PREPARATION OF UNITED STATES' STATE COURT CASES DATASET AND DERIVING AN EFFICIENT EMBEDDING FOR CASES

by

Karthik Ravi

A thesis submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Master of Science in
Computer Science

Charlotte

2019

ABSTRACT

KARTHIK RAVI. Preparation of United States' State Court Cases dataset and deriving an Efficient Embedding for Cases. (Under the direction of DR. SAMIRA SHAIKH)

Data mining techniques gained acceptance as a clear means of finding information in data. In the past, these techniques have been effectively applied to discover patterns, find correlations, extract information from unstructured data. Often overlooked is the major fact that data mining techniques were effectively applied only on scientific datasets like finance, healthcare, physics, and chemistry.

We identified study on the U.S. State Supreme Courts is significantly constrained by the lack of available data. To find an antidote to this problem, I have conducted a research to produce original dataset for every state supreme court ruling from 1953 through 2014. We have utilized dynamic textual analysis to search through the case files of millions of state supreme court decisions and extract critical information on each case. We present analysis on the case distribution, month of submitting the case, regional reporter, and legal issues being heard in front of the court.

Following the synthesis of dataset, we applied generative statistical model to cluster the cases based on word presence in the case text. Meanwhile, we have prepared a vector representation of the cases having similar characteristics based on case text using a neural network architecture.

A validation study conducted on researchers in political science prove that all the three datasets 1. Original Supreme Court Dataset 2. Vector Representation of Case Summary 3. Topic Clusters of the Cases Summary are highly structured, extremely meaningful. Hence, these datasets will offer scholars enormous possibilities to expand the knowledge of judicial politics in the American States.

DEDICATION

This work is dedicated to my mother, Sumathi Sengottiyan and my father, Ravi P.S., family members and friends who constantly motivated and gave extraordinary support throughout my master's.

I would also like to dedicate in the hope that this work in some way contribute to research conducted on the State Supreme Courts of the United States.

ACKNOWLEDGEMENTS

I would like to gratefully acknowledge various people who have been journeyed with me in recent years as I have worked in this thesis. First, I owe an enormous debt and gratitude to my family Sumathi, Ravi, Suvetha, Senthilnathan and my friends Nivetha, Lavanya and Vikram. Throughout the struggles of my thesis they have been a constant source of joy. Thank you.

I would like to thank my advisor, Dr. Samira Shaikh, for giving me an opportunity to work with her. My work would not have been possible without her constant guidance, valuable input and feedback at each and every step. I would like to thank my project committee member and friend Dr.Jason H. Windett, who has guided, financial funded and shared with me infinite questions that our research could answer.

I would also like to express my gratitude to my committee members, Dr. Jason H. Windett and Dr. Minwoo Jake Lee for showing interest in my research work and accepting to join my thesis committee as panel members. They played a vital role instructing my thesis by providing ideas and feedback that guided me towards accomplishing my goal. My sincere appreciation to the University of North Carolina at Charlotte for accepting me as a master student and for providing necessary infrastructure and support to successfully complete my master's degree with thesis.

I am very glad to work on one of the most interesting research topics and complete my master's degree with a wonderful research experience.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

DBOW	Distributed Bag Of Words.
DM	Distributed Memory.
LDA	Latent Dirichlet Allocation.
LSA	Latent Semantic Analysis.
NLP	Natural Language Processing.
PCA	Principal Component Analysis.
PLSA	Probabilistic Latent Semantic Analysis.
SGD	Stochastic Gradient Descent.
T-SNE	T-distributed Stochastic Neighbor Embedding.
TKD	Text Knowledge Discovery.
UMAP	Uniform Manifold Approximation and Projection.

CHAPTER 1: INTRODUCTION

The Chambers of the Supreme Court of the United States contain most of the history of the United States. People misleadingly believe the history of the United States has been written merely in the halls of Congress, in the Executive offices and on the battlefields. Every single decision made in the cases become a page in the history. Theodore Roosevelt said once "In not one serious study of American political life" in honor of Judge Harlan in 1902 "will if be possible to omit the immense part played by the Supreme Court in the creation, not merely the modifications, of the great policies, through and by means of which the country has moved on to her present position. . . the Judges of the Supreme Court of the land must be not only great jurists, they must be great constructive statesmen and the truth of what I say is illustrated by every study of American statesmanship" according to [1].

The Supreme Court of the United States is the highest judicial body in the U.S. for all cases and controversies arising under the Constitution or laws of the U.S. As the final arbiter of the law, the Court ensures the American people equal justice under law, and also functions as interpreter of the Constitution. The role of the Supreme Court derives from its authority to invalidate legislation or executive actions that, in the Court's judgment, conflict with the Constitution. This power of judicial review gives the court the responsibility in assuring individual rights, as well as in maintaining a Living Constitution whose provisions are continually applied to new situations.

The Supreme Court must exercise discretion in deciding which cases to hear, since more than 10,000 civil and criminal cases are filed in the Supreme Court each year from various state and federal courts. Ultimately, plenary review, with oral arguments by attorneys, is granted in about 100 cases per term. Formal written opinions are delivered in 80 to 90 cases. The Supreme Court also has original jurisdiction in a very small number of cases arising out of disputes between states or between states and the federal government. When the Supreme Court rules on a constitutional issue, that judgment is virtually final; its decisions are only rarely altered by constitutional amendment or by a new ruling of the Court.

The core important part is the role Court has played in preserving the Union, in supporting the doctrines of international law and the sanctity of treaties, within the boundaries of the Constitution, the effect in the trend of the economic, social and political developments of the United States. It becomes merely difficult to understand the decisions made by the courts just by reading through the books.

The decisions of the Supreme Court have an impact in largely on the society. For example, the Tinker v. Des Moines Independent School District (1969) held that students could not be punished for wearing black armbands to school to protest the Vietnam War. In this case, the Court held "students do not shed their rights at the schoolhouse gate". This decision has an impact on the high school students.

Judicial Review is one of the best known power of the Supreme Court. Also, one another important power is the ability of the Court to declare a Legislative or Executive act in violation of the Constitution. This is not found within the text of the Constitution itself. The court established this doctrine in the case of Marbury v. Madison (1803).

1.1 Defining (or refining) the Problem Space

The State Court of America has a treasure trove of history for the analysis of political behavior and organizations. The States Court offer a unique forum for comparative analysis owing to their broad range of social, political and organizational characteristics.

Sadly, there has long been a lack of appropriate information in the study of political dynamics in the American states. As a result of the lack of rigorous and detailed quantitative data on state policy actions and structures, the analysis of state policy lags behind similar national policy research. This is also the explanation why the trendy word "artificial intelligence" has taken a long time to enter the realm of national politics.

Last resort courts in the US states have become increasingly influential in contemporary political and political debates over the past few decades. Such courts have taken numerous key decisions that have drawn federal courts, national politicians and voters to support and animosity. Via political leaders, attorneys and voters debating the proper function and nature of these influenceful bodies, state supreme courts representation, appointment processes and organisation have become increasingly important topics of national interest and contention.

But, even though these issues are increasingly important, scholars still know relatively little about national courts, including state courts. Arguments surrounding specific candidates for justice, legal policy, judiciary legislation, and the proper role of courts are often focused on untested scientific hypotheses. There is no question that this discussion will profit from a deeper understanding of the various relationships within the wider social and political framework involving judicial decision-making, recruitment processes, organisation, effects, perceptions, and identification. And scientists have long been conscious that there is a disparity between, for instance, [2] and [3] between the value of observing state courts and how smart they are.

1.2 Accessibility Constraints in State Supreme Court of America

New growth in methods of data collection also freed up new research avenues for the state legislative bodies. Given the development, however, the evaluation of state judicial agencies tends to be constrained by the data available. Such courts have made many important decisions that have drawn federal courts, national politicians, and constituents to support and animosity.

The analysis of these supreme state tribunals is greatly hindered by software accessibility constraints. Although these entities are relevant, academics do not have the kinds of data for State courts which the analysis of federal courts takes for granted. The State Supreme Court Information Portal is the first state court repository. There are many benefits, including breadth and complexity to this list. Nonetheless, there are some major disadvantages to the server as well.

The server, in total, spans just four years, 1995-98. As such, the capabilities of researchers to analyze courts in specific social and political situations are reduced over time.

1.3 Problem Statement

We could see the lack of available structured data in a centralized source for all the cases happened in the Supreme Court in the states of America. The main goal of my research is to address this issue and develop a clean structured data of all the cases that happened in the 52 Supreme court in the United States. Through my research on building this dataset, I seek to better understand 1) what kind of data mining techniques would be appropriate for extracting appropriate information from the text body of the case 2) to what degree we have extracted useful information from the case text 3) apply cutting-edge techniques in Machine Learning on top of the Dataset to understand the relationship between multiple cases. This research seeks further knowledge in the field of Political Science by providing a data source that helps researchers in Political Science and Computer Science on understanding and conducting computational study on the cases happened in Supreme Court of United States. My research addresses several questions within this problem space, as listed in the Research Quesitons section below

1.3.1 Research Questions

- Data Mining on unstructured data is very tough. Can I develop a computational approach to extract the information on the case files?
- Upto which extend can I extract accurate meaningful data from the source ?
- To what extend Artificial Intelligence can be applied on the dataset to make transformative view on the the cases happened ?

Once the dataset is prepared, it becomes hard to believe the importance of the dataset. Hence, we have conducted the first embedding model in the world dedicated to Supreme Court Cases to define the relationship between the cases based on all the cases happened in the period of 1953 - 2014.

In the process of building a structured data set, we will be going through a lot of issues like 1. Handling the different case writing style across various states. 2. Issues with the date format of the cases 3. Identifying the Judges involved in making a decision for a case, based on the judges roaster that has the information of the Judges active time period. 4. Intelligently mapping the decision made by a Judge in each case using intelligent techniques.

