Features by Weight for Model With the Highest Recall on Rules-Based Labeled Data	103

LIST OF FIGURES

Figure		Page
2.1	Example of Support Vector Classification in 2-D Space	16
2.2	Example of Logistic Regression Model Fitment	17
4.1	Strainer Design Diagram	27
4.2	Strainer Data Source Abstract Class	28
4.3	Strainer Data Aggregator Abstract Class	29
4.4	Example of Data Model Nesting	31
4.5	Strainer Feature Extractor Abstract Class	32
4.6	AI4Reporters Summarization[27]	34
5.1	Relevant Article Retrieval	38
6.1	Rule-Based Labeling Results California 2017-2018 Session	64
6.2	Rule-Based Intersection Results California 2017-2018 Session	65
6.3	Articles per Discussion Labeling Results California, 2017-2018 Session	n 66
6.4	Articles per Discussion Labeling Results California, 2017-2018 Session	n 66
6.5	Rules-Based/Articles-Based Labeling Overlap California, 2017-2018 Session	67
6.6	Rules-Based/Articles-Based Labeling Overlap For Each Rule, Cali- fornia 2017-2018 Session	68

Chapter 1

INTRODUCTION

According to recent surveys the number of traditional reporters assigned to cover state houses decreased by 30+% from 2003 to 2014 [32]. Without active coverage of legislative proceedings, government becomes less transparent. This is problematic. It is this transparency which allows governments to truly claim the mantle of democracy. The extent which the populace is aware of, and participates in, government proceedings determines the level of democratic rule. The decline in local news organizations has been found to directly correlate with declines in civic involvement. When public oversight is diminished the legislative process suffers [43]. Legislators are more prone to cater to special interests when aware that their actions are less scrutinized. More problematic is the ease with which politicians can hide their true records from the voting public come election time. With fewer local news outlets actively directing the public's attention to government misdeeds, [41] claims that fewer new candidates run for political office.

The cause of the decline in journalism at the state and local level is a topic currently under investigation [32, 14]. This decrease is often attributed to the advent of the internet age. The web provides a platform for anyone with an internet connection to publish content to the world. Local news outlets reliant upon subscriptions and add revenue saw their profits dwindle from 2008 to 2018. According to the Pew Research Center newspaper revenue declined 62% during this period, triggering massive layoffs across the industry resulting in a 47% decrease in newsroom employment. In an effort to increase transparency some state and local governments have begun to make use of the internet to publish their proceedings for public view. However, this data is often presented in a raw fashion which is difficult to search. This, coupled with the sheer amount of data, makes it nigh impossible for a single citizen to manually filter out relevant items.

Driven by these and other societal issues, former California State Senator Sam Blakeslee in collaboration with California Polytechnic State University, founded the Institute for Advanced Technology and Public Policy in 2012. Under this organization, the Digital Democracy Project was initiated [5]. The goal of this project was simple. Make the proceedings within the state legislature more accessible to the public. In particular, Digital Democracy focused primarily on hearings which take place in the state legislature. In California, these hearings were video recorded and then published online. As a result, these records were neither easily searchable nor convenient to browse.

The Digital Democracy Project utilized both machine learning techniques and human up-leveling to faithfully transcribe these video recordings. These transcriptions were then stored in a database and served to the public, free of charge, via an online platform. With time, Digital Democracy's database not only contained the hearing transcriptions but also information regarding the people speaking and the bills they spoke about. This data was retrieved from a variety of online sources including both state government open databases and registries for each of the upper and lower houses, as well as websites maintained by non-profit organizations. The Digital Democracy Project was the first attempt at combining data from these sources into a single database.

The Digital Democracy Project and others like it, serve as a good first step toward shoring up the current oversight mechanisms at the state level. However, they fall short of replacing the lost value of a dedicated field reporter. Digital Democracy is a passive information system. It requires the user to initiate the search. To do this, the user must know what they are interested in *a priori*.

Current research, of which this thesis is a part, involves the creation of an active information system, based on the legislative data contained within Digital Democracy. Researchers at Cal Poly seek to build a wire service which uses the information with Digital Democracy as its source material. Oriented around discussions of bills, it would surface newsworthy hearing information. Then, having compiled this information, the system generates human-readable summaries/tipsheets which could then be marketed to news outlets or published directly. Similar automated article generation has been successfully used to generate articles describing company quarterly reports and sports events [38]. However, these systems do not determine which topics are newsworthy, they merely generate an article based on the data available.

This thesis, State Transcript RAting for Informed News Entity Retrieval (STRAINER), focuses on the design, implementation and evaluation of an automated newsworthiness rating system for legislative hearing data. This system functions as the first step in an active information system. The STRAINER process can be summarized as follows.

- Data acquisition We collect data surrounding legislative events from Digital Democracy and other data repositories
- 2. Feature extraction We format, collate, and derive features from the retrieved legislative data that we will use for machine learning.
- 3. News value prediction Utilizing multiple machine learning methods, we predict a binary classification expressing the newsworthiness of the hearing.

4. Article data retrieval and formatting - Here we extract the events of the hearing into a format required for its conversion to a news article/tipsheet via natural language generation.

The contributions of this thesis are as follows:

- A system for extracting structured data regarding legislative hearings.
- An extensible framework for feature selection and extraction for hearing data.
- A system for assigning a news rating to legislative hearing proceedings.
- A training dataset of hearing discussion data labeled regarding newsworthiness.
- An system for retrieving legislative data for article generation within AI4Reporters.
- A case study performed on California Legislative data demonstrating the performance of STRAINER.

The rest of this document is organized as follows. Chapter 2, "Background", contains background information regarding the Legislative data, how it is/was collected by Digital Democracy and a brief overview of machine learning relevant to this project. Chapter 3, "Related Work", discusses related work in the area of determining newsworthiness. Chapter 4, "Design", describes the overall structure of the STRAINER pipeline. Chapter 5, "Implementation", details the specific implementation of STRAINER within the context of predicting newsworthiness of bill discussions. Chapter 6, "Evaluation", discusses the evaluation of these predictions. Finally, chapter 7, "Conclusion and Future Work", summarizes the work achieved by STRAINER and future improvements to the system.

Chapter 2

BACKGROUND

This thesis relies on rich, nuanced datasets that have been compiled over the course of several years. In order to understand the STRAINER data pipeline, it is helpful to provide a brief overview of the source of this data, the processes used to manipulate it, and past and ongoing projects that STRAINER is designed to integrate with. This chapter gives an overview of the legislative process, legislative recording, the machine learning processes and tools used to build STRAINER, and finally, a general outline of Digital Democracy Data.

2.1 California Legislature

The state legislature is composed of forty senators and eighty assembly members [3]. These elected officials are supported by over 2000 legislative staff [49]. Both houses form committees composed of legislators from each house. There are three committee types: standing, select and special. Additionally, there are joint committees, which consist of legislators from both houses.

2.1.1 Legislative Process

Each year, approximately 2100 bills are introduced into the legislature. In 2019, over 2576 bills were introduced [20]. At the time of this thesis' writing, the legislative process within the State of California can be summarized as follows [25]:

- Bill Authoring: A legislator or group of legislators put forward an idea for a bill. The idea is converted to an initial draft of a bill by the Legislative Counsel. The legislator(s) then introduce the bill into their respective house. This introduction is a floor reading, consisting of the reading of the bill number, the names of the authors, and the bill's title. Once a bill is introduced, thirty days transpire before further action is taken.
- Hearings: In order to proceed, a bill passes through the rules committee of the chamber in which it was introduced. The rules committee then assigns the bill to relevant committees. Each of these committees can then assign the bill to a hearing. Committee staff generates a bill analysis which summarizes the bill and also enumerates the organizations supporting and opposing the bill. During the hearings, the bill's author presents the bill to the committee members. Testimony by legislators, lobbyists and members of the public can then be heard in support of or in opposition to the bill. After discussion, the committee members vote on whether the bill should proceed in its current form.
- Amendments: If legislators are not content with the current wording of a bill, they may propose amendments to a bill. These amendments are then voted and agreed upon. Amendments may be added up until the final floor vote. As a result, a singular bill may have multiple different versions as it makes its way through the legislature.
- Floor Vote: Once passed out of committee, the bill has two more floor readings. Next, the entire house votes on the bill. If passed, the bill is sent to the governor's desk for signing.
- Governor Signature/Veto: Once a bill reaches the governor's desk it may be signed by the governor, at which point the bill becomes state law. If, instead,

the governor vetos the bill, it is sent back to both houses for reconsideration and further amendment.

2.1.2 Data Recording

The bill texts, bill analysis, and votes are published and available to the public on "http://leginfo.legislature.ca.gov". Video and audio recordings are made of floor proceedings and committee hearings. These recordings are made available online officially via "http://www.calchannel.com/video-on-demand/". However, no official transcripts or minutes of these events are produced [49].

2.2 Digital Democracy

Digital Democracy is a project funded through the Institute for the Advancement of Technology and Public Policy (IATPP) [49]. Its goal is to enhance transparency within government by creating a searchable database of legislative proceedings. Fulfilling this task requires the transcription of the publicly available recordings of legislative proceedings and collection of bill texts. Digital Democracy's database is accessed via an interactive website in which one can search for particular bills, legislators, or political topics. As the project progressed, more contextual information began to be integrated, including information regarding legislators themselves, lobbyists and financial contributions to legislators. Digital Democracy also expanded its operation to other states. Currently, Digital Democracy contains legislative information for California, Texas, New York and Florida.

2.2.1 Data Stored by Digital Democracy

The data stored by Digital Democracy falls into three main categories: person data, bill data, and hearing data.

2.2.1.1 Person Data

Digital Democracy collects data on all the speakers who take part in the legislative hearings. These speakers include legislators, employees of the legislature, state government and governor's office representatives, lobbyists, and members of the general public. For legislators, Digital Democracy collects data including the following:

- The district represented
- Party affiliation
- Committee membership and position
- Terms of office

In the State of California, the law states that an individual who incurs more than \$2500 worth of lobbying activity must register as a lobbyist with the State [35]. Digital Democracy utilized this registration information to identify lobbyists speaking within a hearing. Lobbyist registration also allowed the collection of detailed information regarding lobbyist employment/affiliation with organizations.

When members of the general public speak during a hearing, it is customary for them to give some standard information. This information can include the speakers name, who they represent, where in the state they come from, and their support or opposition to the bill being discussed. Digital Democracy collects their names and any organization affiliation information available.

2.2.1.2 Bill Data

Digital Democracy collected not only information regarding the content of bills but also an historical record of the state of each bill, as it was changed and passed through the legislature. A single bill can be amended many times. For this reason, each bill can have many different versions. As a result, we have the following information available to us:

- Bill title and version texts The official name and text of each version of a bill, as it passed through the legislature.
- Bill digest A short summary of the bill version.
- Bill analyses An explanation of the bill version is usually generated by each committee before it considers/discusses the bill. In addition to an explanation/- summarization, bill analyses usually contain information regarding the organizational endorsements and objections to the bill.
- Vote History A record of each motion vote regarding the bill
- Bill Status History A record of the State of the bill (passed, shelved, etc)

2.2.1.3 Hearing Data

From the hearing videos provided by the state, along with published hearing agendas, Digital Democracy collected the following hearing information:

- Hearing transcripts A transcript of the bill discussion that took place during the hearing, separated into an ordered set of utterances. Each Utterance is tagged with the ID of the speaker and their alignment towards the bill.
- Committee information Each hearing is linked to its corresponding committee.
- Hearing date The day upon which the hearing was held.
- Hearing video information Metadata regarding the video recording of the hearing/bill discussions.
- Hearing vote information records of the vote that took place during the discussion of a bill.

Digital Democracy also stores information regarding the interactions between these data categories, as well as between organizations. Interaction data includes the following:

- Bill/Person interactions Information regarding bill authors, sponsors, personal legislator vote histories.
- Organization/Person interactions Information linking legislators and lobbyists with organizations, including gifts, behests, and employment.
- Bill/Hearing information Bill version discussed at a particular hearing.

While this is not an exhaustive list of the data held by Digital Democracy, it is a sufficient definition of the types of data available to our system.

2.3 Machine Learning

STRAINER is a machine learning pipeline. The phrase "machine learning", when used colloquially, is fraught with ambiguity [17]. This may, in part, be due to Arther Samuel, who is recognized for coining the term in the late 1950's. The following extremely broad definition of machine learning is attributed to him:

"The field of study that gives computers the ability to learn without being explicitly programmed" [40]

In recent years, within the academic/tech community, more stringent definitions have been put forward. In 2012, professor Pedro Domingos, University of Washington, defined machine learning methods as follows:

"Machine learning algorithms figure out how to perform important tasks by generalizing from examples" [13]

Four years later, in 2016, Nvidia offered yet another definition:

"The practice of using algorithms to parse data, learn from it, and then make a determination or prediction about something in the world" [8]

It is this third definition of machine learning which we will utilize to define machine learning.

2.3.1 The Machine Learning Process

Given this definition, the process of designing and implementing a machine learning algorithm follows a mundane data science pipeline/procedure which can be broken into the following stages.

2.3.1.1 Data Acquisition/Data Cleaning

All machine learning models rely on data. Data can be collected in a variety of ways, depending upon the field study. Data can come from quantitative sensors, such as accelerometers, voltmeters, temperature sensors, etc. On the other hand, the data may come from direct human input, such as reviews, recordings, surveys or tests. Regardless of the method of collection, all data needs to be preprocessed to remove spurious readings, malformed data, or duplicate entries. In addition, data may need to be placed on the same scale/unit of measurement. Human-recorded data may need to be filtered and manipulated further. For example, misspellings may need to be fuzzy-matched with the correct words. This data cleaning process requires the attention of a domain expert to properly determine the optimal level of cleanliness.

2.3.1.2 Feature Engineering and Feature Selection

Once the data has been cleaned, the next step of the ML process is feature engineering. At this stage, the data scientist may perform manipulations on the raw data to transform it to a more usable form and extract new features. For quantitative data, this may include binning/aggregating, combining features arithmetically, or creating transformed versions of an original feature. Once features have been generated feature selection is performed. This is a necessary step, due to the fact that utilizing a subset of features often results in a more accurate model than one trained with all possible features. Depending on the model chosen, feature selection would be performed utilizing some form of feature value estimation, such as Lasso [36].

2.3.1.3 Model Selection

Feature engineering is followed by model selection. Conventional machine learning models can be divided into two classes based on the type of prediction required. These classes are regression models and classification models. Regression models are used to predict quantitative values, whereas classification models are used to predict categorical values. Examples of regression models include linear regression [48], regression trees [4], and support vector regression [16]. Examples of classification models include logistic regression [10], Naive Bayes [21], decision trees [42], and K-Nearest Neighbors. Modern neural network models can be adapted to either class, depending on the configuration of the output layer of the network. In the past, the selection of a model was based on a data scientist's past experiences, the type of data available, and the problem to be solved.

2.3.1.4 Model Training and Evaluation

Once a model is chosen, it usually must be fit or trained on data. In order for this to occur, metrics are necessary for evaluating the performance/loss of the model. For example, if a classification model is attempting to predict whether an event is newsworthy or not, then a method of determining its performance is necessary. Several metrics exist for categorical predictions such as accuracy, precision, recall and F-score. These metrics stem from analyzing the result of a categorical prediction. For a binary classification model (such as our newsworthy predictor) there are four potential outcomes:

- True Positive (TP). The model predicted the True when it should have predicted True.
- 2. False Positive (FP). The model predicted True when it should have predicted False.
- 3. True Negative (TN). The model predicted False when it should have predicted False
- 4. False Negative (FN). The model predicted False when it should have predicted True.

From these outcomes we can calculate the performance metrics as follows:

$$Accuracy(\%) = \frac{TP + TN}{TP + FN + FP + TN} \times 100$$
(2.1)

$$Precision = \frac{TP}{TP + FP} \tag{2.2}$$

$$Recall = \frac{TP}{TP + FN} \tag{2.3}$$

$$F_{\beta}Score = (1+\beta^2) \times \frac{Precision \times Recall}{(\beta^2 \times Precision) + Recall}$$
(2.4)

For regression models, evaluation metrics usually measure the numeric difference between a prediction and the true value. Different metrics can weight the amount of distance in different ways. For example, consider Absolute Error

$$AbsoluteError = |Y_1 - Y| \tag{2.5}$$

where Y_1 = Predicted Value and Y = True Value. This relatively simple calculation allows for statistical insight into how accurately a regression model performs.

2.3.1.5 Hyperparameter Tuning

Once the model and metrics are chosen, the next step is hyperparameter tuning. Hyperparameters are parameters which affect the learning behavior of the model. Nearly all of the models mentioned above have one or more hyperparameters. For K-nearest neighbors, the value of K and the similarity metric used for comparison are hyperparameters. For neural networks, learning rate, dropout percentage and the number of CNN filters are examples of possible hyperparameters. Such parameters are normally set before training begins and remain static. How these values are set, significantly affects the performance of the model. As a result, data scientists train models multiple times with different hyperparameter values in an attempt to determine the optimal value for these hyperparameters.

2.4 Supervised Classification Machine Learning Methods

Within STRAINER, we will be classifying events as newsworthy or not newsworthy. To perform this operation we chose to employ two different machine learning methodologies, Logistic Regression and Support Vector Machine. This section describes these methods.

2.4.1 Support Vector Machines

Support Vector Machines were first proposed by Cortes and Vapnik, in 1995 [9]. Within STRAINER, we utilize a Linear Support Vector Machine. Let us suppose that we have a training set X of n samples, each with m features and a corresponding label vector Y where $y_i \in \{1, -1\}$. In our case, 1 indicates newsworthy and -1 indicates not newsworthy. A support vector machine seeks to classify the samples in X by finding the optimal hyperplane which can be used to split the data into two groups (i.e. if a sample lies above the hyperplane then it belongs to the one class, if below, then it belongs to the other).



Figure 2.1: Example of Support Vector Classification in 2-D Space

This hyperplane can be defined as:

$$\mathbf{w}_o \cdot \mathbf{x} + b_o = 0 \tag{2.6}$$

Where \mathbf{w}_o is the set of weights which cause the margins between the hyperplane and the classes to be greatest. Optimizing these weights can be done using stochastic gradient descent(SGD).

2.4.2 Logistic Regression

Logistic Regression is a probabilistic classification method [24]. Given the labeled dataset from 2.4.1 above a logistic regression model predicts the probability of a sample being in a certain class by fitting a sigmoid function to the data where the probability for sample x_i is given by the the equation:



$$p(x) = \frac{1}{1 + e^{-\beta \cdot \mathbf{x}_i}} \tag{2.7}$$

Figure 2.2: Example of Logistic Regression Model Fitment

where β are feature weights. The optimal set of feature weights is typically estimated using Maximum Likelihood Estimation (MLE). This optimization may performed iteratively using SGD.

2.5 Natural Language Processing

A large portion of the the data used by STRAINER is in the form of unstructured text. To utilize this data, STRAINER relies upon Natural Language Processing (NLP). NLP, broadly speaking, is an area of research dedicated to the manipulation of human language, both spoken and written.

Traditional natural language processing begins with tokenization, whereby unstructured text is broken down into individual components of language, i.e. paragraphs, sentences and words, which are called tokens. These tokens are then tagged according to the language's specific part-of-speech rules. NLP is useful for extracting complex and subtle features from unstructured textual data. The Python language, the language chosen for this project, has several packages dedicated to performing NLP tasks. Within STRAINER, we use the spaCy package. This library allowed us to perform named entity recognition within the bill texts and bill analysis texts and utterances.

2.5.1 Named Entity Recognition

Named Entity Recognition refers to the process by which proper names and items are identified with in text. For example consider the sentence: "The California State Senate voted to pass bill SB 42 relating to police body cameras." Within this sentence their are two named entities, "California State Senate" and "SB 42". To identify these entities multiple automated systems have been implemented utilizing neural networks [7] [12]. The Python package SpaCy contains one such system. SpaCyutilized convolutional neural networks to identify and categorize named entities within texts. We use this Python package within STRAINER to perform named entity recognition on legislative texts.

Chapter 3

RELATED WORK

This thesis encompasses several academic fields namely computer science, data science and journalism. In order to find newsworthy events we must first know what characteristics to look for. For that we look to journalistic studies of news value. Once we know what to look for, we can build automated machine learning systems to find newsworthiness. This chapter discusses previous research in the area of newsworthiness and machine learning regarding news value.

3.1 News Value

In order to understand feature engineering choices for STRAINER, it is beneficial to first have some understanding of what aspects of an event make the event relevant and newsworthy. In 1965, Johan Galtung and Mari Holmboe Ruge, published seminal research regarding this topic. In their analysis of the flow of news regarding foreign countries, they proposed a set of 12 features which influence the probability of publication [18]. Descriptions of these twelve features are as follows:

- **Frequency:** The abruptness with which an event occurs. The more abrupt the event, the higher the news value.
- Threshold: Events must reach a certain level of intensity in order to be recorded.
- Unambiguity: The easier an event is to interpret, the higher the news value.

- **Meaningfullness:** The more in line an event is with the audience culture, the higher the news value.
- **Consonance:** The more inline with what the publisher wants, the higher the news value.
- Unexpectedness: The more unexpected the event, the higher the news value.
- Continuity: Events related to previous news has increased news value.
- **Composition:** Events which balance concurrent news entities within a news report have increased value.
- Elite Countries: Events referencing/involving elite/famous countries have higher news value.
- Elite Persons: Events referencing/involving elite/famous persons have higher news value.
- **Human Interest:** Events which affect or involve humans have higher news value.
- Negativity: Negative events have higher news value.

More recent research has questioned the general applicability of these features. In 2001, Tony Harcup and Deirdre O'Neill [22], conducted a study of the news selection process of British newspapers. They analyzed three prominent daily papers and scanned all of their page-lead news articles produced for the month of March, 1999. They scanned the articles for the presence of Galtung and Ruge's twelve features. Their findings corroborated the importance of some features, in particular, Unabiguity, Elite people, Frequency and Negativity. Based on their research, Harcup and O'Neill formulated a new list of ten features based on their findings and by combining and generalizing some of Galtung and Ruge's. These new features are as follows:

- The power elite: Stories concerning powerful individuals, organizations or institutions.
- Celebrity: Stories concerning people who are already famous.
- Entertainment: Stories concerning sex, show business, human interest, animals, an unfolding drama, or offering opportunities for humorous. treatment, entertaining photographs or witty headlines.
- Surprise: Stories which have an element of surprise and/or contrast.
- Bad news: Stories with negative overtones, such as conflict or tragedy.
- Good news: Stories with positive overtones, such as rescues and cures.
- Magnitude: Stories which are perceived as sufficiently significant, either in the numbers of people involved or in potential impact.
- **Relevance:** Stories about issues, groups and nations perceived to be relevant to the audience.
- Follow-ups: Stories about subjects already in the news.
- Newspaper agenda: Stories which set or fit the news organization's own agenda.

It is not within the scope of this thesis to build a system to take all of these issues into account when predicting news value. This is partly due to the nature of the data involved. State government legislative activities are not suited to entertainment. Nor do we consider any particular organization's agenda, as this would contribute to a loss of generality. However, some of these values are relevant for feature engineering within STRAINER. Table 3.1 describes how useful each news value category is when considering STRAINERS role as an impartial filter for news worthiness.

The Power Elite	The legislature certainly can be said to
	have a hierarchy of powerful individuals
	(i.e. governor, minority and majority lead-
	ers, bill authors etc.). Events involving
	such people should be easily identifiable.
Celebrity	Politicians are relatively famous, however,
	legislation concerns more than just famous
	people. During hearings, well known cor-
	porations/organizations, at times, are rep-
	resented/mentioned.
Entertainment	Legislative events are not traditionally
	known for their entertainment value.
Surprise	Some legislative events can cause surprise,
	Legislators voting contrary to expectations
	for example.
Bad News, Good News	Defining good and bad news in the con-
	text of legislative events relies too heavily
	on the perspective of the viewer to be a
	reliable feature for STRAINER.
Magnitude	Data from Digital Democracy allows for
	calculating the number of people involved
	in a legislative event quite easily.
Relevance	Since STRAINER is to be part of a ser-
	vice delivering content as a wire service we
	chose not to consider relevance as a feature.

Table 3.1: News Values Consideration for STRAINER

Follow-ups	At the time STRAINER was tested we did
	not have the correct data in order to utilize
	this feature.
Newspaper Agenda	This feature was not considered as it would
	lead to a loss of generality for STRAINER.

3.1.1 News Values and Machine Learning

In recent years some attempts have been made to automate the identification of news values. In 2012, Nies et al. sought to assign a news value score to existing articles[34]. They proposed a mapping between traditional news values and named entities and topics. With this mapping they utilized named entity recognition and topic modeling to assign a newsworthiness rating to articles. This represents the furthest attempt to classify news value using machine learning. However, this assigned newsworthiness to news articles texts that are already written not to events themselves. In contrast STRAINER seeks to identify and assign newsworthiness to events using raw data generated by the event itself. Additionally, topic modeling from textual data is less of a priority in STRAINER due the metadata surrounding legislative events such as the committee name. Most committees are formed for the purpose of addressing particular topics.

3.2 AI4Reporters

In 2019, a second project was initiated by the IATPP called AI4Reporters. This project sought to build upon the achievements of Digital Democracy by providing summaries of the events within the state legislatures in the form of short news arti-
cles. AI4Reporters built upon a research project offshoot of Digital Democracy, called RobotReporter, overseen by Dr. Foaad Khosmood. The goal of RobotReporter was to build a human readable article describing a legislative item or person within the legislature, such as a bill or a legislator. The articles generated by RobotReporter were extremely formulaic and followed a strictly templated article structure. RobotReporter's code base was repurposed as the starting point for AI4Reporters. The initial goal for AI4Reporters was to generate full news articles that could be distributed as a wire service for news organizations. However, the original RobotReporter article generation was not robust enough for this task. Thus, it was determined that two areas of research needed to be explored. First, article generation needed to be improved to create more human readable and less strictly formatted articles. This research was mainly conducted by Anastasiia Klimashevskaia whose Master's thesis focused on a partial order planner for sentence ordering and creation. [27] Second, a standardized method for identifying and extracting interesting legislative events and their corresponding data was needed. This second research goal was the main motivation for STRAINER. Today, AI4Reporters is expanding from generating an article to building tipsheets, which summarize legislative proceedings and provide journalists with a tailored web interface for quickly finding tipsheets and generated articles that match certain criteria. Portions of STRAINER are directly utilized by AI4Reporters. The extent of the this integration will be further discussed in Chapter 4.

Chapter 4

DESIGN

The goal of STRAINER is to be a machine learning pipeline which can detect newsworthy legislative events and surface relevant information. We enumerate the the design requirements which we place on STRAINER to achieve these goals as follows:

- The system shall allow the utilization of data from a variety of different sources including Digital Democracy, Website APIs, and website scrapers.
- The system shall collate data into segments surrounding individual events, or entities, within the legislature. These events may include bills, bill discussions, people, etc.
- The system shall extract features from the segments suitable for prediction.
- The system shall provide a means by which events can be labeled for the purposes of machine learning.
- The system shall perform predictions on legislative events using extracted features.
- The system shall evaluate prediction performance to allow for comparison of methodologies.
- The system shall be easily updatable as new methodologies are added.

An object oriented approach was adopted to encapsulate each stage of the pipeline. This design choice allows for greater maintainability and versatility of STRAINER. The following diagram provides an overview of STRAINER and its integration with AI4Reporters project:



Figure 4.1: Strainer Design Diagram

Beginning at the top of figure 4.1 we have *Data Sources*. *Data Sources* are use to interface with external data repositories (such as Digital Democracy). *Data Aggregators* query data from *Data Sources* and store the data into lists of *Data Model* objects. The list of *Data Model* objects is then passed to *Event Labelers* and *Event Feature Extractors*. *Event Labelers* tag events with ground truth newsworthiness labels for supervised machine learning. *Feature Extractors* surface and derive machine learning features from the information contained within *Data Models*. Both *Feature Extractors* and *Labelers* produce Pandas *DataFrames* of featurized events and labeled events respectively. These DataFrames are then joined together in order to be used by *Event Predictors*. *Event Predictors* use this training set to train machine learning models to predict newsworthiness. The main objects in figure 4.1 (shown in green) represent abstract Python classes, each with defined abstract methods. These abstract classes are not tethered to specific legislative events. Concrete implementations of these classes (discussed in chapter 5) are specific to certain events and utilize these defined methods to pass event data along the STRAINER pipeline. This allows pipeline control to remain the same for different legislative event types, increasing the flexibility of STRAINER to accommodate multiple event types.

In the rest of this chapter, we will discuss each of the abstract Python classes from figure 4.1 in more detail. In addition, we will discuss the integration of some of these classes into the AI4Reporters project.

4.1 Data Sources

Data Source classes within STRAINER provide an interface by which legislative event data can be accessed. Sources can be a database such as the Digital Democracy database, a website, or an api.

Abstract Class Data Source	
Abstract Method	Returns
Initialize(initial_params)	None
Close(termination_params)	None

Figure 4.2: Strainer Data Source Abstract Class

Each *Data Source* class inherits two abstract class methods which must be implemented: *Initialize* and *Close*. *Initialize* is used to perform any pre-data-retrieval tasks, such as connecting to a database or logging into websites etc. The *Close* method allows for any housekeeping functions, which must be performed after a *Data Aggregator* is finished with the *Data Source*. Concrete *Data Source* classes contain source-specific methods to return data to *Data Aggregators*. Splitting *Data Sources* from *Data Aggregators* allows multiple *Data Aggregators* to utilize the same source without code duplication.