CHAPTER 2: RELATED WORK

Systematic empirical data collection was a key element in furthering the judicial policy research like Harold Spaeth's most popular U.S. server [4]. Hundreds of empirical studies have been focused on the Supreme Court's rulings in recent decades, giving academics researching at the high court readily accessible and commonly understood information. The State Supreme Court Software Project generated by Paul Brace and Melinda Gann Hall [5] is the largest data collection initiative at national level. Information on state supreme Court rulings in the 50 states during their sessions of 1995 - 1999 is included in the Brace & Hall registry. The list includes over 21,000 decisions taken by more than 400 supreme tribunals. The competitive value of the information from Brace and Hall is the reliability of data on each case. Our data includes hundreds of information items in the data set on every occasion, including vital variables that only human coders may document, such as the subject matter of the conflict. A list also includes biographical information of judges from 1995-98.

Numerous essential studies are encouraged by the brace and hall information. The authors who used their information analyzed court decisions in specific fields, such as death penalty [6] and political disputes [7]. Some studied media coverage (Vining and Wilhelm 2010), incidents of salience [8], nationality [9], litigant strategies [6], dock structure [10] & [11]. The outcome of the Brace and Hall have opened up new exciting opportunities for superior state courts, as these and many others have demonstrated. Nevertheless, the time span of the brace and hall information is limited.

Our information cover just four words, so many interesting research problems can't be studied for years. For example, the capacity of data users to analyze the actions of state supreme court judges in relation to changing political circumstances over time is restricted.

The culture, philosophy and actions of state and federal political bodies, the federal judiciary, or even the state supreme tribunals themselves as well as adjustments in the social and economic conditions of states will influence the conduct of State superior tribunals. Therefore, national supreme courts can control over time a broad range of social, political and economic conditions. The Brace and Hall software time frame also restricts the estimation of these complexities.

Some researchers have created their own information to respond to questions about the supreme state courts. Due to the difficult demands of data collection, these analyses generally limit the scope to specific areas of concern [12], [13] & [14].

In accordance to Brace and Hall's person programming technique, our methodology provides advantages and disadvantages. Above all, our methodology promotes larger-scale data collection. Our initial data collection covers 16 years in all 50 countries [15]; our second data collection phase was back to 1953. Second our methodology removes the inevitable mistakes found in any plan based on human coders. Of course the limits inherent in our automated strategy are also present on our results. Most specifically, the details in the Brace and Hall data lack the depth of knowledge, such as conflict and political conclusions. The comparative advantages of our method, though, offer new ways to explore a variety of study problems.

CHAPTER 3: BACKGROUND

Data mining is a new discipline that has sprung up at the confluence of several other disciplines, stimulated chiefly by the growth of large databases according to [16]. The underlying motivation force driving data mining is to insure that these large datasets provide valuable information, but that this information is hidden and stays unnoticed within the mass of uninteresting data. That is, one searches for shocking data, which is fresh, which is unusual or interesting, and this is to be collected. The topic is therefore closely connected to an analyzes of exploratory information. Nonetheless, problems arising from the scale of datasets and concepts and resources borrowed from other fields indicate that data mining is more than just exploration.

3.1 Data Mining on Text Data

Text mining is a variant on a field called data mining, which attempts, from large databases, to find interesting trends. Text processing, also known as Smart Text Analysis, Text Data Mining, or Text Knowledge Discovery (TKD), generally involves the extraction process from unstructured texts for important and non-trivial information and knowledge. Text mining is a new interdisciplinary field focused on the collection of information, data mining and machine learning. Since the most data (more than 80%) is contained in the text, the commercial value of text mining is considered to be strong. Intelligence from many sources of information can be uncovered but unstructured texts are the greatest intelligence source readily available.

Text mining is close to data extraction, except in the case of data mining tools, but text mining may operate for unstructured or semi-structured data sets, such as e-mails, text messages, full text, and HTML files, etc. text mining. Nonetheless, up to now many R&D activities have been based on data mining using structured data. The issue of text mining is evident: natural language has been created to interact and document data for people and machines are far away from learning the natural language.

3.1.1 Steps Involved in Text Mining

The steps involved in Text mining can broken down into five stages as shown in 3.1.

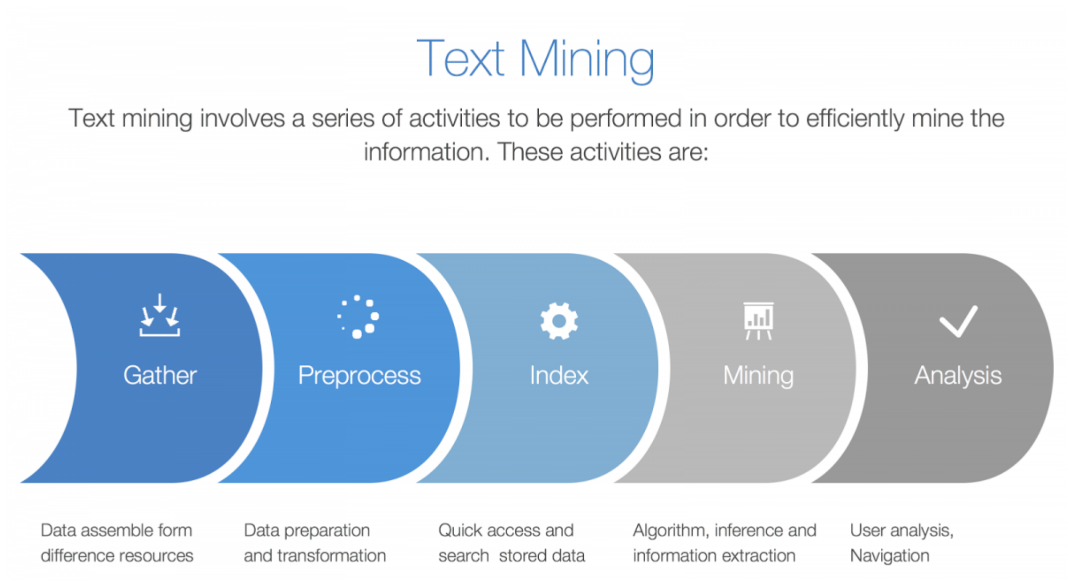

Text Mining
Text mining involves a series of activities to be performed in order to efficiently mine the information. These activities are:
Gather
Preprocess
Index
Mining
Analysis
Data assemble form difference resources
Data preparation and transformation
Quick access and search stored data
Algorithm, inference and information extraction
User analysis, Navigation

Figure 3.1: Text Mining Process.

- Gathering: data collection from various resources like databases, messages, consumer feedback, and paper documents.

- Pre-processing, such as content identification / extraction of representative features

  Text clean-up: Deleting unused or inappropriate data from websites, such as advertising.

  Tokenization: Only a machine sees a character string without the ability to identify sentences, phrases, or letters, for instance.Tokenization breaks the text into specific objects with existing white spaces and punctuations (words, sentences, etc).

Feature extraction (also called attribute selection): A quantitative measurements are a method of characterizing a document. For example, word frequency in the message, word size, syntactic information. Such features can be used for further analysis.

- Index: Create a certain index of terms, locations and numbers. The stored data can be obtained and organized easily. Mining:The text is correctly preprocessed at this point and can now be' mined.' To order to expose new knowledge, we use various methods of software discovery. These can include for instance, the connection to specific terms, the link to a dictionary or disambiguation thesaurus, and the description of links between different terms.
- Analysis: The mining process yield harsh outcomes. These have to be evaluated and viewed in order to interpret the questions that the text miner wishes to examine.

3.1.2 Application of Text Mining

According to [17], text mining application have experienced tremendous advances because of web 2.0 and social networking applications. The recent advances in hardware and software technology have lead to a number of unique scenarios where text mining algorithms are learned. Mining text data introduces an important niche in the text analytic field, and is an edited volume contributed by leading international researchers and practitioners focused on social networks and data mining. Broad range of consumers, including government agencies, research institutions and enterprises, are now using text mining software for their daily needs. Many examples of use in various fields are as follows:

- Research: e.g. exploration of data, medical / health care-it took a long time for a researcher to examine and collect useful information in the past. Such data was in some instances not even available. Text mining was structured to provide researchers with faster and more efficient data.

- Business: for instance risk management, resume sorting-large companies are using text mining to assist consumer decision making and address customer requests rapidly.

- Security: for instance, anti-terrorism-blog and other online text outlets analyzes are used for stopping web crimes and fighting fraud.

- Regular review-texts are used by e-mail web sites with reliable and effective sorting processes, e.g. spam filtering, social media data analytics. It is also used to define customer-to-product interactions or to assess consumer opinions on certain topics for social media purposes.

3.2 Deep Learning

Deep learning is a method of machine learning, which teaches algorithms how to do what naturally comes with humans invented by [18]. Profound intelligence is a crucial innovation behind driverless cars which helps them to identify or differentiate between a person and a lampstand. It is the secret to voice control in popular devices such as telephones, smartphones, TVs and hands free speakers. Late and for good reason, deep learning attracts most coverage. Results which were always impossible to achieve. In the process of a fundamental education, a computer model discovers how to recognize pictures, text or sound explicitly. Deep learning models may attain cutting-edge precision and sometimes surpass human performance. Models are equipped by a large array of marked and multi-layered information and neural network architectures.

On the context of application, Deep Learning algorithms can pull useful abstract meaning of the raw data by using an hierarchical multi-level learning approach. The higher-level are more abstract. Complex representations are learnt based on the less abstract concepts and representations in the lower level(s) of the learning hierarchy. While Deep Learning can be applied to learn from labeled data if it is available in sufficiently large amounts, it is primarily attractive for learning from large amounts of unlabeled/unsupervised data [19], [20], [21] making it attractive for extracting meaningful representations and patterns from Big Data.

3.2.1 Neural Network

Neural artificial networks can better be seen as weighted maps, where artificial neurons constitute the nodes, and the relation between neuron outputs and neuronal inputs can be seen by directional rims with weights. In the shape of a pattern and object as a function, the Artificial Neural Network generates the input signal from the outside universe. The x(n) labels for each n-number of outputs are then mathematically defined. The resulting weights (these weights are the information used for artificial neural networks to solve a particular problem) are then combined by all outputs. In general, such weights are the frequency of interconnection in the artificial neural network between neurons. All the weighted outputs (and another artificial neuron) are listed in the processing system. If the weighted sum is zero, a bias is applied to insure that the performance is not null or to scale up to the device answer. Bias always has the weight and weight equal to 1. Bias. Here the maximum input weighted can be between 0 and positive. A certain minimum value is benchmarked so that the answer stays within the optimal range. Then the weighted outputs are summarized by the activation function. In general, the activation mechanism is the collection of transfer features used to achieve the necessary efficiency. The trigger method has different flavors, but mostly linear or nonlinear function sets. The Single, Sigmoidal (linear) and Tan hyperbolician (non-linear) triggering functions are some of the most common set of activation functions. Let us now discuss every one of them, in some detail:

The value of the binary variable is either 0 or 1, respectively. A minimum value is set to do this. The final output of the activation function is returned as one, otherwise the output is returned as 0. If the total weighted neuron input is greater than 1 The conditional trigger function output is either 0 or 1, respectively. A minimum value is set to do this. If the net weighted neuron input reaches 1 the end activation function output will be preserved as 1 or the output will be returned as 0.

They need to see what a standard neural network comprises to grasp the structure of an artificial neural network. This contains a large number of synthetic neurons, for the purpose of defining a normal neural network (of course this is why it is considered an artificial neural network). The Figure 3.2 shows the different layer forms of an artificial neural network:

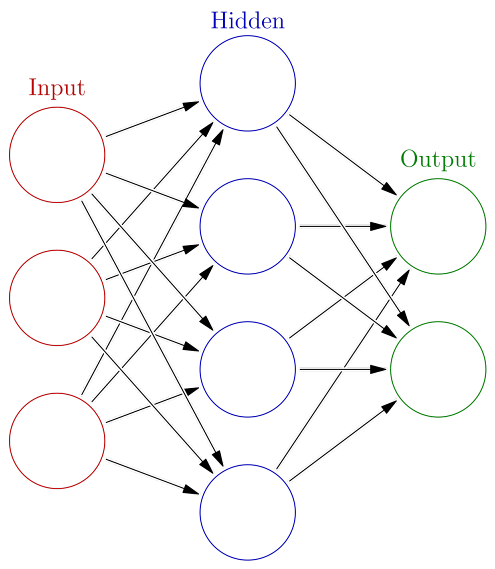


Figure 3.2: Architecture of Neural Network.

- Input layer: The Eingabe layers include the synthetic neurons that transmit feedback from the outside environment (called units). This is where the real network training takes place, for otherwise it will be recalled.

- Output layer: The output layers include units that react to the data fed into the process and also to know whether or not a function has been completed.

- Hidden layer: The hidden layers between input layers and output layers are described. The only task of a secret layer is to transform the data into something useful that can be used by the output layer / unit.

Many neural artificial networks are linked, so that every layer of the neurons in its input layer and its output layer are associated independently, leaving nothing in the sky. It facilitates the whole learning process and also the peak learning happens when after each iteration the weights within the artificial neural network is modified.

3.2.2 Embedding Layer

Data embedding is used in many machine learning applications to create low-dimensional feature representations, which preserves the structure of data points in their original space

A strongly networked computer aggregation algorithm is built here. An extremely nonlinear multi-layer integration feature catches the complicated differences between the heterogeneous knowledge within the network.

Here we build a deep integration feature with a multi-resolution, representing local and global network architectures and rendering the incoming embedding usable for a range of data mining activities. In addition, a rich content and communication knowledge can be extracted from a heterogeneous network so that the correlations between cross-modal data can be specifically calculated in a specific embedding area.

The implementation of off-shelf algorithms for managing vector representations will address a wide variety of data mining problems.

3.2.3 Latent Dirichlet Allocation

Latent Dirichlet Allocation is a probablisitic model for collections of discrete data such as text corpora developed by [22]. Let us break down the terms in the acronym LDA topic modelling.

- Latent: This refers to everything to every information that is known a priori and are hidden in the data. The themes or topics that document consists of are unknown, but they are believed to be present as the text is generated based on those topics.

- Dirichlet: This means a 'distribution of distributions'. For example, if we have a machine that produces dice, we will be able to control whether the machine will always produce a dice with equal weight to all sides, or will there be any bias for some sides. So, the machine producing dice is a distribution as it is producing dice of different types. Also, we know that the dice itself is a distribution as we get multiple values when we roll a dice. This is what it means to be a distribution of distributions and Dirichlet is. Here, in the context of topic modeling, the Dirichlet is the distribution of topics in documents and distribution of words in the topic.

- Allocation: This means that once we have Dirichlet, we will allocate topics to the documents and words of the document to topics.

The above is a brief explanation about LDA and the terms involved. To recap on what an LDA says is that each word in each document comes from a topic and the topic is selected from a per-document distribution over topics. Hence we have two matrices involved here:

- Theta(td) = P(t|d) which is the probability distribution of topics in documents

- Weight(wt) = P(w|t) which is the probability distribution of words in topics

The probability of a word given document i.e. P(w|d) is equal to

$$\sum_{t \epsilon T} p(w|t,d)\, p(t|d)$$

where T is the total number of topics. Let's assume that there is W number of words in our vocabulary for all the documents. If we assume conditional independence, we can say that

P(w|t,d) = P(w|t)

And hence P(w|d) is equal to:

$$\sum_{t=1}^{T} p(w|t)\, p(t|d)$$

which is the dot product of Theta(td) and Weight(wt) for each topic t.

Representing this in the form of a matrix gives the following:

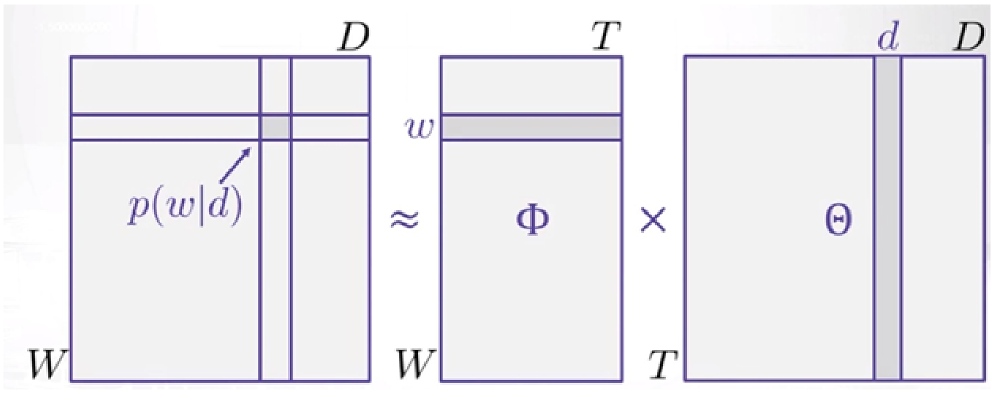


Figure 3.3: Matrix Representation.

Looking at the above equation we can think of LDA similar to that of matrix factorization or SVD, where we decompose the probability distribution matrix of word in document in two matrices consisting of distribution of topic in a document and distribution of words in a topic.

We have the below output at the end of day as shown in Figure 3.3.

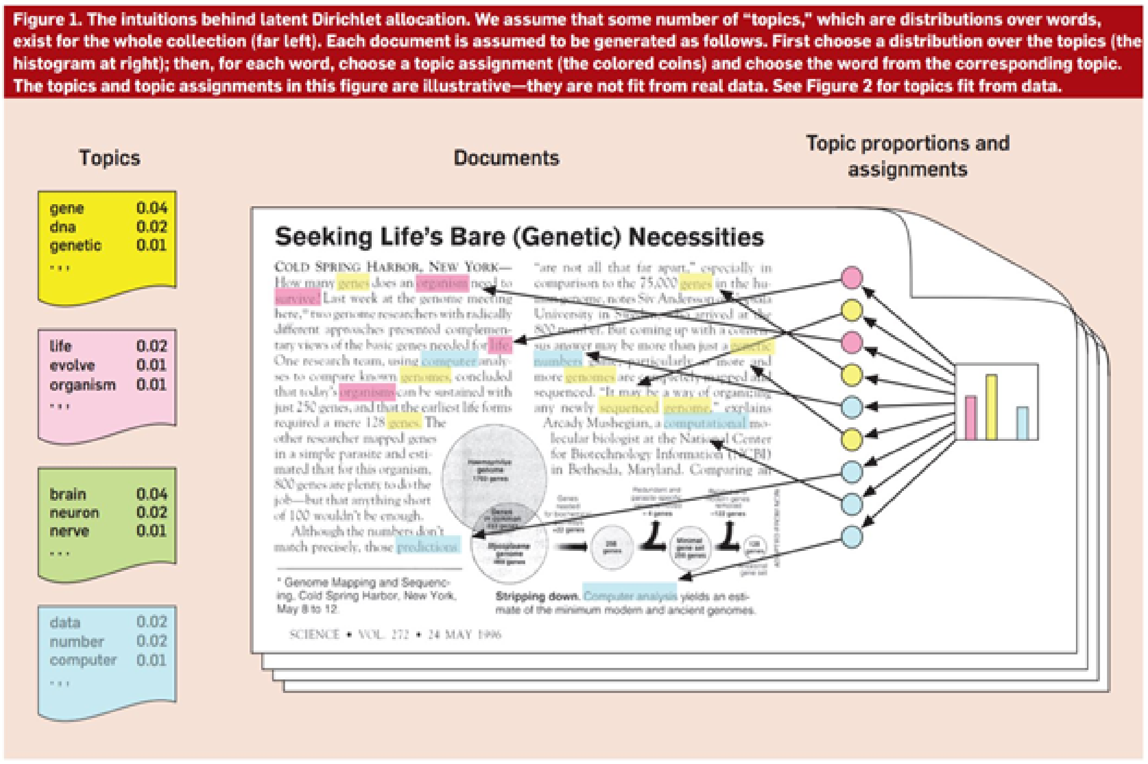


Figure 1. The intuitions behind latent Dirichlet allocation. We assume that some number of "topics," which are distributions over words, exist for the whole collection (far left). Each document is assumed to be generated as follows. First choose a distribution over the topics (the histogram at right); then, for each word, choose a topic assignment (the colored coins) and choose the word from the corresponding topic. The topics and topic assignments in this figure are illustrative—they are not fit from real data. See Figure 2 for topics fit from data.