4.2 Data Aggregators

The *Data Aggregator* class utilizes the functionality provided by Data Sources to collect data concerning and surrounding legislative events. The *Data Aggregator* is an abstract class from which specific event *Data Aggregators* inherit. Each concrete event *Data Aggregator* is assigned to collect data for a single event type, such as bill discussions, hearings, votes, bill introductions, etc. Each event *Data Aggregator* shares the same general functionality, namely, collecting data from *Data Sources* for a list of legislative events.

Abstract Class Data Aggregator		
Abstract Method	Returns	
Aggregate(event_ids)	list of Event Data Models	

Figure 4.3: Strainer Data Aggregator Abstract Class

A *Data Aggregator* class has only one inherited method *Aggregate*. This method compiles all data regarding a list of events from all available *Data Sources* and combines it into a single list of event *Data Model* objects.

4.3 Event Data Models

For the purposes of STRAINER, we can define an "Event" as any aspect of the legislative process that could be the subject of a news article. This could be a bill discussion, a legislator, a bill, etc. An *Event Data Model* is used to encapsulate all data related to such an event. *Event Data Models* store event data in a nested structure, i.e. a *Data Model* can contain references to other *Data Model* objects.



Figure 4.4: Example of Data Model Nesting

Figure 4.4 illustrates this nested structure. For example, a bill discussion event can reference multiple distinct models such as utterances and bill versions. In turn, utterances can reference particular speakers. *Data Aggregators* store retrieved event data from *Data Sources* into event *Data Model* objects. Designing our data structures with aggregation in mind gives us the ability to model multiple different legislative events, without having to design completely separate data structures each time. Event *Data Models* do not perform any operations on data and, therefore, do not have any methods. An argument can be made that event data could be contained within primitive dictionary-like structures. Dictionaries are very flexible data structures that can grow as needed. However, due to the nested nature of legislative event data, such dictionaries would become quite complex. Any additions of new data to a dictionary would have to be well documented outside of the code. Using defined class objects allow for more intuitive understanding of the relationships between legislative data.

4.4 Feature Extractors

Once data is stored in *Data Model* objects, the next step in the STRAINER process is feature engineering. This is accomplished by passing the data through a *Feature Extractor* class. This class identifies, isolates and calculates specific features of an event. Each *Feature Extractor* inherits one method, *get_features*, from a common parent class, *get_features* takes in a list of event *Data Model* objects*Data Models* and returns a Pandas *DataFrame* of featurized events. This ensures that future extraction methodologies can be integrated into STRAINER's pipeline with minimum code refactoring. Only one *Feature Extractor* concrete class is implemented for each event type.

Abstract Class Feature Extractor		
Abstract Method	Returns	
get_features(EventDataModels)	Pandas Dataframe of events with features	

Figure 4.5: Strainer Feature Extractor Abstract Class

4.5 Event Labelers

In order to perform supervised machine learning prediction, training, and test data must be labeled. In STRAINER, *Event Labelers* perform this action. Labeling can be as simple as assigning an existing event attribute as the event label. On the other hand, it can be as complex as creating a GUI interface for a manual human event evaluation. However, whether simple or complex, the end result is the same, namely, a collection of labeled events. STRAINER can support all types of event labeling by encapsulating event labeling into class objects. These objects share a common method *label* which takes in a set of events and returns a labeled event set.

4.6 Predictors

The final step in the STRAINER data pipeline is prediction, which is handled by *Event Predictors. Event Predictors* utilize the results of the *Feature Extractors* and *Event Labelers* to train and evaluate individual machine learning methodologies. Similar to feature engineering, prediction methods continue to be discovered and refined. Therefore, to ensure updatability, *Event Predictors* are class objects which inherit from a parent class containing the abstract methods, *fit, predict, evaluate.* When implemented in concrete classes, the fit method accepts a labeled dataset of events and prepares/trains the predictor model. Implemented *predict* methods accept an unlabeled dataset and return a set of predictions. The *evaluate* method returns a collection of information regarding the performance of the predictor.

4.7 AI4Reporters Integration

As this was a parallel project with AI4Reporters [27], some elements of AI4Reporters were taken directly from STRAINER. To understand our contributions to AI4Reporters (AI4R), it is necessary to briefly describe the design of its summarization system. AI4R has two main goals. The first is to summarize a bill discussion by generating a short human-readable news article. The second is to generate tipsheet documents, which contain the bulleted lists and figures highlighting interesting features of a bill discussion, bill or legislator. This tipsheet can then be utilized by reporters to write a news article. Both of these goals rely upon extractive summarization to formulate sentences describing the discussion. To implement this extractive summarization process, we designed a system for identifying and describing phenomena which occur within a bill discussion. Each phenomenon identification process is implemented within a class. If a phenomenon was found, the *Phenom* class generated a sentence describing what it found. These sentences were generated via a template system. Template sentences were filled with data extracted by their respective *Phenom* classes. The following figure describes this system.



Figure 4.6: AI4Reporters Summarization[27]

The AI4Reporers pipeline begins with a bill discussion. Data for this bill discussion is extracted from a database (DigitalDemocracy). This data is first used to determine if the discussion is newsworthy. STRAINER's event *Predictors* are used as a newsworthiness filter for AI4Reporters, prioritizing which bill discussions are summarized. The *Phenoms* are called in the order that sentences should appear in the generated article. Some *Phenoms* would depend on other *Phenoms* generating sentences in order to generate their own. To preserve sentence ordering, the *Phenoms* would be called in a partially ordered manner. Once all of the top level *Phenoms* and their dependents wrote a sentence, the article was considered complete.

4.7.1 EventLabeler to Phenoms

Within STRAINER's implementation for bill discussion events, we created a rules based event *Labeler* (more on this in Chapter 5). This *Labeler* would label a discussion newsworthy based on the presence of one or more criteria. It came quickly to our attention that these criteria could be seen as individual phenomena suitable for inclusion within AI4R's summarization. Thus, we broke down the event *Labeler* into a set of *Phenom* classes which could write headlines describing the interesting event within the bill discussion.

4.7.1.1 Data Source object as API for AI4Reporters

All of the *Phenoms* within AI4R relied upon data sourced from the Digital Democracy database. STRAINER's implementation of Digital Democracy's *Data Source* object was used as a basis for an API which served the needs of AI4R's *Phenoms*. This API acted as an interface between AI4R and the Digital Democracy database.

4.7.1.2 Metadata Phenoms in AI4R

Some phenoms in AI4R served to merely summarize the facts surrounding a bill discussion, such as voting records, bill subject, and speaker number. These were labeled as *metadata Phenoms* and were based upon STRAINER'S *Bill Discussion Feature Extractor* class. This class (described fully in Chapter 5) already calculated and extracted these types of summarization facts for the purposes of prediction, and therefore, was ideally suited for aiding in the construction of *metadata Phenoms*.

Chapter 5

IMPLEMENTATION

In the context of the AI4Reporters project, our goal is to use STRAINER to rate the newsworthiness of bill discussions which take place in the legislature. Achieving this goal resulted in an automated filter system for prioritization of bill discussion summarization and/or tipsheet generation. Thus the "event" which we chose to apply STRAINER to was a "Bill Discussion" event. The following chapter discusses the STRAINER implementation specific to this task.

5.1 Bill Discussion Data Sources

We implemented two *Data Source* classes to gather data surrounding bill discussions and their relative newsworthiness. The first *Data Source* class dealt with the extraction of bill news data from the article records-keeping website *newspapers.com*. The second class dealt with the extraction of discussion data from the existing Digital Democracy database.

5.1.1 Newspapers.com Data Source

Newspapers.com, an offshoot of Ancestry.com, was founded in 2012. It is an online historical database containing over 68 million historical newspaper clippings from California news organizations including The Sacramento Bee, Los Angeles Times, The San Francisco Chronicle among many others [33]. The *Newspapers* class was created in order to retrieve articles from this database relevant to bill discussions. The definition of "relevant" in this context is discussed in section 5.5.2. The Newspapers class contains a single unique public method get_papers. Given a list of bill discussion model objects this method returns a list of discussion IDs, each labeled with the number of relevant articles found on newspapers.com. Using the Selenium Python package to automate web browser interactions, the process for obtaining relevant articles is as follows.



Figure 5.1: Relevant Article Retrieval

When Newspapers' get_data method is called with a set of discussions, each discussion model is parsed and the bill name is extracted. This bill name is the short name for the bill such as "SB 42". For the purpose of retrieving articles that mention the bill, we build the long name as well, i.e. "SB 42" would become "Senate Bill 42". Using the date of the bill discussion hearing, we generate a date range of acceptable articles. This date range is defined as thirty days before the discussion took place until seven days after. Next, newspapers.com is queried to find all articles written by California newspapers that were published within the date range and contained one of the versions with that bill name. The results of this search includes "jpg" images of the newspaper articles, the name of the newspaper, and the publication date. Then this data is downloaded from the website and the article images are converted to test strings using the Pytesseract Python library, a Python wrapper for Tesseract OCR[30]. Next, the resulting text is double-checked to ensure that the bill names appeared within the article. If the article passes this check, its data is stored as relevant to the bill discussion. Once this process completes for all discussions, the resulting table of relevant articles is returned from *get_data*. After roughly two months of scraping this data from *newspapers.com*, we had at our disposal, a dataset of article information for the 2017-2018 session year.

DiscussionID	Bill Type	Bill Number	Hearing Date	Article Source	Article date
25659	AB	162	2017-05-26	None	None
26524	AB	72	2017-05-30	None	None
19913	SB	12	2017-03-15	The Los Angeles Times	2017-02-23
19913	SB	12	2017-03-15	The Los Angeles Times	2017-03-02
19913	SB	12	2017-03-15	The Los Angeles Times	2017-02-24
19913	SB	12	2017-03-15	The Fresno Bee	2017-03-12

Table 5.1: Excerpt From Article Labeling Initial Dataset

A slice of this data is shown in table 5.1. This dataset was then flattened on DiscussionID to produce a table containing only discussion IDs and the number of associated articles.

5.1.2 Digital Democracy Data Source

The primary Data Source for bill discussion data is the Digital Democracy database. All methods used to extract data from this data source were contained within a *Data Source* class named *DDDatabase*. The following table summarizes the methods implemented within this class.

Method		Description
query(q,vs, df=I	False,	Used internally within the database class
fetchone=False)		to perform the mysql query to retrieve data
		from the database.
get_bill_analysis(self, did)		Retrieves the raw text of a bill analysis for
		a particular bill discussion identified by the
		discussion ID (did).
get_bill(self, did)		Retrieves the bill ID of the bill under dis-
		cussion.
get_discussion_ids(self, session_	year,	Retrieve all discussion IDs, along with the
state)		dates that they took place and the hear-
		ing IDs which identify the hearings within
		which the discussions took place for the
		given session year and the given state.
get_hearing_ids(self, time_fr	ame,	Retrieves the hearing IDs for all the hear-
state):		ings within a given session year for a given
		state.
get_committee_id(self, did)		Retrieves the committee ID conducting the
		discussion given the discussion ID.
$get_committee_name(self, cid)$		Retrieves the committee name given the
		committee ID
get_num_speakers(self, did)		Retrieves the number of unique speakers
		for the given discussion.
$get_avg_num_speakers(self, cid)$		Retrieves the average number of discussion
		speakers for a given committee

Table 5.2: DDDatabase Clase Methods

get_stdev_num_speakers(self, cid)	Retrieves the standard deviation of the
	number of discussion speakers for a given
	committee
get_speaker_pids(self, did)	Retrieves the person IDs (pids) of the dis-
	cussion speakers for a given discussion.
$get_voter_pids(self, vote_id)$	Retrieves the pids of the voters who took
	part in a given vote
$get_bill_authors(self, bid)$	Retrieves the pids of the bill authors of a
	given bill
get_legislator_first_bill(self, pid)	Retrieves the bill ID of the first bill intro-
	duced by the given legislator pid.
get_vote_id(self, did)	Retrieves the vote ID corresponding to the
	vote that took place after a given discus-
	sion.
get_nay_voters(self, vote_id)	Retrieve the pids of the legislators who
	voted against a motion for a given vote.
get_voters(self, vote_id)	Retrieves the pids of the legislators who
	voted for a given vote ID.
get_bill_discussion_utterances(self,	Retrieves the utterances of a bill discussion
discussion_id)	in order complete with speaker ids and ut-
	terance alignment information.
get_vote(self, pid, vote_id)	Retrieve the individual vote result for a
	given legislator and a given discussion.

$get_vote_info(self, did)$	Retrieves information about the vote that
	took place within a given discussion re-
	turning a dictionary containing the follow-
	ing attributes (vote_id, ayes, nays, abstain,
	motionText, result, voter information (it-
	self a list of dictionaries containing: pid,
	first, last, vote, party, district)
get_speaker_utterances(self, did, pid)	Retrieves the utterance records for a given
	speaker and a given discussion.
get_bill_discussions_from_	Retrieves the discussion IDs, correspond-
hearing(self, hid, hearing_date)	ing bill IDs and bill version IDs for a given
	hearing and hearing date.
get_bill_type(self, bid)	Retrieves the bill type for a given bill
get_bill_versions(self, bid)	Retrieves all bill versions for a given bill ID
	along with the version date.

5.2 Data Models for Bill Discussion Events

Information for each bill discussion is encapsulated within a network of hierarchical data models. The highest level data model is *BillDiscussion*. This model is implemented as a Python class with the attributes listed in Table 5.3.

Attribute	Description
cid	Committee ID
committee_name	The committee given name
hearing_id	Hearing ID
date	Discussion date
bill_id	ID of bill under discussion
discussion_id	ID of the discussion
utterances	Python list of individual utterances made
	during the discussion.
utterance_string	All utterances combined into one string
	representing the entire discussion.
bill_version_id	the specific version of the bill that is under
	discussion.
num_speakers	The number of unique speakers that took
	part in the discussion.
num_voted_for	The number of legislators that voted in fa-
	vor of the motion regarding the bill after
	the discussion took place.
$num_voted_against$	The number of legislators that voted
	against the motion regarding the bill after
	the discussion took place.
num_abstain	The number of legislators that abstained
	from voting on the motion regarding the
	bill after the discussion took place.

 Table 5.3: Bill Discussion Data Model Class Attributes

$num_rep_voted_for$	The number of Republicans that voted in
	favor.
$num_rep_voted_against$	The number of Republicans that voted
	against.
$num_rep_abstain$	The number of Republicans that abstained
	from voting.
$num_dem_voted_for$	The number of Democrats that voted in
	favor.
$num_dem_voted_against$	The number of Democrats that voted
	against.
num_dem_abstain	The number of Democrats that abstained
	from voting.
is_dopass	Boolean value indicating whether or not
	the motion voted on was a motion to pass
	the bill on to the next step in the legislative
	process.
vote_result	A categorical variable indicating the out-
	come of the vote on the motion (pass, fail).
bill_analysis_text	The text of the bill analysis prepared for
	the committee discussing the bill.