Figure 3.4: LDA Overview.

Tying back to our example of dice, we could say each word in the distribution of words in a topic is similar to a side of the dice, and we have Dirichlet parameter to control if all the words have the same probability in a topic or will that topic have an extreme bias towards some words. Same intuition is for distribution of topics in a document.

3.2.3.1 Learning weight of two matrix

To start with, let's randomly assign weights to both the matrices and assume that our data is generated as per the following steps:

- Randomly choose a topic from the distribution of topics in a document based on their assigned weights. In the previous example, let's say we chose pink topic
- Next, based on the distribution of words for the chosen topic, select a word at random and put it in the document

- Repeat this step for the entire document

To identify the correct weights, we will use an algorithm called Gibbs sampling. Let's now understand what Gibbs sampling is and how does it work in LDA.

3.2.3.2 Gibbs Sampling

Gibbs sampling [23] is an algorithm for successively sampling conditional distributions of variables, whose distribution over states converges to the true distribution in the long run. This concept needs a good understanding of Monte Carlo Markov Chains and Bayes theorem. Let us assume that we know 'Theta' and 'weight' matrices. Now we will slowly change these matrices and obtain a value that maximizes the likelihood of the data. We will do process on each word by changing the topic assignment of one word. We don't have the topic assignment of the given word but we know the assignment of all other words in the text and we will try to infer what topic will be assigned to the group of words. Looking at the mathematical model, this approach tries to find conditional probability distribution of a single word's topic assignment conditioned on the rest of the topic assignments. Ignoring all the mathematical calculations, we get a conditional probability equation that looks like this for a single word w in document d that belongs to topic k:

$$p(z_{d,n} = k | \vec{z}_{-d,n}, \vec{w}, \alpha, \lambda) = \frac{n_{d,k} + \alpha_k}{\sum_i^K n_{d,i} + \alpha_i} \frac{v_{k,w_{d,n}} + \lambda_{w_{d,n}}}{\sum_i v_{k,i} + \lambda_i}$$

Figure 3.5: Matrix Representation.

In the above equation:

- n(D, κ): Number of times document d use topic k
- v(κ, ω): Number of times topic k uses the given word
- $\alpha\kappa$: Dirichlet parameter for document to topic distribution
- $\lambda\omega$: Dirichlet parameter for topic to word distribution

There are two parts in this equation. First part tells us how much each topic is present in a document and the second part tells how much each topic likes a word. Note that for each word, we will get a vector of probabilities that will explain how likely this word belongs to each of the topics. In the above equation, it can be seen that the Dirichlet parameters also acts as smoothing parameters when n(d,k) or v(k,w) is zero, there will still be some chance that the word will choose a topic going forward.

Document Vectorization The current textual format data needs to be transformed into numeric feature vector to be fed into machine learning models. We have many sophisticated methods like 1 hot encoding, word2vec, doc2vec to compute the feature vector and each method has its own advantage. One of the most common fixed length features is bag of words. Despite their popularity, bag-of-words features have two major weaknesses: they lose the ordering of the words and they also ignore semantics of the words. For example, "powerful," "strong" and "Paris" are equally distant.

3.2.3.3 Usage of Paragraph Vector

In order to learn the fixed length feature representation from variable length pieces of texts, such as sentences, paragraphs and documents, we use Paragraph Vector [24]. This algorithm represents each document by a dense vector which is trained to predict words in the document.Its construction gives algorithm the potential to overcome the weaknesses of bag-of-words models. Empirical results show that Paragraph Vectors outperform bag-of-words models as well as other techniques for text representations.

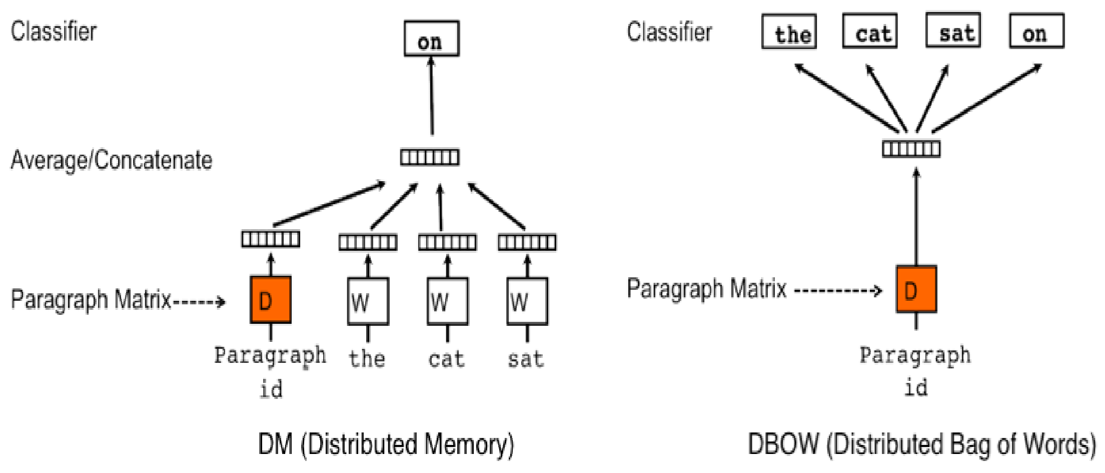


Figure 3.6: Paragraph Vector.

This method is inspired from the famous Word to Vector concept. We compute the word vectors W like the usual process of predicting the next word to occur in the given context of input words. In addition to this, we have a vector to represent the Paragraph, called as the paragraph Vector represented as a column in matrix D. In the next steps, the paragraph vector and the word vector gets averaged or concatenated to give a new vector as mentioned in the image. The paragraph ID is imagined as another word. This acts as a memory that remembers the missing current context or the topic of the paragraph. For this reason, this model is also called as Distributed Memory Model of Paragraph Vectors. This method has successfully outperformed the conventional bag of words model giving higher accurate results in text classification. We have not used the DBOW method mentioned in the figure. The DBOW method is a reverse training process. The DBOW method ignores inputting the context words . Instead, it forces the model to predict words randomly sampled from the paragraph in the output.

The main reason that motivated us to use the Paragraph vectors is its ability to learn and produce vectors from unlabelled data. Thus, it can work well for the CASE SUMMARY information that does not have accurate labelled information on the topic to which it belongs.

CHAPTER 4: METHOD

Our approach to develop a centralized dataset for the Supreme Court of United States is based applying text mining techniques on the top of the data source collected. We have validated the logics used to transform the unstructured data during the tenure of the project and improved them as we find errors. After preparing a clean dataset, we have applied topic modelling and vectorization process on the text information in the data to cluster the cases based on similarity.

4.1 Data Collection Process

In this chapter, we have explained the method used to collect the data. We have collected the data of all the state supreme court judgements from 1053 to 2014 using LexisNexis Academic. The Lexis Nexis Academic provides access to the full text information of the cases happened in the State Courts. This is a standard element in many research and information literacy programs.

During the collection of data from lexisnexis, we get chunk of text files corresponding to each state by each year. A text file contains the cases that happened in a particular year in the form of documents, where each document is a case information. These documents are the key source of processing for this project.

We have 50 states in the United States Country. Each of the state has only 1 state court. However, the state of Oklahoma and Texas is divided into Supreme Court and Court of Criminal Appeal. Hence we have 52 courts in the dataset.

4.2 Methods in Data Processing

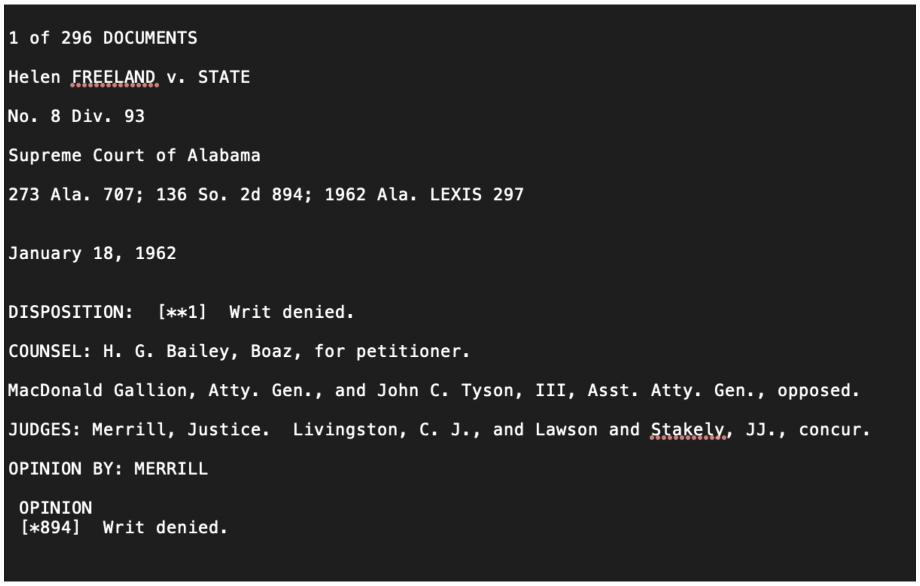
In this section, we will explain the step by step process that is followed to extract the information from the text files.

4.2.1 Step 1: State Indexing

Each state has a folder, we index the folders of all the states. Secondly, we check for the number of text that are present in each of that state folder. Each text file, usually represents a particular year. For example, a folder with the name "ALABAMA", containing a text file "1956.txt". This file will have the cases happened in the year 1956 at the Supreme Court of Alabama.txt in the Supreme Court of Alabama. Before processing each of the state, we index the list of files that is present in the STATE folder.

4.2.2 Step 2: Case Document Indexing

We open each of the text files in the STATE folder. Once the text file is opened, we check for the number of documents present in the text file. For example, "1 of 300 Documents" indicates that we have 300 documents in the text file. Each document is separated by a text mentioning the document number. To identify the text corresponding to a case, we extract the body of information from the first line to the last line in between the two document index information. We call this text body as CASE TEXT. The Figure 4.1 shows an example of one CASE TEXT for a case happened on January 18, 1962 in the state of ALABAMA. Similarly we index for every next case and pass the body of text to the case parser.



```
1 of 296 DOCUMENTS
Helen FREELAND v. STATE
No. 8 Div. 93
Supreme Court of Alabama
273 Ala. 707; 136 So. 2d 894; 1962 Ala. LEXIS 297

January 18, 1962

DISPOSITION:  [**1]  Writ denied.
COUNSEL: H. G. Bailey, Boaz, for petitioner.
MacDonald Gallion, Atty. Gen., and John C. Tyson, III, Asst. Atty. Gen., opposed.
JUDGES: Merrill, Justice.  Livingston, C. J., and Lawson and Stakely, JJ., concur.
OPINION BY: MERRILL
 OPINION
 [*894]  Writ denied.
```

Figure 4.1: Text Parse.

4.2.3 Step 3: Case Parser

The Case parser is a function written in python to extract as much as structured information in the case text as possible.

- Get Case Party: In the first line of the CASE TEXT, we have information related to the two case party involved in the case separated by " v. ". The two case party are located on the left and right of the " v. ". In the above example the two case party are "Helen FREELAND" and "STATE". We defined a function called getCaseparty that can parse this information and return the two case party information.

- Lexis Number: The Lexis Number is a unique identifier assigned to each new case to find and reference them. The LEXIS number contains the year of decision in the same position as the volume number on a standard reporter citation, an abbreviation of the court, agency or document type and a sequential document number. The last number in a standard LEXIS cite is a document number, not a page number LEXIS cites are assigned sequentially. For an example, 1987 U.S. App. LEXIS 1480 is a separate case from 1987 U.S. App. LEXIS 1481 because the last number is not a page number. The LEXIS pagination for every case starts at page 1. For example, page 5 of 1987 U.S. App. LEXIS 1480 is cited as 1987 U.S. App. LEXIS 1480, 5, not 1987 U.S. App. LEXIS 1480, 1484. The LEXIS number is a part of the full citation number. The full citation contains three sections: STATE REPORTER, REGIONAL REPORTER and LEXIS NUMBER. As an example, the full citation is usually placed in the 3rd line of the CASE TEXT. The python script checks for the term LEXIS present in the 3rd to 7th line in the CASE TEXT. If the term is present, we extract that sentence and screen it through a Regular expression to split based on the occurrence of STATE REPORTER, REGIONAL REPORTED and LEXIS

NUMBER. We have accounted in to handle exceptions in processing this information.

- Date Processing:
- Court Name Processing: It becomes mandatory to identify the name of the court in which the case happened. The data source contains not only the supreme court information but also many other low level courts like the court of appeal included in the dataset. In order to intelligently handle and focus only on the cases that we are working on, we have created a function on this. The Court Name Processing function checks on the LEXIS number to identify the court name. The LEXIS NUMBER contains the court name, mentioned as an acronym.
- Case Text Parsing: In this section, we loop on each of the sentences of the case text body and try to extract information from each line.
- Subsequent History: The text followed by the heading 'Subsequent History: ' contains the information related to subsequent history. Subsequent History re refers to the opinions issued by the court that review the case after the opinion you are citing.
- getPriorHistory: If a particular case was handled before by a court and a judge, and due to specific reasons if a case gets appealed in a different court, then we do have the information related to the court on which the case happened previously and the judges handling the case previously. The getPriorHistory function helps to extract the court and judge information from the text body. Hence this function return the prior court and the judge information if the case was appealed in court previously.
- Disposition: The disposition is a text that informs the court's final determina-

tion of a case or issue. The disposition of a case starts with the text "disposition:". The text followed by this heading is extracted. The text in the DISPOSITION is a finite number of text and decision that can happen. Hence we have created a dictionary for the possible word, associating them with the corresponding numerical index that represents the meaning of disposition. Transforming this to a numerical value can support in conducting statistical analysis in the future. The 'Disposition Value' contains the numerical values of the disposition.

- Case Summary: The case summary text is split into 3 sections: Procedural Posture, Overview and the Outcome.

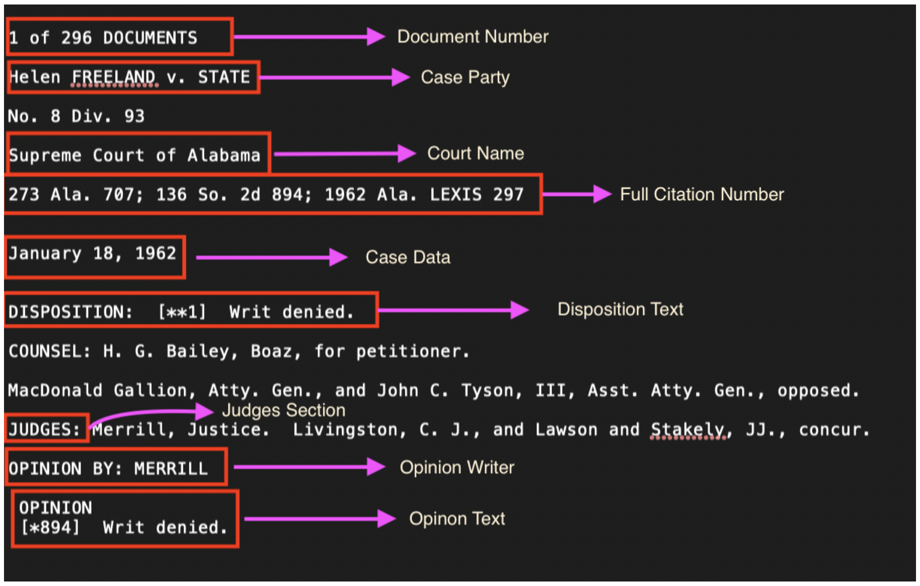
  - The Procedural posture describes the case procedural history on how the case arrived before this court.
  - Overview describes a brief review of the underlying facts, legal issues and the court's holding(s)
  - Outcome contains the ultimate procedural disposition of the issue(s)

  We have extracted these information form the CASE SUMMARY in the CASE TEXT. Each of the text body starts with the corresponding heading like 'Procedural Posture:','Outcome:' and 'Overview:'.

- Judges: The text body followed by the 'JUDGES:' heading contains the list of judges involved in handling the case.

- Opinion, Dissent, Consur Writer: Similar to the previous sections, we have extracted the opinion writer, dissent writer, concur writer based on the heading and the text followed by the heading. As an exception, if there is no opinion writer present in the text, we populate the text 'Per Curiam' on the opinion writer section. We have incorporated exceptional cases and logics used by researchers in the Supreme Court in the code script.

- Concur, Dissent, Opinion Text Body: Each of the concur writer comes up with a writing to explain the reason for them to concur. We have followed an intelligent technique to identify the extraction of the text body corresponding to the concur statement. The text followed by the concur writer name is extracted until the next section of statement related to opinion or dissent follows. Similarly we have extracted the text related to opinion and dissent section. The text body present here follows a lot of false structuring.



```
1 of 296 DOCUMENTS
Helen FREELAND v. STATE
No. 8 Div. 93
Supreme Court of Alabama
273 Ala. 707; 136 So. 2d 894; 1962 Ala. LEXIS 297

January 18, 1962

DISPOSITION:  [**1]  Writ denied.
COUNSEL: H. G. Bailey, Boaz, for petitioner.
MacDonald Gallion, Atty. Gen., and John C. Tyson, III, Asst. Atty. Gen., opposed.
JUDGES: Merrill, Justice.  Livingston, C. J., and Lawson and Stakely, JJ., concur.
OPINION BY: MERRILL
 OPINION
 [*894]  Writ denied.
```

Figure 4.2: Text Body.

The Figure 4.2 shows a sample of useful information present in the text body. Each of the text body extracted from the CASE TEXT is cleaned and stored in the spreadsheet. The cleaning process removes the unwanted numerical and symbolic characters present in between the text.

4.2.4 Step 4: Writing Outputs

The parser function generates multiple outputs by extracting the information from the CASE TEXT. In this section we have explained the outputs generated by the parser function.

- Spreadsheets: The parser function prepares a spreadsheet file that contains all the information extracted from each of the case document. The spreadsheet contains each of the case as an entry in the spreadsheet. The data description of the dataset is attached in the appendix section for reference.