Since the legislature can hold multiple hearings, multiple bill discussions can reference the same bill version. To save memory space, bill information is stored within separate data structures and can be referenced by multiple bill discussion models. The two classes for *Bill* and *BillVersion* are summarized in Tables 5.4 and 5.5.

Attribute	Description
bid	Bill ID
type_	The type of bill e.g. Assembly (AB) or
	Senate (SB) There are 22 unique bill types.
versions	List of Bill Versions for the bill
num_versions	Count of the number of versions for the
	bill.
version_max_length	The max version length
version_min_length	The min version length

 Table 5.4: Bill Data Model Class Attributes

Table 5.5: Bill Version Model Class Attributes

Attribute	Description
date	Version Date
$version_id$	Version ID
bid	Bill ID
subject	Bill Subject; A short text description of
	the bill
$is_{-}appropriation$	A Boolean indicating if the bill is an ap-
	propriation
$version_text$	The text of the bill version
$version_digest$	The digest of the bill version
$version_length$	The length of the bill version
$digest_length$	The length of the bill digest

5.3 Bill Discussion Event Data Aggregator

The Bill Discussion Event Data Aggregator is the first active portion of the STRAINER pipeline. It has a single public function *aggregate* which requires two arguments, session_year and state. Using these arguments, this function queries the *DDDatabase* data source and finds all the data for bill discussions from the given session year and state. Some filtering occurs here as well. Only bill discussions that discuss a bill, contain vote information, contain utterances, and are not discussing bills of type "BUD" are retrieved¹. In addition, party specific vote information (e.g. num_rep_voted_for) is pre-calculated and stored into data models. It returns this data as lists of the data models described in the previous section, namely *BillDiscussion, Bill*, and *BillVersion*. These lists of data models are then passed to the *Bill Discussion Feature Extractor*.

5.4 Bill Discussion Feature Extractor

While simple features can be extracted directly from the data models themselves, others must be derived from the data. This occurs within the *Bill Discussion Feature Extractor*. This class ingests the collections of *Data Models* filled by the *Data Aggregator* and outputs a Pandas *Dataframe* where each row is a bill discussion and the columns are either numeric, categorical, or unstructured text features suitable for feeding into a scikit learn machine learning pipeline. The following table describes the methods within *Bill Discussion Feature Extractor*.

¹ "BUD" bills are budget bills and, while certainly interesting, they were excluded from this implementation of STRAINER, since discussions surrounding these bills do not focus on a particular topic.

Method	Description
init(self, data)	Constructor. The <i>data</i> argument contains
	the collection of data models generated by
	the Bill Discussion Data Aggregator. This
	takes the collection and splits it into its re-
	spective data models ready for extraction.
get_features(self, numeric_only =	This is the inherited public facing method
True)	of the <i>Feature Extractor</i> . This function re-
	turns a <i>DataFrame</i> of features extracted
	from the Data Models.
extract_features(self, discussions)	Function called by get_feature_dataframe
	generates features for a given list of dis-
	cussions.
$get_named_entity_counts(bill_digest,$	Utilizes Spacy to retrieve counts of the dif-
<pre>bill_text, utterances, bill_analysis)</pre>	ferent types of named entities that occur
	within textual data. Called by <i>extract</i> _
	features.
$get_word_count(utterances)$	calculates the number of words in the dis-
	cussion.
utterance_word_count(utterance)	calculates the number of words in a single
	utterance, called within <i>get_word_count</i> .

 Table 5.6: Methods of Bill Discussion Feature Extractor

The main methods of the feature extractor are *get_features* and *extract_features*. Due to the amount of legislative data, feature extraction is parallelized within *get_features*. The lists of discussion data models are split and separate processes are created to call

extract_features for each section of the data. Once all the processes are finished, the resulting tables are joined together into a singular table. For this implementation of STRAINER, the *Bill Discussion Feature Extractor* creates the following features.

Feature	Feature Type	Description
DiscussionID	Categorical	Unique Id to identify a bill
		discussion. Used only to
		link to labels for supervised
		machine learning.
date	Date	The date of the bill discus-
		sion
committee_id	Categorical	The committee that dis-
		cussed the bill
committee_name	Categorical	The committee name
discussion_word_count	Numeric	The number of words spo-
		ken in the discussion.
avg_words_per_speaker	Numeric	The average number of
		words per speaker in the dis-
		cussion.
bill_version_length	Numeric	The number of characters in
		the text of the bill under dis-
		cussion.
bill_is_appropriation	Categorical	Boolean flag indicating
		whether or not the bill is an
		appropriations bill.
bill_subject	Text	Subject of the bill.

 Table 5.7: Top Level Extracted Features

bill_version_digest	Text	A brief description of the
		bill
bill_version_text	Text	The full text of the bill.
bill_analysis_text	Text	Full text of the most recent
		bill analysis created prior to
		the bill discussion.
number_of_speakers	Numeric	The number of speakers who
		took part in the discussion.
num_voted_for	Numeric	The number of legislators
		who voted in favor of the
		motion proposed at the end
		of the discussion.
num_voted_against	Numeric	The number of legislators
		who voted against the mo-
		tion.
num_abstain	Numeric	The number of legislators
		who abstained from voting.
num_rep_voted_for	Numeric	The number of Republicans
		who voted for the motion.
num_rep_voted_against	Numeric	The number of Republicans
		who voted against the mo-
		tion.
num_rep_abstain	Numeric	The number of Republicans
		who abstained.
num_dem_voted_for	Numeric	The number of Democrats
		who voted for the motion.

$num_dem_voted_against$	Numeric	The number of Democrats
		who voted against the mo-
		tion.
$num_dem_abstain$	Numeric	The number of Democrats
		who abstained.
is_dopass	Boolean	A Boolean value which indi-
		cates whether the motion is
		a "dopass" motion.
vote_result	Boolean	Indicator for whether the
		motion passed or failed.

Additionally, for each of the "Text" type features in table 5.7 a set of sub-features is produced. Each text is passed through a named entity recognition system powered by the Spacy Python library. The entities are split into the following types.

Abbreviation	Description
PERSON	People, including fictional.
NORP	Nationalities or religious or political
	groups.
FAC	Buildings, airports, highways, bridges, etc.
ORG	Companies, agencies, institutions, etc.
GPE	Countries, cities, states.
LOC	Non-GPE locations, mountain ranges,
	bodies of water.
PRODUCT	Objects, vehicles, foods, etc. (Not ser-
	vices.)
EVENT	Named hurricanes, battles, wars, sports
	events, etc.
WORK_OF_ART	Titles of books, songs, etc.
LAW	Named documents made into laws.
LANGUAGE	Any named language.
DATE	Absolute or relative dates or periods.
TIME	Times smaller than a day.
PERCENT	Percentage, including "%".
MONEY	Monetary values, including unit.
QUANTITY	Measurements, as of weight or distance.
ORDINAL	"first", "second", etc.
CARDINAL	Numerals that do not fall under another
	type.

 Table 5.8: Spacy Named Entity Categories [11]

The number of occurrences of each of these entities within each text is recorded and added as a numeric feature of the bill discussion event.

5.5 Bill Discussion Labelers

Bill discussions begin as unlabeled events. While some analysis can be conducted on unlabeled data (clustering, anomaly detection), it was our goal to train a model which could predict newsworthiness of discussion events. Conventional supervised models require labeled training sets to achieve this. We utilized two labeling methods to label discussion event data: rules-based and proximal association.

5.5.1 Rules Based

The first labeling system is a rules-based Bill Discussion Labeler. Certain phenomena surrounding a discussion which, if present, was deemed as evidence of a newsworthy discussion. Five "newsworthy" phenomena were chosen for this labeling system.

R1. Author voted against their own bill. After consulting with domain experts, we chose to label any discussion which resulted in the bill author voting against their own bill as a newsworthy event. This event is extremely rare, likely caused by amendments to the bill or the type of motion being voted on.

R2. Maverick voters. In legislative houses it is common and expected that members of the same party vote the same way. A legislator voting opposite the way of their party likely indicates a contentious and perhaps controversial issue.

R3. Unusual attendance. Some discussions attract an inordinate amount of speakers. The popularity of such discussions likely indicates a newsworthy event. For labeling purposes, we considered a bill discussion interesting if it drew more than two standard deviations above the average number of speakers, for the committee holding the discussion.

R4. Flip Flop. Some speakers may speak in favor of a bill but in the end vote against it. Within the Digital Democracy database, each utterance made by a legislator is labeled as "support", "against", "neutral" or "unknown", with regard to their position towards the bill under discussion. This label was assigned at the time of transcription. We found it unusual and thus interesting when a legislator's overall utterances were in support, but the legislator ended up voting against the bill.

R5. First bill of a legislator. The first bill of a legislator is likely of interest to the residents of that legislators district. Therefore, any discussion of the first bill of a legislator is likely to be interesting.

This labeler was implemented as the Python class *BusinessRulesEventExtractor*. This class determines how many of the rules described above are valid for each discussion and labels them accordingly. For the purposes of supervised learning, if one or more of the rules are valid then the discussion is marked as interesting. Table 5.9 describes the principle methods within this class. Each rule is encapsulated within its own method and handcrafted to check for the rule triggers. These methods rely upon the *DDDatabase* class and utilizes its "get_" methods to perform the necessary mySQL database queries to extract information from the Digital Democracy database. Any number of rules can be added in the future by adding further methods within *BusinessRulesEventExtractor*.

Method	Description
init(db)	Public method which sets up the Labeler,
	the parameter "db" is an instance of the
	DDDatabase class.
label(bill_discussions)	Public method which takes in a list of bill
	discussion IDs and individually assigns a
	label to them depending upon how many
	rules were triggered by the discussion. Re-
	turns a DataFrame of labeled discussions.
check_bill_discussion(bill_discussion_	A private method called by <i>label</i> . Checks
id)	for any rules triggered by that discussion.
legislator_voted_against_own_	Private method called by <i>check_bill_</i>
bill(did)	discussion. Queries the DDDatabase to
	check if any of the bill's authors voted
	against during the discussion. Any au-
	thors who did vote against are returned
	to $check_bill_discussion$ as a list containing
	their pids(Person Ids).

 Table 5.9:
 BusinessRulesEventExtractor
 Class
 Methods

$legislator_get_voted_against_$	Private method called by <i>check_bill_</i>
party(did)	discussion. Checks for maverick voters by
	querying <i>DDDatabase</i> for vote information
	for the given discussion. Next, it splits
	this information by party and determines
	which way the majority of the party voted.
	Any legislators with opposing votes are
	tagged as maverick votes and these legis-
	lator's pids returned as a list to <i>check_bill_</i>
	discussion.
$calculate_attendance_irregularity_$	Private method called by <i>check_bill_</i>
level(did)	discussion. Compares the number of
	speakers at the discussion with the aver-
	age number of speakers for that committee.
	Returns the number of standard deviations
	that the discussion's attendance is from the
	average. If it is more than two, then the
	attendance is considered irregular.
$legislator_flipped_when_voted(did)$	Private method called by <i>check_bill_</i>
	discussion. Queries DDDatabase for the
	alignment of the utterances of discussion
	voters. If a majority of a voter's utterances
	are in favor, but the voter votes "nay", or
	vice-versa, then the voters pid is recorded
	in a list. This list of flip-floppers is re-
	turned.

legislator_first_bill(did)	Private method called by <i>check_bill_</i>
	discussion. Checks the bill authors of the
	bill under discussion. Any first time bill
	authors are returned in a list.

5.5.2 Article Proximal Association

The second labeler is based upon the bill discussed by the legislature and human generated news articles referencing a specific bill. Some bills were mentioned by name in newspaper articles published around the time of the bill discussion. The more controversial/popular the bill, the more newspapers would write about the bill or buy the rights to publish the same article about the bill. Given this observation, we claim that a bill discussion is more likely to be newsworthy if the bill under discussion has been written about by news outlets. Thus we designed a labeler to label the bill discussions based upon the number of news articles referencing the bill discussed. Any discussion with at least one proximal news story is considered to be an interesting and therefore newsworthy discussion.

Articles were considered proximally associated if the following set of criteria was met:

- 1. The article must be published no earlier than 30 days before the discussion took place and no later than 7 days after.
- 2. The article must mention the name of the bill under discussion at least once.

Article data gathered programatically from newspapers.com as outlined in section 5.1.2 was used to label discussions according to the criteria.

5.6 Bill Discussion Predictors

We created two machine learning models to predict newsworthiness, a linear support vector machine and a logistic regression model. These predictors were structured from models within sklearn machine learning package [37]. The linear support vector machine model was encapsulated within a *Predictor* Python class named *BillDiscus*sionSVMPredictor. The logistic regression model was contained within a Python class named *BillDiscussionLRPredictor*. These classes were quite similar to each other. They both contained three public methods, fit, predict, evaluate. The fit method ingested a set of training data created by joining discussion feature dataframes with labels from either the rules based labeler or the articles based labeler. While feature extraction is conducted prior to passing the data to event predictors, some feature manipulation is easier done within the sklearn pipeline, generating TF-IDF matrices, for example. Therefore, some basic feature manipulation occurs within the predictors. For all Text features in the dataset (version text, digest, subject, and bill analysis) TF-IDF matrices are generated. The use of a stopword list can be optionally employed when generating these TF-IDF matrices. All numeric variables are scaled and categorical variables are one-hot encoded. The discussion date field, discussion_id and *committee_id* fields are not used.

The addition of the TF-IDF matrices to our final training dataset resulted in a large amount of features for our machine learning models. To improve training performance, both the linear SVM and the Logistic regression models were implemented using sklearn's SGDClassifier with L2-Norm regularization on accuracy. Stochastic Gradient Descent offers better performance as parameter optimization is nearly linear with respect to the number of samples[45]. Once the *fit* method has been called, the *predict* method can be called. This method ingests a Dataframe of unlabeled discussion features and returns binary newsworthy predictions. The evaluate method ingests a Dataframe of labeled discussion features and separates it into test/train datasets. 80% of the data is used for training, 20% is withheld for testing. When training the models, training and testing data were under-sampled in order to maintain an even 50/50 split of data labels.

5.7 Implementation within AI4Reporters

This thesis was produced in conjunction with a second thesis project undertaken by Anastasiia Klimashevskaia [27]. Her work focused on generating summaries of individual bill discussions. This process followed a similar path to that of STRAINER, however, instead of creating a dataset of many bill discussions used for machine learning, the goal was rather to extract phenomena from an individual discussion and then generate sentences and build an article describing said phenomena. The combination of this thesis' work and that described in [27] form the majority of the AI4Reporters summarization pipeline.

5.7.1 Metadata extraction

Since AI4Reporters relied upon the same database of legislative data, we were able to use STRAINER's *DDDataBase* Data Source as an API to extract the data to be summarized. Metadata such as discussion dates, vote records and bill information were used to trigger Metadata Phenoms within AI4Reporters. These Metadata Phenoms, in turn, generate sentences describing the discussion. Metadata Phenoms tell the article reader when the discussion took place, what bill was discussed, how legislators voted, and what the outcome of the vote was. All of these phenoms relied upon the methods within *DDDatabase*.

5.7.2 Newsworthiness Filter

While it is possible for AI4Reporters to generate an article about every single discussion that takes place, the sheer number of discussions makes this quite resource intensive. In addition, many discussion articles that contained little content often generated sparse articles. Therefore, STRAINER is used as a pre-filter for AI4Reporters. Specifically, the Rules-Based Bill Discussion labeler, which was created early on in the course of the project, was used as a guide for finding discussions that are more likely to be interesting.

5.7.3 Headline and Intro Phenoms

Once discussions were filtered by the rules-based labeler, it became apparent that each of the rules within the labeler described an interesting phenomenon of the bill discussion. Therefore, each rule was used as a basis for a "Headline" phenom. These *Phenoms* would generate headlines for the AI4Reporters' articles based upon the valid rule found. Also, when a headline phenom sentence was generated, secondary *Phenoms* would place supporting sentences within the introduction of the article to further support the headline and/or explain why the headline was interesting.

Chapter 6

EVALUATION

As mentioned in the introduction of this thesis, the primary goal of STRAINER is to reduce the decline in journalistic oversight at the state level by surfacing interesting events. To that end, STRAINER makes predictions on the newsworthiness of legislative event data. Our hope is that STRAINER can identify relevant and interesting hearing information which would normally be found by human newsreporters/editors.

This evaluation chapter is organized as follows. Section 6.1 outlines how we set up and conducted our evaluation. Section 6.2 defines the precise bill discussion data used for evaluation; it also inspects the results of labeling this data. Section 6.3 discusses the specific configurations of the machine learning models tested, along with a discussion of the feature sets tested. Section 6.3.1 outlines the test set results of the best and worst *Predictor* model configurations and discusses and these *Predictors*' results.

6.1 Study Design

We designed our study to test the performance of our bill discussion newsworthiness predictors as follows. We define the following set of hypotheses.

- STRAINER can be trained to perform better than random at predicting general newsworthiness.
- STRAINER's trained performance approaches the performance of human systems.
The first of these hypotheses is straightforward to test. There are two categories, newsworthy and not newsworthy. Therefore, given a set of events in which half are labeled newsworthy, STRAINER must achieve an accuracy level greater than 50%.

The validation of the second hypothesis is more complex to determine. There is no current human system available to determine the *general* newsworthiness of a bill discussion. To be sure, each newspaper currently has a method for choosing which stories to highlight, however, these methodologies are often unquantifiable, gut-feeling decisions made by editors. Using this type of system to evaluate the performance of STRAINER would result in a loss of generality. Editors of specific newspapers are biased to their audience and ideals. Countering this bias would require soliciting the opinions of as many distinct newsrooms as possible. This kind of evaluation was not possible within the scope of this thesis.

In lieu of this constraint, we must rely upon the two labeling techniques, described in Chapter 5, to serve as the ground truth for a bill discussion's newsworthiness and as placeholders for a human newsworthiness rating system. However, these labeling techniques are not without their own set of limitations. The primary concern is that these techniques only identify some positive cases of newsworthiness. They do not identify negative examples of newsworthiness at all. This affected how we judged the performance of STRAINER and which metrics we chose for evaluation.

6.1.1 Metrics

Performance metrics for classification tasks typically include overall accuracy, precision, recall and f-score for each individual class (or, just for the positive class, as is the case with our classification problem). Our work needs to address several caveats related to the selection of appropriate metrics. **Both sources of ground truth.** Both rules-based labeling and proximal labeling present highly imbalanced classes (see Section 6.2), with the positive class (news-worthy bill discussions) being about 8-12% of our dataset. In such an environment, overall accuracy is not the best classifier performance metric, as a classifier selecting just the majority class can easily achieve 90% accuracy. As the result, we **do not use overall accuracy** in our evaluation.

Rules-based labeling. As we do not claim that the set of rules implemented in our rules-based labeling process is complete, this process surfaces only a subset of articles that could potentially be considered newsworthy. At the same time, the labeling is complete on our training data w.r.t. the specific selected rules. We agree to treat the respective classification task as predicting whether a given discussion matches one of the rules. In this situation, both errors of commission (discussions our classifier thought were a match, but that did not satisfy any of our rules) and errors of omission (discussions that satisfied at least one of our rules, but were not discovered by the classifier) are meaningful errors, and need to be evaluated. As a result, for rules-based labeling, we look at precision, recall, and the f-measure of the positive ("newsworthy") class. To determine the best results, we use the f1-measure.

Proximal article labeling. In evaluating the performance of our classifiers trained on proximal article labeling data, we have to contend with two issues that make *errors* of commission unreliable. First, our collection method for proximal article labeling may have not discovered a published story about a specific bill (that followed a bill discussion). This alone means that only the positive labels received from our proximal labeling process are trustworthy, while the negative labels are not evidence that a bill discussion is not newsworthy. This is enhanced by our belief that the classifier trained using proximal article labeling may actually surface *bona fide* newsworthy bill discussions whose proximal article labels are negative, and where no news stories were actually published. As a result, for proximal article labeling, only the errors of omission (bill discussions for which a proximal article exists that were not caught by the classifier) can be properly evaluated. We, on the other hand, cannot claim that any errors of commission (bill discussions for which no proximal article was surfaced, but which are caught by the classifier) were actual mistakes made by the classifier.

Thus, for proximal article labeling, we shall only use recall for the positive class as the measure of performance. We report precision and f-measures for the reader's information, but do not use them in our evaluation of the classifier performance. To ensure that the classifiers are not achieving high recall by being indiscriminate in their labeling of positive examples, we also report selectivity of each classifier: i.e., the percentage of all possible bill discussions that the classifier labeled as "newsworthy". Sufficiently low selectivity protects us from selecting a classifier that achieves high recall only by being indiscriminate in its labeling.

6.1.2 Variables

For the purposes of evaluation, we treated the individual machine learning models within STRAINER as the independent variables in our test. The dependent variables consisted of the different accuracy metrics calculated from the predictions for bill discussions.

6.2 Dataset Inspection

For evaluation, we chose to examine legislative discussion data from the 2017-2018, California legislative session year. Not all discussions stored within Digital Democracy were used. Some discussions were procedural and do not reference legislative bills. Only discussions that had an associated bill and a recorded vote were considered. In total, 11204 bill discussions met this criteria and were passed through STRAINER. This process yielded a data frame of 11204 sets of discussion features, each possessing 168 attributes useful for machine learning. Four of these attributes (bill_digest, bill_version_text, bill_subject and bill_analysis), were further expanded into TF_IDF matrices within the Predictors.

6.2.1 Rules Based Labeling Results

When the rules-based labeler was run across the set of 11204 discussions, 1347 discussions (12%) were labeled as newsworthy as shown in figure 6.1.



Figure 6.1: Rule-Based Labeling Results California 2017-2018 Session



Figure 6.2: Rule-Based Intersection Results California 2017-2018 Session

Figure 6.2 is an Upset plot of the intersections which exist within the rules-based labeling of the bill discussions. Upset plots were introduced in 2014, by Lex et al., as a way to visualize the intersection aggregates of multiple sets[31]. The total number of bill discussions which triggered each rule is visualized on the left bar chart. Every intersection (and non-intersection) of the rules is shown and quantified on the top bar chart. The dot-plot serves as a legend representing the intersection sets of the top bar chart. For example, we can see that a total of 308 bill discussions had unusually high attendance, 261 triggered this rule, 31 discussions had both high attendance and discussed a legislator who voted against their party, 6 discussion had high attendance and discussed a legislator's first bill, and 4 discussions had high attendance, legislator flip-flops and legislators voting against their party. Overall, 1280 bill discussions had only one rule triggered. 63 had two rules triggered while only 4 had three rules triggered. In total, 1347 out of 11204 discussions triggered at least one rule, which is roughly 12% of discussions. Labeling the data in this way resulted in a dataset that is quite unbalanced and influenced how we trained the bill discussion predictors.

6.2.2 Proximal Article Labeling

Labeling the 11204 discussions with the Proximal Article Labeler yielded 914 (8%) discussions with at least one proximal article, shown in figure 6.3.



Figure 6.3: Articles per Discussion Labeling Results California, 2017-2018 Session



Figure 6.4: Articles per Discussion Labeling Results California, 2017-2018 Session

Figure 6.4 represents the distribution of proximal articles across bill discussions. Each bar on the graph indicates the amount of discussions proximal to a given number of news articles. Again, like the rules-based labeler, there are proportionally very few positive cases (918) of discussions linked to news articles.



Figure 6.5: Rules-Based/Articles-Based Labeling Overlap California, 2017-2018 Session



Figure 6.6: Rules-Based/Articles-Based Labeling Overlap For Each Rule, California 2017-2018 Session

Figure 6.5 is a Venn diagram showing the overlap between the two labeling techniques. 250 discussions were labeled newsworthy by both labeling methods. Figure 6.6 shows the labeling overlaps split into overlap of discussions labeled "article-newsworthy" vs each "rule-newsworthy". It appears that the "Unusual High Attendance" rule, with 137 overlaps, has an out-sized amount of overlaps compared to the rest of the rules.

6.3 Predictor Model Configuration and Training

Labeling bill discussion events resulted in a dataset of 11204 events labeled two ways (Rules-Based and Article-Based). Since the labeling of the data was so unbalanced for both labeling methods, the data was undersampled before we split it into training and testing sets. This resulted in a dataset of rules-labeled data consisting of 1347 newsworthy discussions and 1361 non-newsworthy. The undersampled article-labeled dataset, consisted of 918 newsworthy discussions and 902 non-newsworthy. Twenty percent of this labeled data was withheld as a separate test set. The remaining 80%

was used for training. When randomly dividing the 80/20 split, the proportion of positive and negative labels in the train and test sets was held at 50% +/- 0.1%.

For both machine learning models used, we varied input parameters to allow us to view and discover the most useful characteristics of a bill discussion for determining newsworthiness. Then, each configuration's performance on the holdout test set was then recorded. For the purposes of machine learning, a numeric label of '1' represents newsworthy and '-1' represents not newsworthy. When conducting our search for the best feature set, we broke the features into several data groups.

- Participation Data: This contained only the num_speakers feature
- Bill Analysis Data: This contained the TF-IDF matrix calculated from the Bill Analysis.
- Bill Text Data: This contained TF-IDF features from the Bill Subject text, Bill version text and the Bill Digest Text
- Named Entity Recognition Data: This contained all the numeric counts of recognized entity types found within the discussion utterances, and bill text data (bill version, digest, and analysis)

In addition, a default set of features remained constant throughout. The default set contains all other numeric and categorical features produced by STRAINER's Bill Discussion Feature Extractor. This included vote data committee data and Bill type data. For the TF-IDF features, we also varied whether or not to filter stopwords from the text features (Bill Text, Bill Digest, Bill Subject, Bill Analysis).

Abbreviation	Description
BT	Bill Text Features
BA	Bill Analysis Features
NER	Names Entity Features
Р	Participation Feature
D	Default Features
NS	No Stopwords Removed

 Table 6.1: Feature Abbreviations

6.3.1 Results Analysis

Varying the feature sets used resulted in 122 different models tested. We split these results into two sections. The first section contains the results for the *Predictors* trained on articles-labeled data. The second contains the results for those trained on rules-labeled data. For the models with the highest recall for each labeling method, we inspected the top features by model weight, to gain some insight into what may have caused these models to outperform the others.

For each model configuration, we recorded the type of machine learning model used (Linear Regression or Support Vector Machine), the feature sets given to the model and the model's precision, recall and F1, F2 and F3-scores.

6.3.1.1 Articles-Based Labeling Results

Table 6.2 presents the top ten predictor configurations trained on articles-labeled data and, for comparison purposes the worst overall predictor in terms of recall. The table is sorted in order of descending recall score. The full set of model results are located in Appendix A. Feature categories have been abbreviated according to the format in table 6.1.

Model Type	Features Used	Precision	Recall	Selectivity	F1
LR	BT,NS,D	0.700000	0.807692	0.581717	0.750000
LR	BT,P,NS,D	0.701923	0.802198	0.576177	0.748718
LR	NER,BT,NS,D	0.647059	0.785714	0.612188	0.709677
LR	NER,BT,P,NS,D	0.648402	0.780220	0.606648	0.708229
LR	BA,P,NS,D	0.657143	0.758242	0.581717	0.704082
SVM	BT,NS,D	0.736559	0.752747	0.515235	0.744565
LR	BT,D	0.702564	0.752747	0.540166	0.726790
LR	BT,P,D	0.709845	0.752747	0.534626	0.730667
SVM	BT,P,NS,D	0.758427	0.741758	0.493075	0.750000
SVM	BT,BA,D	0.710526	0.741758	0.526316	0.725806
SVM	BA,P,NS,D	0.731544	0.598901	0.412742	0.658610

 Table 6.2: Articles-Labeled Results

The best performing model for articles-labeled data was a linear regression model, utilizing only the default features and the bill text features unfiltered by stopwords. This model achieved a recall score of 80.77% with a precision of 70%. Table 6.3 represents a selection of the highest-weighted features for this model. The top feature by far was *discussion_word_count*. Since this is an article-labeled *Predictor*, this implies that there may be a relationship between the length of a bill discussion and the presence of proximal news articles. In other words, the longer the discussion the more likely a news article was written concerning the bill.

Feature Name	Feature Weight				
discussion_word_count	7.155854				
"act"	2.458883				
"instruction"	2.382595				
"offenders"	2.000567				
"lupus"	1.886624				
"zoning"	1.860046				
"pupil"	1.834292				
"cannabis"	1.808412				
committee_name_Senate Standing Committee on Labor and	1.717110				
Industrial Relations					
"funding"	1.702790				
"beaches"	1.664979				
"charges"	1.644414				
"law"	1.633917				
committee_name_Assembly Standing Committee on Jobs,	1.616134				
Economic Development, and the Economy					
committee_name_Assembly Standing Committee on Gov-	1.614255				
ernmental Organization					
"and"	1.606875				
"costs"	1.557263				
"promise"	1.545135				
"vehicles"	1.541647				
Continued on next page					

Table 6.3: Highest Weighted Features From the Logistic Regression Modelwith the Highest Recall on Articles- Based Labeled Data

Feature Name	Feature Wight
"reproductive"	1.539838
"transparency"	1.529751
"farmworker"	1.504687
"attorneys"	1.503470
"employers"	1.493098
committee_name_Assembly Standing Committee on Public	1.488074
Employees, Retirement, and Social Security	
"American"	1.482353
"residential"	1.481125
$committee_name_Senate\ Standing\ Committee\ on\ Appropri-$	1.480209
ations	
"protection"	1.440802
"voting"	1.435741

Table 6.3 – continued from previous page

6.3.1.2 Rules-Based Labeling Results

Table 6.4 contains the evaluation results for models trained rules-labeled data. The fields within this table are the same as in table 6.2.