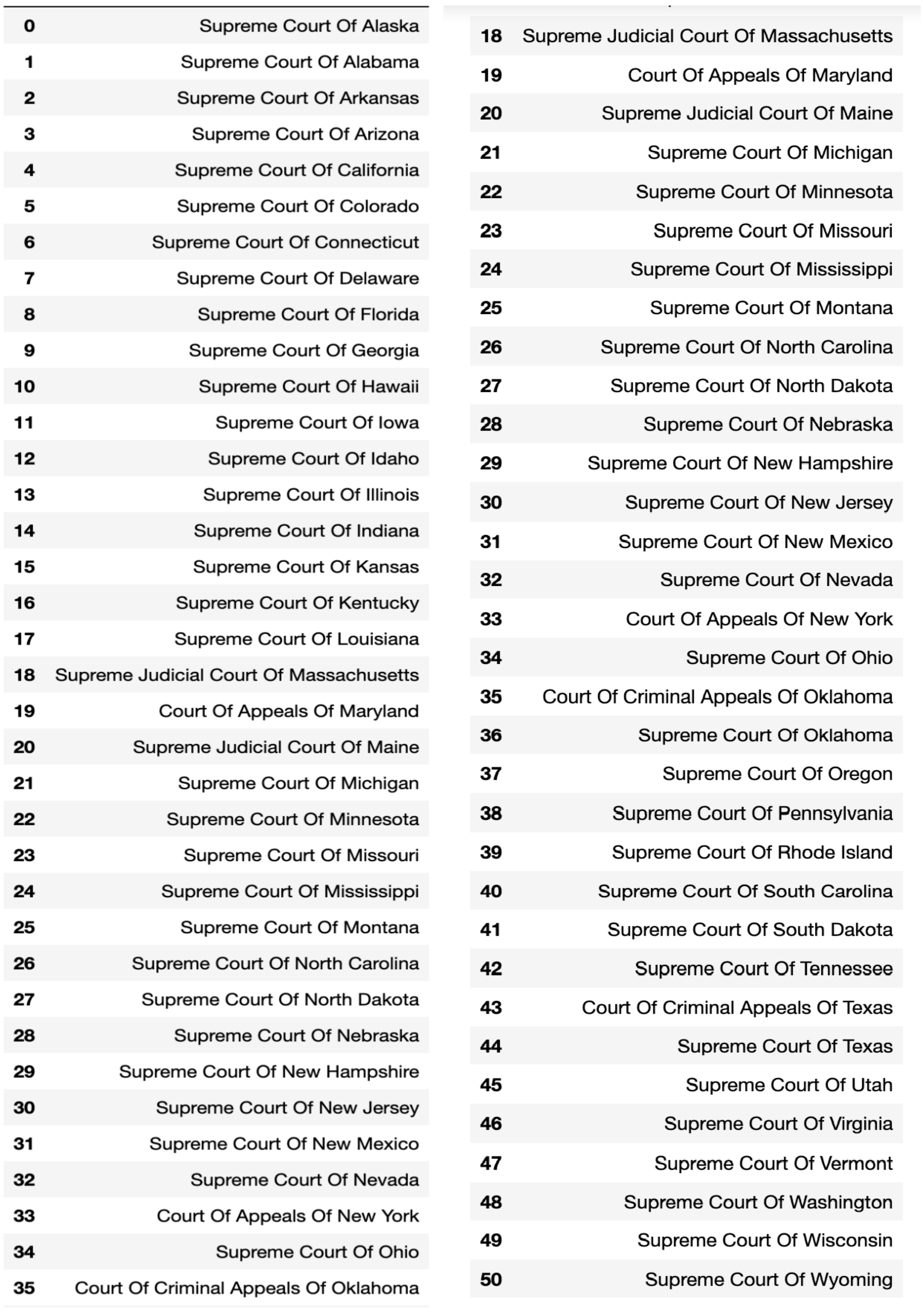
- Opinion Text: The parser function creates an external text file with the directory "/output/$STATE/opinion/" to store the opinion text identified in each of the CASE TEXT.

- Dissent Text: The parser function creates an external text file with the directory /output/$STATE/dissent/ to store the dissent text identified in each of the CASE TEXT.

- Concur Text: The parser function creates an external text file with the directory /output/$STATE/concur/ to store the concur text identified in each of the CASE TEXT. Hence in the Data processing section we have explained about the steps followed to extract the useful information present in each of the case and transformed this into a clean Comma Separated Value file for future analysis.

4.3 Analysis on dataset prepared

Throughout the process of extracting and preparing the dataset, we have handled 6255 text files corresponding to 52 states. We have extracted 2081581 files after a repeated error analysis and improvement in the logic used to extract the information. We have the following list of courts present in the dataset as shown in the Figure 4.3
.



0	Supreme Court Of Alaska
1	Supreme Court Of Alabama
2	Supreme Court Of Arkansas
3	Supreme Court Of Arizona
4	Supreme Court Of California
5	Supreme Court Of Colorado
6	Supreme Court Of Connecticut
7	Supreme Court Of Delaware
8	Supreme Court Of Florida
9	Supreme Court Of Georgia
10	Supreme Court Of Hawaii
11	Supreme Court Of Iowa
12	Supreme Court Of Idaho
13	Supreme Court Of Illinois
14	Supreme Court Of Indiana
15	Supreme Court Of Kansas
16	Supreme Court Of Kentucky
17	Supreme Court Of Louisiana
18	Supreme Judicial Court Of Massachusetts
19	Court Of Appeals Of Maryland
20	Supreme Judicial Court Of Maine
21	Supreme Court Of Michigan
22	Supreme Court Of Minnesota
23	Supreme Court Of Missouri
24	Supreme Court Of Mississippi
25	Supreme Court Of Montana
26	Supreme Court Of North Carolina
27	Supreme Court Of North Dakota
28	Supreme Court Of Nebraska
29	Supreme Court Of New Hampshire
30	Supreme Court Of New Jersey
31	Supreme Court Of New Mexico
32	Supreme Court Of Nevada
33	Court Of Appeals Of New York
34	Supreme Court Of Ohio
35	Court Of Criminal Appeals Of Oklahoma

18	Supreme Judicial Court Of Massachusetts
19	Court Of Appeals Of Maryland
20	Supreme Judicial Court Of Maine
21	Supreme Court Of Michigan
22	Supreme Court Of Minnesota
23	Supreme Court Of Missouri
24	Supreme Court Of Mississippi
25	Supreme Court Of Montana
26	Supreme Court Of North Carolina
27	Supreme Court Of North Dakota
28	Supreme Court Of Nebraska
29	Supreme Court Of New Hampshire
30	Supreme Court Of New Jersey
31	Supreme Court Of New Mexico
32	Supreme Court Of Nevada
33	Court Of Appeals Of New York
34	Supreme Court Of Ohio
35	Court Of Criminal Appeals Of Oklahoma
36	Supreme Court Of Oklahoma
37	Supreme Court Of Oregon
38	Supreme Court Of Pennsylvania
39	Supreme Court Of Rhode Island
40	Supreme Court Of South Carolina
41	Supreme Court Of South Dakota
42	Supreme Court Of Tennessee
43	Court Of Criminal Appeals Of Texas
44	Supreme Court Of Texas
45	Supreme Court Of Utah
46	Supreme Court Of Virginia
47	Supreme Court Of Vermont
48	Supreme Court Of Washington
49	Supreme Court Of Wisconsin
50	Supreme Court Of Wyoming

Figure 4.3: Courts List.

The spreadsheets generated by the data processing will be made open source very soon by the authors of this project. Here is a list of analysis that was performed on top of the dataset that was generated in the previous steps:

- We have made an understanding on the number of cases that happened in each of the state and plotted to view the top 10 states that had more number of cases in the Supreme Courts. The Figure 4.4 indicate that New York has the most number of cases, following which is California.

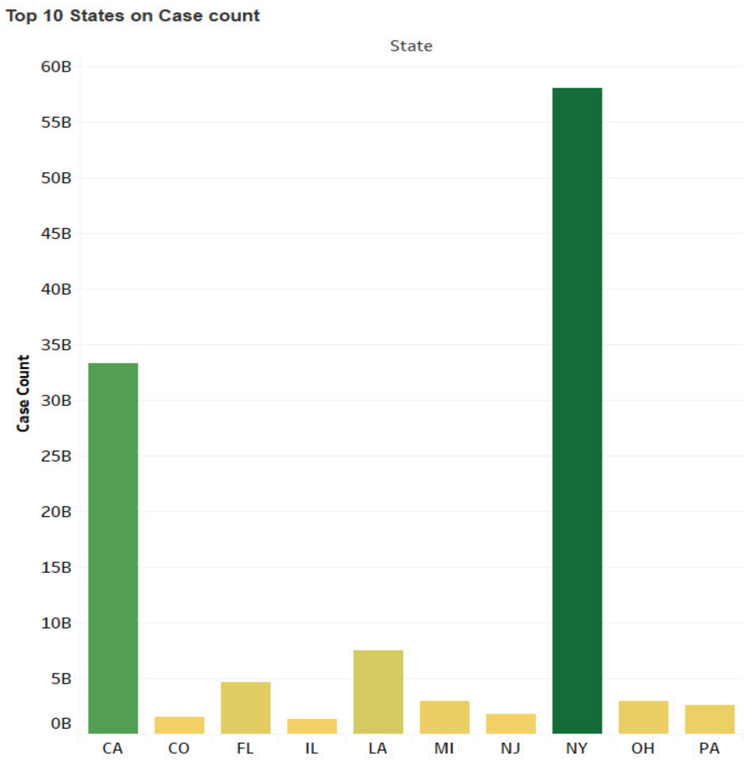


Figure 4.4: Top 10 States on Case Count.

- The west published 7 region reporters. West publishes seven regional reporters (Atlantic, North Eastern, North Western, Pacific, Southern, South Eastern, South Western). The National Reporter System covers the appellate courts of all the states and the District of Columbia. The Second and Third Series of West's Regional Reporters are located in the Main Reading Room (MRR). The regional reporters, their abbreviations and the location and call numbers are:

- Atlantic Reporter A., A.2d, A.3d Golding, MRR KF135.A7 A8
- California Reporter Cal. Rptr., Cal. Rptr.2d, Cal. Rptr.3d Golieb NYUL KA-C2 61
- New York Supplement N.Y.S., N.Y.S.2d B2S, MRR NYUL KA-N7 65
- North Eastern Reporter N.E., N.E.2d Golding, MRR KF135.N6 N63
- North Western Reporter N.W., N.W.2d Golding, MRR KF135.N7 N62
- Pacific Reporter P., P.2d, P.3d Golding, MRR KF135.P2 P33-34x
- South Eastern Reporter S.E., S.E.2d Golding, MRR KF135.S6 S612
- South Western Reporter S.W., S.W.2d, S.W.3d Golding, MRR KF135.S7
- S612-613x
- Southern Reporter So., So.2d, So.3d Golding, MRR KF135.S8 S612

• We performed an analysis on the regional reporter number and the following visual explains distribution of regional reporters. The Figure 4.5 explains to us that there is more cases happening in N.E. region and least in S.W.