Model Type	Features Used	Precision	Recall	Selectivity	F1
SVM	NER,P,D	0.707246	0.829932	0.583756	0.763693
LR	BT,BA,P,NS,D	0.679887	0.816327	0.597293	0.741886
LR	BT,BA,NS,D	0.675141	0.812925	0.598985	0.737654
LR	NER,BT,BA,P,NS,D	0.686217	0.795918	0.576988	0.737008
SVM	BA,P,D	0.723602	0.792517	0.544839	0.756494
LR	NER,BT,BA,NS,D	0.682353	0.789116	0.575296	0.731861
SVM	BA,D	0.718266	0.789116	0.546531	0.752026
SVM	NER,BT,P,D	0.731861	0.789116	0.536379	0.759411
SVM	BT,P,D	0.715170	0.785714	0.546531	0.748784
SVM	BT,D	0.705521	0.782313	0.551607	0.741935
LR	NER,P,D	0.777778	0.666667	0.426396	0.717949

Table 6.4: Rules-Labeled Results

In contrast to the results of the articles labeled data, the best performing model was a SVM model. This model achieved a recall score of 82.99%. In addition, this high scoring model did not use any of the TF-IDF text features. Instead, it was trained with the Named Entity Recognition count features, the participation feature and the defaults numeric features. To more fully understand the importance of each of the features chosen, table 6.5 reports a selection of the highest weighted features of this model. The complete list of the top fifty features by weight is located in Appendix B.

Table 6.5: Highest Weighted Features From SVM Model With the HighestRecall on Rules-Based Labeled Data

Feature Name	Feature Weight
discussion_word_count	1.591007
num_rep_abstain	1.318801
num_rep_voted_against	1.052736
num_voted_against	0.868066
committee_name_Senate Standing Committee on Judiciary	0.732448
num_abstain	0.714020
num_dem_voted_against	0.708172
committee_name_Assembly Standing Committee on Public	0.646757
Safety	
committee_name_Assembly Floor	0.606020
committee_name_Senate Standing Committee on Health	0.518896
committee_name_Assembly Standing Committee on Health	0.472075
committee_name_Assembly Standing Committee on Natural	0.360336
Resources	
committee_name_Assembly Standing Committee on Busi-	0.262507
ness and Professions	
committee_name_Assembly Standing Committee on Educa-	0.259165
tion	
committee_name_Assembly Standing Committee on Veter-	0.255960
ans Affairs	
committee_name_Senate Standing Committee on Govern-	0.243455
mental Organization	
Cont	inued on next page

Feature Name	Feature Weight			
committee_name_Assembly Standing Committee on Envi-	0.241544			
ronmental Safety and Toxic Materials				
committee_name_Assembly Standing Committee on Water,	0.237337			
Parks, and Wildlife				
committee_name_Assembly Standing Committee on Labor	0.235604			
and Employment				
committee_name_Assembly Standing Committee on Local	0.212506			
Government				
committee_name_Assembly Standing Committee on Hous-	0.196007			
ing and Community Development				
committee_name_Senate Standing Committee on Energy,	0.185062			
Utilities and Communications				
committee_name_Assembly Standing Committee on Trans-	0.176703			
portation				
avg_words_per_speaker	0.155858			
committee_name_Assembly Standing Committee on Gov-	0.150718			
ernmental Organization				
committee_name_Assembly Standing Committee on Rev-	0.148544			
enue and Taxation				
committee_name_Senate Standing Committee on Public	0.146383			
Safety				
committee_name_Senate Standing Committee on Banking	0.146116			
and Financial Institutions				
committee_name_Senate Floor	0.135126			
Continued on next pag				

Table 6.5 - continued from previous page

Feature Name	Feature Weight
committee_name_Senate Standing Committee on Human	0.124582
Services	

Table 6.5 – continued from previous page

This SVM model also weights *discussion_word_count* highest. The next highest weighted features appear to be those describing the vote that took place during the discussion, followed by one-hot encoded committees.

In addition to overall recall for rules based labeling, we also calculated the recall of these models with respect to each rule. Table 6.7 contains the per-rule recalls for each rule for the 10 models with the best overall recall. The rule recall abbreviations are defined in table 6.6.

- •	\mathbf{U}	tuic rectair report viation	11
	Abbreviation	Description	
	R A	Recall - High Attendance	
	R AA	Recall - Author Against	
	R FB	Recall - First Bill	
	R FF	Recall - Flip Flop	
	R AP	Recall - Against Party	

Table 6.6: Per-Rule Recall Abbreviations

Model	Features	R A	R AA	R FB	R FF	R AP
SVM	NER,P,D	0.878788	NA_NR	0.434783	1.000000	0.860577
LR	BT,BA,P,NS,D	0.863636	NA_NR	0.913043	0.636364	0.807692
LR	BT,BA,NS,D	0.848485	NA_NR	0.913043	0.636364	0.807692
LR	NER,BT,BA,P,NS,D	0.848485	NA_NR	0.869565	0.727273	0.783654
SVM	BA,P,D	0.757576	NA_NR	0.478261	0.727273	0.855769
LR	NER,BT,BA,NS,D	0.833333	NA_NR	0.869565	0.727273	0.778846
SVM	BA,D	0.742424	NA_NR	0.478261	0.727273	0.855769
SVM	NER,BT,P,D	0.772727	NA_NR	0.608696	0.727273	0.826923
SVM	BT,P,D	0.712121	NA_NR	0.652174	0.727273	0.836538
SVM	BT,D	0.696970	NA_NR	0.652174	0.636364	0.841346

 Table 6.7: Rules-Labeled Per-Rule Recall Results

The "Author Against" rule recall was not calculated since no instances of that rule occurred in the random test set (this rule was only triggered 3 times during all of the 2017-2018 session discussions). Recall for "High Attendance" mirrored the overall recall of the models. Recall for the "First Bill" rule experienced the highest variation in recall performance with the best overall model only capturing 43.48% of the discussions which triggered this rule.

6.3.2 Analysis

The results of our models confirm that STRAINER can consistently perform better than random at identifying labeled newsworthy bill discussions. It can successfully identify more than 75% of discussions associated with human generated news articles and over 80% of articles associated with newsworthiness rules. Comparing the models across labeling techniques yields some interesting facts and may highlight some limitations regarding our event predictors.

Looking at the top performing models for each labeling method, we can see some similarities. Both rank *discussion_word_count* as the highest weighted feature within the models. This makes some intuitive sense for the rules-labeled model. One of the business rules relates directly to the number of people who attended the meeting (unusually high participation). It stands to reason that if an unusually high number of people spoke, then more total words are likely to have been spoken. Thus, it is logical to assume that the machine learning model would attempt to learn this rule by weighting *discussion_word_count* higher. More interesting is the fact that the best articles-labeled trained model also gave *discussion_word_count* the highest weight. This lends credence to the theory that the more people discuss a bill in hearings, the more newsworthy that bill will be.

However, *discussion_word_count* seemed to be the only significant common feature across labeling methodologies. The best rules-labeled method's next most heavily weighted set of features were related to the voting of the legislators. In contrast, the best articles-labeled model's most heavily weighted features were made up of individual words from bill text features and one-hot encoded committee names. This may indicate over-fitting to the labeling methodology. The rules-labeled model may favor voting features more, since two of the rules concern voting patterns (Vote against party and vote against own bill). The reliance of the article-labeled model on bill texts reflects the fact that the proximal news articles mentioned the bill and were not necessarily about the happenings of the bill discussion.

Chapter 7

CONCLUSION AND FUTURE WORK

STRAINER is a machine learning pipeline that can filter out newsworthy legislative event data from non-newsworthy data. Our hope is that STRAINER, in conjunction with AI4Reporters, can be a useful aid for both journalists and the general public. In order accomplish this, we chose specific criteria that would be used to define and locate what is newsworthy. We then devised a program that would be able to filter through massive amounts of data to detect those criteria/features. We developed ways of labeling bill discussion events with respect to newsworthiness. We generated a dataset of bill discussions temporally linked to human generated news articles. Finally, we were able to train predictive models which can capture over 75% of positively-labeled newsworthy events. From our preliminary testing, we can say that STRAINER works. It successfully acts as an expandable filter pipeline for legislative event data.

7.1 Future Work

Our work has been successful in helping to predict newsworthy events, however, there is more work that can be done in this area. This first implementation of STRAINER only scratches the surface of what information is possible to mine from legislative data. Further work is necessary to fully mitigate the decline in legislative oversight. We present the following as areas of improvement and future research.

7.1.1 Training On a Union of Labelers

During the course of our predictor model training we did not train a model using the union of labels from both the Rules-Labeler and Articles-Labeler since the labeling domains overlap only slightly. Still, it would be interesting to see if a general model could be trained to identify discussions for both labels at the same time.

7.1.2 Improved Data Labeling

The major difficulty encountered during the design and implementation of STRAINER was the lack of truly ground truth labels regarding newsworthiness. The solutions, which we ended up using, found only potential positive examples of newsworthiness. This forced our prediction model evaluation to be biased towards recall. Better forms of labeling are needed to alleviate this. One possible method would be to use active feedback from AI4Reporters operating on live legislative data. The current version of STRAINER could feed current legislative event data to AI4Reporters, which would in turn, generate tip-sheets and present these tipsheets to reporters and newsrooms. Each newsroom could then give positive or negative feedback regarding the newsworthiness of the event described by the tipsheet. By aggregating these ratings, we could build a labeled set of events that are both positively and negatively labeled.

7.1.3 STRAINER as a Ranking System

This iteration of STRAINER focused on answering a binary question, namely, is an event newsworthy or not? However, STRAINER can be easily pivoted to answer the question, how newsworthy is an event?. To accomplish this would require the relabeling of the data and retraining a regression model. The rules-based labeler would record the number of rules triggered rather than whether a rule was triggered. The articles-based labeler would use the number of articles found, instead of the presence of articles. This would allow STRAINER to serve as a ranking system for AI4Reporters. Events with a higher predicted newsworthiness rating would be given precedence over those with lower ratings.

7.1.4 Expanding STRAINER Beyond Bill Discussions

During the development of AI4Reporters, when discussing the system with members of the journalism community, we realized that many other legislative stories, beyond the day to day goings-on within the legislature, are often newsworthy. Such events are not tied to specific timed events. News stories often focus on individual legislators, their voting history and affiliations. Others focus on particular industries and bills pertaining to those industries. All of this data is present within the Digital Democracy database and would likely be present in a new live version. Within AI4Reporters, we have already begun implementing phenoms designed to extract analytical data across an entire session year for legislators. Adding feature data aggregators, extractors and predictors to STRAINER for analytical phenomena would pave the way for ranking these events' newsworthiness.

7.1.5 Testing STRAINER Across Multiple Session Years

We only tested STRAINER on one session year's worth of data. As a result some models may have chosen higher weightings based upon political events that came to light only during that particular session year. For example, the discussions that took place in the Assembly Standing Committee on Public Safety was given a high weight in the best rules-based SVM model. However, During the 2017-2018 session year, the California legislature was actively debating high profile laws regarding police body cams[50]. As a result, this committee's hearings may have been given higher newsworthiness. Increasing the data range could enhance the general predictive capacity of STRAINER and reduce over-fitting to particular issues that may only be relevant during the political climate of one session year.

7.1.6 Testing STRAINER Prediction Using Non-iid Machine Learning Methods

When considering the newsworthiness of bill discussions, one must keep in mind that some discussions can be linked together in several different ways. The most obvious link is by the bill. A bill often is discussed over the course of many different hearings, across a number of committees. Also, legislators and lobbyists speak at multiple hearings. Lobbyists especially speak at hearings that usually concern their lobbying interests. These facts illustrate that bill discussions are not independent events. The newsworthiness of one may be successfully predicted by looking at the newsworthiness of previous discussions. However, STRAINER's predictors do not currently take this into account. Both of STRAINER's bill discussion predictors assume that the bill discussions follow the i.i.d. principle. They assume that the newsworthiness of discussions are independent and identically distributed. In reality, this is not the case. Machine learning methods do exist which do not make the i.i.d assumption. One such example is probabilistic soft logic (PSL) models. These models allow for the creation of a weighted set of rules that govern the relationship of one sample's prediction to another [1]. It would be interesting to see the performance of a predictor that could take advantage of the newsworthiness of previous discussions.

BIBLIOGRAPHY

- E. Augustine. Getting started with psl, Jul 2018. https://psl.linqs.org/blog/2018/07/15/getting-started-with-psl.html.
- S. Baccianella, A. Esuli, and F. Sebastiani. SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *Proceedings* of the Seventh conference on International Language Resources and Evaluation (LREC'10), Valletta, Malta, May 2010. European Languages Resources Association (ELRA). http://www.lrec-conf.org/proceedings/lrec2010/pdf/769_Paper.pdf.
- [3] Ballotopedia. California state legislature. https://ballotpedia.org/California_ State_Legislature.
- [4] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. Classification and regression trees, 1984. https://pdfs.semanticscholar.org/8017/ 699564136f93af21575810d557dba1ee6fc6.pdf?_ ga=2.56049067.1492467296.1557960653-22507109.1548713759&_ gac=1.11690752.1557981188.
 EAIaIQobChMIILabmZyf4gIVB8pkCh1XMgUmEAAYASAAEgJ_cfD_BwE.
- [5] A. Budhwar, T. Kuboi, A. Dekhtyar, and F. Khosmood. predicting the vote using legislative speech. In M. Janssen, S. A. Chun, and V. Weerakkody, editors, Proceedings of the 19th Annual International Conference on Digital Government Research: Governance in the Data Age, DG.O 2018, Delft, The Netherlands, May 30 - June 01, 2018, pages 35:1–35:10. ACM, 2018.

- [6] Z. Cheng, J. Caverlee, and K. Lee. You are where you tweet: A content-based approach to geo-locating twitter users. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pages 759–768, New York, NY, USA, 2010. ACM. http://doi.acm.org/10.1145/1871437.1871535.
- J. P. C. Chiu and E. Nichols. Named entity recognition with bidirectional lstm-cnns. Transactions of the Association for Computational Linguistics, 4:357–370, 2016.
- [8] M. Copeland. The difference between ai, machine learning, and deep learning? — nvidia blog. The Official NVIDIA Blog, Jul 2016. https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificialintelligence-machine-learning-deep-learning-ai/.
- C. Cortes and V. N. Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.
- [10] J. Cramer. The origins of logistic regression, 2002. https://pure.uva.nl/ws/files/2172772/56_02119.pdf.
- [11] M. Davidsson. Explorations in named entity recognition, and was eleanor roosevelt right?, Jul 2019. https://towardsdatascience.com/explorations-innamed-entity-recognition-and-was-eleanor-roosevelt-right-671271117218.
- [12] F. Dernoncourt, J. Y. Lee, and P. Szolovits. Neuroner: an easy-to-use program for named-entity recognition based on neural networks. In *EMNLP*, 2017.
- P. M. Domingos. A few useful things to know about machine learning. *Commun. ACM*, 55:78–87, 2012. https://homes.cs.washington.edu/~pedrod/papers/cacm12.pdf.

- [14] J. Dorroh. Statehouse exodus: Ajr's latest survey of the nation's state capitols finds a dramatic decrease in the number of newspaper reporters covering state government full time. a handful of digital news outlets are springing up to fill the breach. when will these efforts be enough to compensate for the loss of the newspaper watchdogs? *American Journalism Review*, 2009. https://link.galegroup.com/apps/doc/A199122138/AONE?u=calpolyw_ csu&sid=AONE&xid=959059e6.
- [15] C. N. dos Santos and M. A. de C. Gatti. Deep convolutional neural networks for sentiment analysis of short texts. In *COLING*, 2014. https://www.aclweb.org/anthology/C14-1008.
- [16] H. Drucker, C. J. C. Burges, L. Kaufman, A. J. Smola, and V. Vapnik. Support vector regression machines. In NIPS, 1996. https://pdfs.semanticscholar. org/e52f/b14e4beccc5e88a33c1fe5c7d6e780831ae1.pdf?_ ga=2.167571774.1492467296.1557960653-22507109.1548713759&_ gac=1.41600662.1557981188.
 EAIaIQobChMIlLabmZyf4gIVB8pkCh1XMgUmEAAYASAAEgJ_cfD_BwE.
- [17] D. Faggella. What is machine learning? *Emerj*, February 2019. https://emerj.com/ai-glossary-terms/what-is-machine-learning/.
- [18] J. Galtung and M. H. Ruge. The structure of foreign news: The presentation of the congo, cuba and cyprus crises in four norwegian newspapers. Journal of Peace Research, 2(1):64–90, 1965. https://doi.org/10.1177/002234336500200104.
- [19] E. Grieco. Fast facts about the newspaper industry's financial struggles as mcclatchy files for bankruptcy, May 2020.

- [20] K. Grimes. California legislature introduces 2,576 new bills for 2019. California Globe, Feb 2019. https://californiaglobe.com/legislature/californialegislature-introduces-2576-new-bills-for-2019/.
- [21] D. J. Hand and K. Yu. Idiot's bayes—not so stupid after all? International Statistical Review, 69(3):385–398, 2001. https: //onlinelibrary.wiley.com/doi/abs/10.1111/j.1751-5823.2001.tb00465.x.
- [22] T. Harcup and D. O'Neill. What is news? galtung and ruge revisited. Journalism Studies - JOURNAL STUD, 2:261–280, 05 2001. https://doi.org/10.1080/14616700118449.
- [23] T. Harcup and D. O'Neill. What is news? Journalism Studies, 18(12):1470–1488, 2017. https://doi.org/10.1080/1461670X.2016.1150193.
- [24] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant. Applied logistic regression, volume 398. John Wiley & Sons, 2013.
- [25] C. L. Information. Overview of legislative process. http://www.leginfo.ca.gov/bil2lawx.html.
- [26] R. Judd. When local media struggles, so does our democracy, Mar 2018. https://www.seattletimes.com/pacific-nw-magazine/when-local-mediastruggles-so-does-our-democracy/.
- [27] A. Klimashevskaia, R. Gadgil, T. Gerrity, F. Khosmood, C. Gütl, and P. Howe. Automatic news article generation from legislative proceedings: A phenom-based approach. In SLSP, 2021. https://doi.org/10.1007/978-3-030-89579-2_2.
- [28] B. Kovach and T. Rosenstiel. Are watchdogs an endangered species? Columbia Journalism Review, 40:50, 2019/5/21/ 2001.

https://link.galegroup.com/apps/doc/A75084882/AONE?u=calpolyw_ csu&sid=AONE&xid=5deffe39.

- [29] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer. Neural architectures for named entity recognition. In *HLT-NAACL*, 2016. https://arxiv.org/abs/1603.01360.
- [30] M. Lee and S. Hoffstaetter. Pytesseract. https://pypi.org/project/pytesseract/.
- [31] A. Lex, N. Gehlenborg, H. Strobelt, R. Vuillemot, and H. Pfister. Upset: Visualization of intersecting sets. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1983–1992, 2014.
- [32] K. E. Matsa and J. L. Boyles. America's shifting statehouse press. Pew Research Center's Journalism Project, Jul 2014. https: //www.journalism.org/2014/07/10/americas-shifting-statehouse-press/.
- [33] Historical newspapers from 1700s-2000s. https://www.newspapers.com/.
- [34] T. D. Nies, E. D'heer, S. Coppens, D. V. Deursen, E. Mannens, S. Paulussen, and R. V. de Walle. Bringing newsworthiness into the 21st century. In *WoLE@ISWC*, 2012.
- [35] C. S. of State. Lobbying filing requirements. https://www.sos.ca.gov/campaign-lobbying/lobbying-disclosurerequirements/lobbying-filing-requirements.
- [36] T. Park and G. Casella. The bayesian lasso, 2008. https://www.semanticscholar.org/paper/The-Bayesian-Lasso-Park-Casella/5b51ce3bbe7791e1533be7d4d76b2452bf043954.
- [37] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel,M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas,

A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay.
Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

- [38] J. Peiser. The rise of the robot reporter. The New York Times, Feb 2019. https://www.nytimes.com/2019/02/05/business/media/artificialintelligence-journalism-robots.html.
- [39] C. Poly. Cal Poly Github. http://www.github.com/CalPoly.
- [40] J. F. Puget. What is machine learning? What Is Machine Learning? (IT Best Kept Secret Is Optimization), May 2016. https://www.ibm.com/developerworks/community/blogs/jfp/entry/What_ Is_Machine_Learning?lang=en.
- [41] M. E. Rubado and J. T. Jennings. Political consequences of the endangered local watchdog: Newspaper decline and mayoral elections in the united states. https://doi.org/10.1177/1078087419838058.
- [42] S. R. Safavian and D. A. Landgrebe. A survey of decision tree classifier methodology. *IEEE Trans. Systems, Man, and Cybernetics*, 21:660–674, 1991. https://www.semanticscholar.org/paper/A-survey-of-decision-treeclassifier-methodology-Safavian-Landgrebe/973cb57bd2f4b67c99aea0bbdb74c41a29bf6d2a.
- [43] S. H. @sarahsholder Feed Sarah Holder and CityLab. Shrinking newsrooms means less local political engagement. *CityLab*, Apr 2019. https://www.citylab.com/life/2019/04/local-news-decline-journalist-newsdesert-california-data/586759/.
- [44] S. Schulhofer-Wohl and M. Garrido. Do newspapers matter? short-run and long-run evidence from the closure of the cincinnati post. *Journal of Media*

Economics, 26(2):60–81, 2013. https://doi.org/10.1080/08997764.2013.785553.

- [45] scikit learn.org. 1.5. stochastic gradient descent. https://scikit-learn.org/stable/modules/sgd.html#sgd.
- [46] S. slyngbae Lyngbaek. Spork: A summarization pipeline for online repositories of knowledge. Master's thesis, California Polytechnic State University, 2013. https://digitalcommons.calpoly.edu/theses/1036/.
- [47] J. F. Staab. The role of news factors in news selection: A theoretical reconsideration. European Journal of Communication, 5(4):423–443, 1990. https://doi.org/10.1177/0267323190005004003.
- [48] J. M. Stanton. Galton, pearson, and the peas: A brief history of linear regression for statistics instructors. *Journal of Statistics Education*, 9(3):null, 2001. https://doi.org/10.1080/10691898.2001.11910537.
- [49] D. D. L. Team. Digital democracy. http://www.iatpp.calpoly.edu/projects/digitaldemocracy.asp.
- [50] ThePolicingProject. California body camera policy. https://www.policingproject.org/california-body-camera-policy.
- [51] A. J. Wang. Tongs: Tldr; opinion network guide system. Master's thesis, California Polytechnic State University, 2017. https://digitalcommons.calpoly.edu/theses/1798/.
- [52] C. Wu. Skewer: Sentiment knowledge extraction with entity recognition. Master's thesis, California Polytechnic State University, 2016. https://doi.org/10.15368/theses.2016.83.

[53] J. Wu. Whisk: Web hosted information into summarized knowledge. Master's thesis, California Polytechnic State University, 2016. https://doi.org/10.15368/theses.2016.118.

APPENDICES

Appendix A

MODEL RESULTS

Table A.1: Articles Labeled Results

Model	Features	Prec	Rec	Sel	F1
LR	BT,NS,D	0.700000	0.807692	0.581717	0.750000
LR	BT,P,NS,D	0.701923	0.802198	0.576177	0.748718
LR	NER,BT,NS,D	0.647059	0.785714	0.612188	0.709677
LR	NER,BT,P,NS,D	0.648402	0.780220	0.606648	0.708229
LR	BA,P,NS,D	0.657143	0.758242	0.581717	0.704082
SVM	BT,NS,D	0.736559	0.752747	0.515235	0.744565
LR	BT,D	0.702564	0.752747	0.540166	0.726790
LR	BT,P,D	0.709845	0.752747	0.534626	0.730667
SVM	BT,P,NS,D	0.758427	0.741758	0.493075	0.750000
SVM	BT,BA,D	0.710526	0.741758	0.526316	0.725806
SVM	NER,BT,NS,D	0.736264	0.736264	0.504155	0.736264
SVM	BT,P,D	0.720430	0.736264	0.515235	0.728261
SVM	BT,BA,NS,D	0.751412	0.730769	0.490305	0.740947
SVM	BT,BA,P,NS,D	0.760000	0.730769	0.484765	0.745098
LR	NER,BA,D	0.630332	0.730769	0.584488	0.676845
LR	NER,BA,P,D	0.633333	0.730769	0.581717	0.678571

Model	Features	Prec	Rec	Sel	F1
LR	NER,BT,D	0.693122	0.719780	0.523546	0.706199
LR	NER,BT,P,D	0.693122	0.719780	0.523546	0.706199
SVM	NER,BT,P,NS,D	0.769231	0.714286	0.468144	0.740741
SVM	NER,BT,D	0.718232	0.714286	0.501385	0.716253
SVM	BT,BA,P,D	0.706522	0.714286	0.509695	0.710383
SVM	NER,BT,BA,NS,D	0.750000	0.708791	0.476454	0.728814
LR	NER,BT,BA,NS,D	0.724719	0.708791	0.493075	0.716667
LR	NER,BA,P,NS,D	0.658163	0.708791	0.542936	0.682540
LR	NER,BA,NS,D	0.649746	0.703297	0.545706	0.675462
SVM	NER,BT,BA,P,D	0.711111	0.703297	0.498615	0.707182
SVM	NER,BT,P,D	0.721591	0.697802	0.487535	0.709497
SVM	BA,NS,D	0.703911	0.692308	0.495845	0.698061
SVM	NER,BA,P,NS,D	0.707865	0.692308	0.493075	0.700000
SVM	BT,D	0.732558	0.692308	0.476454	0.711864
SVM	BA,D	0.692308	0.692308	0.504155	0.692308
SVM	BA,P,D	0.692308	0.692308	0.504155	0.692308
SVM	NER,BT,BA,D	0.720000	0.692308	0.484765	0.705882
SVM	NER,BA,P,D	0.688525	0.692308	0.506925	0.690411
SVM	,D	0.621891	0.686813	0.556787	0.652742
SVM	P,D	0.618812	0.686813	0.559557	0.651042
LR	BT,BA,P,NS,D	0.742515	0.681319	0.462604	0.710602
SVM	NER,BT,BA,P,NS,D	0.765432	0.681319	0.448753	0.720930
LR	BT,BA,P,D	0.729412	0.681319	0.470914	0.704545
SVM	NER,BA,NS,D	0.706897	0.675824	0.481994	0.691011

Model	Features	Prec	Rec	Sel	F1
LR	BT,BA,NS,D	0.739394	0.670330	0.457064	0.703170
LR	BT,BA,D	0.726190	0.670330	0.465374	0.697143
LR	NER,BT,BA,P,NS,D	0.764331	0.659341	0.434903	0.707965
SVM	NER,P,D	0.674157	0.659341	0.493075	0.666667
LR	NER,D	0.672316	0.653846	0.490305	0.662953
LR	NER,P,D	0.672316	0.653846	0.490305	0.662953
SVM	NER,D	0.682081	0.648352	0.479224	0.664789
SVM	NER,BA,D	0.710843	0.648352	0.459834	0.678161
LR	NER,BT,BA,D	0.751592	0.648352	0.434903	0.696165
LR	NER,BT,BA,P,D	0.746835	0.648352	0.437673	0.694118
LR	P,D	0.639344	0.642857	0.506925	0.641096
LR	,D	0.637363	0.637363	0.504155	0.637363
LR	BA,NS,D	0.768707	0.620879	0.407202	0.686930
LR	BA,P,D	0.758621	0.604396	0.401662	0.672783
SVM	BA,P,NS,D	0.731544	0.598901	0.412742	0.658610
LR	BA,D	0.751724	0.598901	0.401662	0.666667

 Table A.2: Rules Labeled Results

Model	Features	Prec	Rec	Sel	F1
SVM	NER,P,D	0.707246	0.829932	0.583756	0.763693
LR	BT,BA,P,NS,D	0.679887	0.816327	0.597293	0.741886

Model	Features	Prec	Rec	Sel	F1
LR	BT,BA,NS,D	0.675141	0.812925	0.598985	0.737654
LR	NER,BT,BA,P,NS,D	0.686217	0.795918	0.576988	0.737008
SVM	BA,P,D	0.723602	0.792517	0.544839	0.756494
LR	NER,BT,BA,NS,D	0.682353	0.789116	0.575296	0.731861
SVM	BA,D	0.718266	0.789116	0.546531	0.752026
SVM	NER,BT,P,D	0.731861	0.789116	0.536379	0.759411
SVM	BT,P,D	0.715170	0.785714	0.546531	0.748784
SVM	BT,D	0.705521	0.782313	0.551607	0.741935
SVM	P,D	0.746753	0.782313	0.521151	0.764120
SVM	BA,NS,D	0.743506	0.778912	0.521151	0.760797
SVM	BA,P,NS,D	0.748366	0.778912	0.517766	0.763333
SVM	NER,D	0.708978	0.778912	0.546531	0.742301
SVM	NER,BT,D	0.722397	0.778912	0.536379	0.749591
SVM	NER,BT,BA,P,NS,D	0.733119	0.775510	0.526227	0.753719
LR	NER,BT,D	0.705882	0.775510	0.546531	0.739060
LR	NER,BT,P,D	0.707165	0.772109	0.543147	0.738211
SVM	NER,BT,BA,D	0.728155	0.765306	0.522843	0.746269
SVM	NER,BT,BA,P,D	0.726384	0.758503	0.519459	0.742097
SVM	,D	0.762887	0.755102	0.492386	0.758974
SVM	BT,P,NS,D	0.763066	0.744898	0.485618	0.753873
LR	BT,P,NS,D	0.742373	0.744898	0.499154	0.743633
SVM	NER,BA,P,NS,D	0.757785	0.744898	0.489002	0.751286
SVM	NER,BT,BA,NS,D	0.741497	0.741497	0.497462	0.741497
SVM	BT,BA,P,NS,D	0.731544	0.741497	0.504230	0.736486