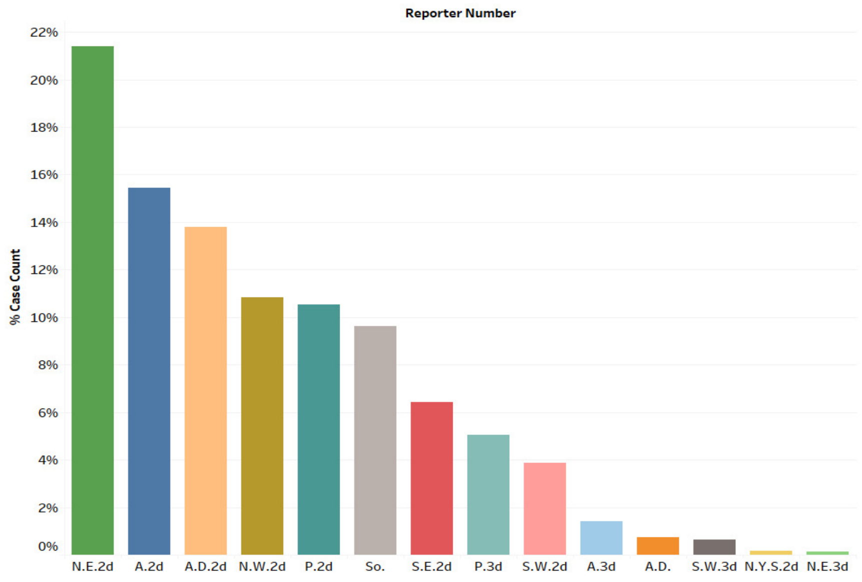


Figure 4.5: Regional Reporters.

- An analysis on the distribution of cases happening across various months of the year is shown in the Figure 4.6. We could infer that, August has the least number of cases happening compared to the remaining 11 months and June has the highest number of cases happening in a year.

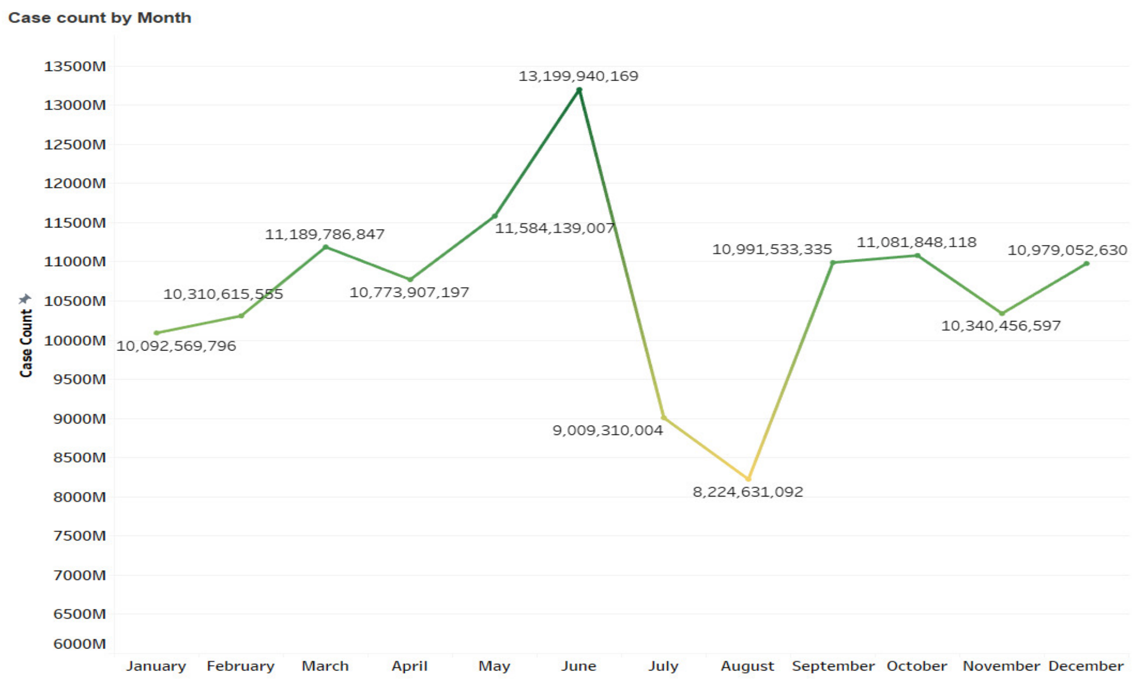


Figure 4.6: Case Count by Month.

4.4 Machine Learning on processed dataset

In this section we have explained about a potential research that can be done in the dataset prepared to identify and create a correlation across the 2 Million cases in the dataset and make interesting insights from them. Language understanding is a challenge for the computers. Subtle nuances of communication that human toddlers can understand still confuses the most powerful machines. Even though advanced techniques like deep learning can detect and replicate complex language patterns, machine learning models still lack fundamental conceptual understanding of what our words really mean. That said, 2018 did yield a number of landmark research breakthrough which pushed the domain of Natural Language Processing (known predominantly as

NLP), understanding and generation forward.

Having given as the advancements that happen in the domain of NLP and Deep Learning, we have attempted to apply multiple cutting edge techniques like Topic Modelling, Document Vectorization, Embedding Neural Network to explore hidden insights that these model can explain to us.

4.5 Natural Language Processing on Case Summary

Huge corpus like the United States' State Court data with more than 2 million files contain heavy volume of text data. It is mandatory to follow efficient methods to handle the memory issue which is a limiting factor in practical applications. This will enable us to build scalable machine models and ease of use. There is a lot of software advancements done in the domain to efficiently converting the text information into a numerical form. In the next couple of sections, we have explained about 2 powerful algorithms in Natural Language Processing that is used widely in processing text data chunks and extract meaningful information.

Algorithms for Text Data Processing:

- Latent Dirichlet Allocation
- Embedding

We have processed the text using the above mentioned algorithms to extract meaningful information that can be fed as input to the future user embedding task. In the below section, we have explained on the techniques used to clean the text data before feeding to a topic modelling process.

4.5.1 Text Cleaning on Case Summary

In order to get the full understanding of the case, we make use of the 'CASE SUMMARY' section. This section provides the background information of the case and it is divided into 3 sections:

- Procedural Posture - This section describes the case's procedural history on how this case arrived before this court

- Overview - This section provides a brief review of the underlying facts, legal issues and the courts holdings

- Outcome: This contains the ultimate procedural disposition of the issue

We crunch the above mentioned 3 sections to extract as much useful information as possible. These sections contain the information in the text format and each text is of varying length. Not all the cases promise to contain the 3 section information. The below table explains the number of cases and the ratio of cases that contains CASE SUMMARY text.

Table 4.1: Case Ratio table.

Data Description	Count	Percentage
Total Cases	2,081,581	100%
Cases with Procedural Posture	562,137	27%
Cases with Overview	567,437	27.25%
Cases with Outcome	567,430	27.25%

4.5.2 Processing Case Summary

Transforming the text in column of the dataset into something that an algorithm can digest is a complicated process. We have followed 4 different parts:

- Cleaning consist of getting rid of the less useful parts of text through stopword removal, dealing with capitalization, punctuation and characters and other details

- Annotation consists of the application of a scheme to texts. Annotation may include structural markup and part of speech tagging

- Normalization consists of the translation of terms in the scheme or linguistic reductions through stemming, lemmatization and other forms of standardization
- Analysis consists of statistically probing, manipulating and generalizing from the dataset for feature analysis.

For the Court Case dataset, we have followed the same generic steps involved in preprocessing of the text as shown in the process flow diagram 4.7 below.