Model	Features	Prec	Rec	Sel	F1
SVM	NER,BA,NS,D	0.756098	0.738095	0.485618	0.746988
LR	NER,BA,P,NS,D	0.766784	0.738095	0.478849	0.752166
SVM	NER,BT,P,NS,D	0.744828	0.734694	0.490694	0.739726
SVM	BT,NS,D	0.757042	0.731293	0.480541	0.743945
LR	NER,BA,NS,D	0.767857	0.731293	0.473773	0.749129
SVM	BT,BA,D	0.730375	0.727891	0.495770	0.729131
LR	BT,NS,D	0.734483	0.724490	0.490694	0.729452
SVM	BT,BA,NS,D	0.736111	0.721088	0.487310	0.728522
SVM	NER,BA,D	0.762590	0.721088	0.470389	0.741259
SVM	NER,BA,P,D	0.759857	0.721088	0.472081	0.739965
LR	NER,BT,BA,P,D	0.741259	0.721088	0.483926	0.731034
LR	BT,P,D	0.742958	0.717687	0.480541	0.730104
LR	NER,BT,BA,D	0.742958	0.717687	0.480541	0.730104
LR	BA,P,NS,D	0.780669	0.714286	0.455161	0.746004
SVM	NER,BT,NS,D	0.760000	0.710884	0.465313	0.734622
LR	BA,NS,D	0.784906	0.707483	0.448393	0.744186
LR	BT,D	0.729825	0.707483	0.482234	0.718480
LR	BT,BA,P,D	0.742857	0.707483	0.473773	0.724739
LR	NER,BA,D	0.761029	0.704082	0.460237	0.731449
LR	NER,BA,P,D	0.763838	0.704082	0.458545	0.732743
LR	BA,D	0.792308	0.700680	0.439932	0.743682
LR	BA,P,D	0.792308	0.700680	0.439932	0.743682
LR	P,D	0.791506	0.697279	0.438240	0.741410
LR	BT,BA,D	0.737410	0.697279	0.470389	0.716783
Model	Features	Prec	Rec	Sel	F1
-------	---------------	----------	----------	----------	----------
SVM	BT,BA,P,D	0.750000	0.693878	0.460237	0.720848
LR	NER,BT,P,NS,D	0.763158	0.690476	0.450085	0.725000
LR	NER,BT,NS,D	0.756554	0.687075	0.451777	0.720143
LR	,D	0.788235	0.683673	0.431472	0.732240
LR	NER,D	0.771654	0.666667	0.429780	0.715328
LR	NER,P,D	0.777778	0.666667	0.426396	0.717949

Table A.3: Rules Labeled Models, Recall Results Per Rule

Model	Features	R A	R AA	R FB	R FF	R AP
SVM	NER,P,D	0.878788	NA_NR	0.434783	1.000000	0.860577
LR	BT,BA,P,NS,D	0.863636	NA_NR	0.913043	0.636364	0.807692
LR	BT,BA,NS,D	0.848485	NA_NR	0.913043	0.636364	0.807692
LR	NER,BT,BA,P,NS,D	0.848485	NA_NR	0.869565	0.727273	0.783654
SVM	BA,P,D	0.757576	NA_NR	0.478261	0.727273	0.855769
LR	NER,BT,BA,NS,D	0.833333	NA_NR	0.869565	0.727273	0.778846
SVM	BA,D	0.742424	NA_NR	0.478261	0.727273	0.855769
SVM	NER,BT,P,D	0.772727	NA_NR	0.608696	0.727273	0.826923
SVM	BT,P,D	0.712121	NA_NR	0.652174	0.727273	0.836538
SVM	BT,D	0.696970	NA_NR	0.652174	0.636364	0.841346
SVM	P,D	0.712121	NA_NR	0.347826	0.727273	0.865385
SVM	BA,NS,D	0.727273	NA_NR	0.521739	0.727273	0.841346
SVM	BA,P,NS,D	0.742424	NA_NR	0.521739	0.727273	0.836538
SVM	NER,D	0.833333	NA_NR	0.347826	0.818182	0.817308

Continued on next page

Model	Features	R A	R AA	R FB	R FF	R AP
SVM	NER,BT,D	0.803030	NA_NR	0.652174	0.636364	0.798077
SVM	NER,BT,BA,P,NS,D	0.833333	NA_NR	0.782609	0.727273	0.769231
LR	NER,BT,D	0.848485	NA_NR	0.826087	0.636364	0.759615
LR	NER,BT,P,D	0.848485	NA_NR	0.826087	0.636364	0.754808
SVM	NER,BT,BA,D	0.787879	NA_NR	0.652174	0.727273	0.783654
SVM	NER,BT,BA,P,D	0.787879	NA_NR	0.652174	0.727273	0.774038
SVM	,D	0.696970	NA_NR	0.347826	0.636364	0.836538
SVM	BT,P,NS,D	0.757576	NA_NR	0.739130	0.545455	0.759615
LR	BT,P,NS,D	0.757576	NA_NR	0.826087	0.545455	0.750000
SVM	NER,BA,P,NS,D	0.803030	NA_NR	0.434783	0.727273	0.774038
SVM	NER,BT,BA,NS,D	0.803030	NA_NR	0.739130	0.727273	0.735577
SVM	BT,BA,P,NS,D	0.742424	NA_NR	0.739130	0.545455	0.759615
SVM	NER,BA,NS,D	0.787879	NA_NR	0.434783	0.727273	0.769231
LR	NER,BA,P,NS,D	0.787879	NA_NR	0.565217	0.727273	0.754808
SVM	NER,BT,P,NS,D	0.803030	NA_NR	0.695652	0.636364	0.730769
SVM	BT,NS,D	0.712121	NA_NR	0.739130	0.636364	0.754808
LR	NER,BA,NS,D	0.772727	NA_NR	0.565217	0.727273	0.750000
SVM	BT,BA,D	0.681818	NA_NR	0.695652	0.636364	0.764423
LR	BT,NS,D	0.712121	NA_NR	0.826087	0.545455	0.735577
SVM	BT,BA,NS,D	0.712121	NA_NR	0.695652	0.636364	0.745192
SVM	NER,BA,D	0.787879	NA_NR	0.391304	0.727273	0.750000
SVM	NER,BA,P,D	0.772727	NA_NR	0.391304	0.727273	0.754808
LR	NER,BT,BA,P,D	0.727273	NA_NR	0.739130	0.636364	0.730769
LR	BT,P,D	0.712121	NA_NR	0.739130	0.636364	0.730769

Continued on next page

Model	Features	R A	R AA	R FB	R FF	R AP
LR	NER,BT,BA,D	0.727273	NA_NR	0.739130	0.636364	0.725962
LR	BA,P,NS,D	0.681818	NA_NR	0.565217	0.636364	0.759615
SVM	NER,BT,NS,D	0.772727	NA_NR	0.695652	0.636364	0.706731
LR	BA,NS,D	0.651515	NA_NR	0.565217	0.636364	0.759615
LR	BT,D	0.696970	NA_NR	0.739130	0.636364	0.721154
LR	BT,BA,P,D	0.742424	NA_NR	0.782609	0.636364	0.701923
LR	NER,BA,D	0.757576	NA_NR	0.521739	0.727273	0.721154
LR	NER,BA,P,D	0.757576	NA_NR	0.521739	0.727273	0.721154
LR	BA,D	0.651515	NA_NR	0.478261	0.636364	0.759615
LR	BA,P,D	0.681818	NA_NR	0.478261	0.636364	0.750000
LR	P,D	0.742424	NA_NR	0.304348	0.909091	0.735577
LR	BT,BA,D	0.712121	NA_NR	0.782609	0.636364	0.701923
SVM	BT,BA,P,D	0.651515	NA_NR	0.608696	0.636364	0.735577
LR	NER,BT,P,NS,D	0.772727	NA_NR	0.739130	0.636364	0.673077
LR	NER,BT,NS,D	0.757576	NA_NR	0.739130	0.636364	0.673077
LR	,D	0.696970	NA_NR	0.304348	0.818182	0.735577
LR	NER,D	0.787879	NA_NR	0.260870	0.818182	0.682692
LR	NER,P,D	0.803030	NA_NR	0.260870	0.818182	0.677885

Appendix B

TOP 50 FEATURE WEIGHTS FROM BEST PERFORMING MODELS

Table B.1: Top 50 Features by Weight for Regression Model With theHighest Recall on Articles-Based Labeled Data

Feature Name	Feature Weight
discussion_word_count	7.155854
andwhereas	-3.073173
committee_name_Senate Standing Committee on Budget	-2.664211
and Fiscal Review	
act	2.458883
instruction	2.382595
committee_name_Assembly Standing Committee on Envi-	-2.312958
ronmental Safety and Toxic Materials	
university	-2.201979
committee_name_Assembly Standing Committee on Veter-	-2.101556
ans Affairs	
offenders	2.000567
lupus	1.886624
management	-1.861929
committee_name_Assembly Standing Committee on Privacy	-1.861784
and Consumer Protection	
zoning	1.860046
family	-1.848161
Cont	inued on next page

Feature Name	Feature Wight
pupil	1.834292
cannabis	1.808412
authorities	-1.762962
education	-1.739032
$\operatorname{committee_name_Senate}$ Standing Committee on Labor and	1.717110
Industrial Relations	
funding	1.702790
beaches	1.664979
charges	1.644414
law	1.633917
committee_name_Assembly Standing Committee on Jobs,	1.616134
Economic Development, and the Economy	
committee_name_Assembly Standing Committee on Gov-	1.614255
ernmental Organization	
and	1.606875
committee_name_Assembly Standing Committee on Elec-	-1.561920
tions and Redistricting	
costs	1.557263
commission	-1.553450
promise	1.545135
committee_name_Assembly Standing Committee on Budget	-1.543815
vehicles	1.541647
reproductive	1.539838
privacy	-1.537758
Cont	inued on next page

Table B.1 – continued from previous page

Feature Name	Feature Wight
transparency	1.529751
2016	1.525033
farmworker	1.504687
bill	-1.503804
attorneys	1.503470
alameda	-1.497316
reports	-1.494601
employers	1.493098
committee_name_Assembly Standing Committee on Public	1.488074
Employees, Retirement, and Social Security	
american	1.482353
residential	1.481125
exemptions	-1.480575
committee_name_Senate Standing Committee on Appropri-	1.480209
ations	
protection	1.440802
hospital	-1.438678
voting	1.435741

Table B.1 – continued from previous page

Table B.2: Top 50 Features by	Weight for	Model	With the	Highest	Recall
on Rules-Based Labeled Data					

Feature Name	Feature Weight		
discussion_word_count	1.591007		
$num_rep_abstain$	1.318801		
$num_rep_voted_against$	1.052736		
num_dem_voted_for	-0.970388		
$num_voted_against$	0.868066		
num_voted_for	-0.757226		
committee_name_Senate Standing Committee on Judiciary	0.732448		
$num_abstain$	0.714020		
$num_dem_voted_against$	0.708172		
committee_name_Assembly Standing Committee on Public	0.646757		
Safety			
committee_name_Assembly Floor	0.606020		
committee_name_Senate Standing Committee on Health	0.518896		
num_rep_voted_for	-0.490057		
committee_name_Assembly Standing Committee on Judi-	-0.474289		
ciary			
committee_name_Assembly Standing Committee on Health	0.472075		
committee_name_Assembly Standing Committee on Privacy	-0.466589		
and Consumer Protection			
is_dopass_0	-0.391315		
committee_name_Assembly Standing Committee on Utili-	-0.375455		
ties and Energy			
Continued on next page			

Feature Name	Feature Wight		
committee_name_Senate Standing Committee on Elections	-0.363534		
and Constitutional Amendments			
$committee_name_Assembly\ Standing\ Committee\ on\ Natural$	0.360336		
Resources			
committee_name_Senate Standing Committee on Agricul-	-0.349942		
ture			
committee_name_Senate Standing Committee on Environ-	-0.329437		
mental Quality			
vote_result_passed	-0.325316		
committee_name_Senate Standing Committee on Trans-	-0.307210		
portation and Housing			
num_dem_abstain	-0.297714		
committee_name_Assembly Standing Committee on Higher	-0.297340		
Education			
committee_name_Assembly Standing Committee on Elec-	-0.283121		
tions and Redistricting			
committee_name_Assembly Standing Committee on Jobs,	-0.264076		
Economic Development, and the Economy			
committee_name_Assembly Standing Committee on Busi-	0.262507		
ness and Professions			
committee_name_Assembly Standing Committee on Educa-	0.259165		
tion			
committee_name_Assembly Standing Committee on Veter-	0.255960		
ans Affairs			
Continued on next page			

Table B.2 – continued from previous page

Feature Name	Feature Wight	
committee_name_Senate Standing Committee on Govern-	0.243455	
mental Organization		
committee_name_Assembly Standing Committee on Envi-	0.241544	
ronmental Safety and Toxic Materials		
committee_name_Assembly Standing Committee on Water,	0.237337	
Parks, and Wildlife		
committee_name_Assembly Standing Committee on Labor	0.235604	
and Employment		
committee_name_Assembly Standing Committee on Local	0.212506	
Government		
committee_name_Senate Standing Committee on Budget	-0.203548	
and Fiscal Review		
committee_name_Assembly Standing Committee on Hous-	0.196007	
ing and Community Development		
committee_name_Senate Standing Committee on Energy,	0.185062	
Utilities and Communications		
committee_name_Assembly Standing Committee on Trans-	0.176703	
portation		
committee_name_Senate Standing Committee on Natural	-0.169809	
Resources and Water		
avg_words_per_speaker	0.155858	
committee_name_Assembly Standing Committee on Human	-0.152450	
Services		
Continued on next page		

Table B.2 – continued from previous page

Feature Name	Feature Wight
committee_name_Assembly Standing Committee on Gov-	0.150718
ernmental Organization	
committee_name_Assembly Standing Committee on Rev-	0.148544
enue and Taxation	
committee_name_Senate Standing Committee on Public	0.146383
Safety	
committee_name_Senate Standing Committee on Banking	0.146116
and Financial Institutions	
committee_name_Senate Floor	0.135126
committee_name_Senate Standing Committee on Human	0.124582
Services	
committee_name_Senate Standing Committee on Education	-0.121308

Table B.2 – continued from previous page