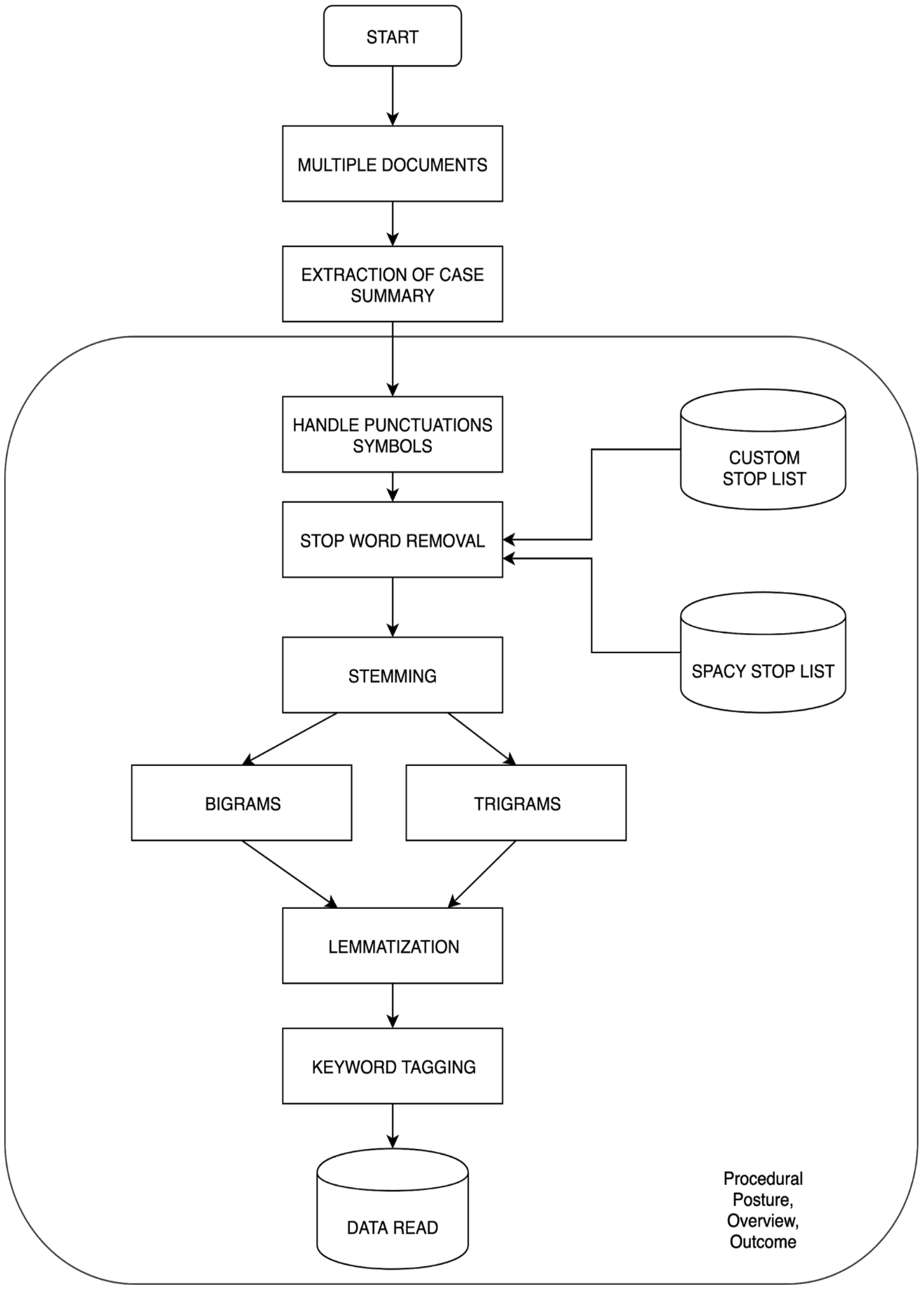


Figure 4.7: Steps in Preprocessing.

After all the preprocessing steps mentioned above is performed, the data is free of all the noise and dust. Now we have only important information that needs to be focused in a text.

4.5.3 Generating Topic of the Case

Topic modeling is an unsupervised natural language processing which is used to represent a text document with the help of several topics, that can best explain the underlying information in a particular document. Here we attempt to cluster the words together that occur for a particular topic.

We can see a huge amount of textual data lying in an unstructured format over the OVERVIEW, OUTCOME, PROCEDURAL POSTURE sections. We need a way to understand, organize and label this data to make informed decisions. Given the success of topic modelling on various applications like finding similar questions on stack overflow that are similar to each other, news flow aggregation and analysis, recommender systems, etc., we will apply this technique on the text data of the SUPREME COURT DATASET. All of these focus on finding the hidden thematic structure in the text, as it is believed that every text that we write be it a tweet, post or a research paper is composed of themes like sports, physics, aerospace etc. There are many different methods of performing topic modelling like LDA, Hierarchical LDA, LSA, PLSA, LDA2VEC and we have followed the LDA method for its promising results.

The following are the steps followed in conducting topic modelling on the all the documents on the case files.

- The data from the text cleaning is use das the source in this topic modelling.
- The dataset is fed to the Topic Modelling algorithm. We have conducted topic modelling on different number of topics and the model with the high coherence score is used for further processing.
- From the coherence score analysis, we understood 17 is the optimal number of topics present in the dataset.

- We have generated an interactive topic modelling visualization to understand the word distribution across each topic. By doing this, we could visually interpret the topic that a cluster belongs to.

- Later, we have manually labelled the names for each of the topic and appended the results to the case documents. Thus the dataset is exported.

For the purpose of performing topic modelling on this huge dataset, we have used the High Performance Computing of the University of North Carolina. The high performance cluster computers support the process of running the topic modelling repeatedly for various number of topics.

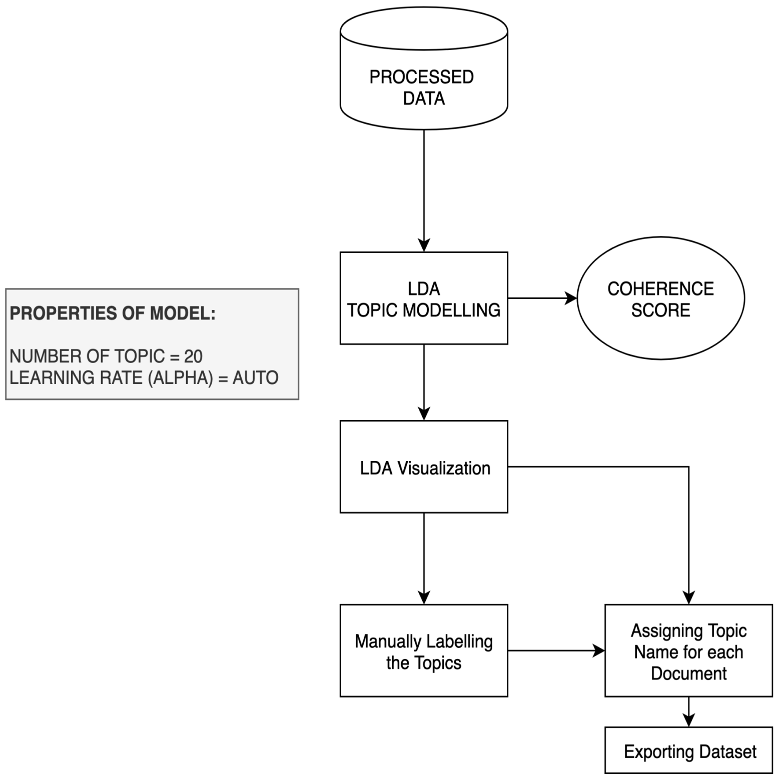


Figure 4.8: LDA Topic Modeling.

4.5.4 Generating Embedding

The cleaned keyword tagged data in the previous stage is used for the producing the embeddings. Embedding is the mapping of a categorical or discrete variable to a vector of continuous numbers. In the context of a neural network, embeddings are low dimensional, learned continuous vector representation of discrete variables. The advantage of using a neural network embeddings is, they can reduce the categorical variables and meaningfully represent categories in the transformed space. Neural Network Embedding have 3 primary purposes:

- Finding nearest neighbour in the embedding space. These can be used to make recommendations based on user interests tot cluster categories.
- As input to a machine learning model for a supervised task
- For visualization of concepts and relations between categories.

For all these advantages, we have worked on creating an embedding for the cases. We build the vocabulary list that contains all the words used in the dataset. We feed both the vocabulary list and the cleaned keyword tagged data as input to the model. We have made a study on identifying the right number of dimensions to be set to extract as much information as possible from the text body of the CASE SUMMARY. For this, we have repeatedly performed the process on various length of dimension like 20, 30, 50, 100, 200, 300 dimensions. The number of epochs to be trained is set to 100. The model is trained on a distributed memory to get efficient results in a fast manner. We have used the High Performance Computer called COPPERHEAD with 1 core: 4 node : 1 gpu, configuration for the training process. The optimization method followed is explained in the next section.

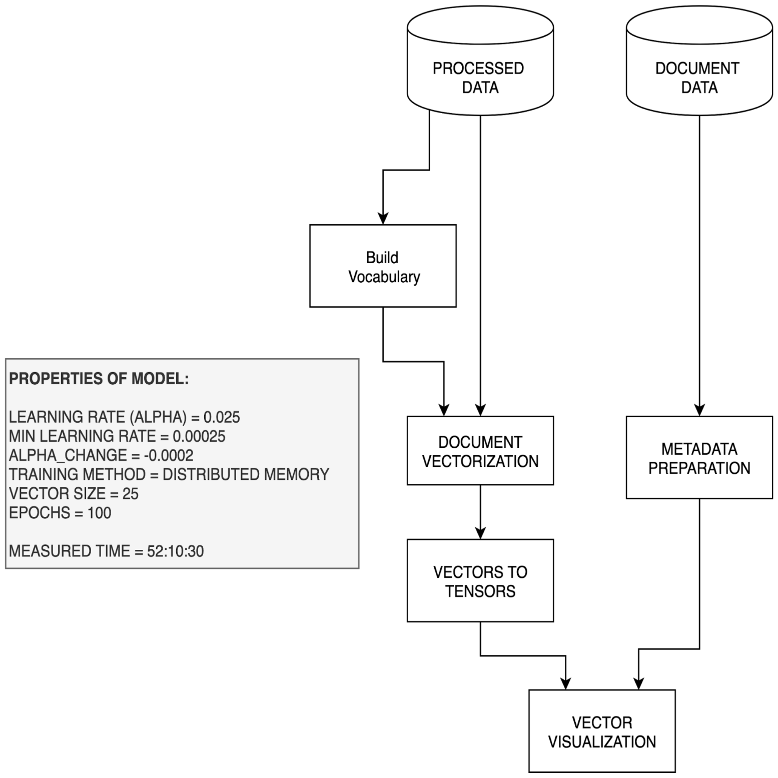


Figure 4.9: Generating Embeddings.

4.5.5 Optimizer Configuration

The model has an initial learning rate of 0.025 and it is set to a variable learning rate. The learning rate decreases by 0.0002 after each epoch of training until it reaches 0.00025. This basically informs the learning rate changes until it reaches 123 epochs and later on it remains stable at 0.00025. The advantage of having a variable learning rate is to enable the model to converge and find the optimal minimum value without getting bounced back from the convergence curve while doing the stochastic gradient descent optimization. In the next section we have explained about gradient descent optimization and stochastic gradient descent optimization.

4.5.6 Gradient Descent Optimization

Gradient descent is an iterative algorithm that starts from a random point on a function and travels down its slope in steps until it reaches the lowest point of that function.

This is the crux of the algorithm. The general idea is to start with a random point and find a way to update this point with each iteration such that we descend the slope.

The steps of the algorithm are

- Find the slope of the objective function with respect to each parameter/feature. In other words, compute the gradient of the function.
- Pick a random initial value for the parameters.
- Update the gradient function by plugging in the parameter values.
- Calculate the step sizes for each feature as : step size = gradient * learning rate.
- Calculate the new parameters as : new params = old params - step size Repeat steps 3 to 5 until gradient is almost 0.

The "learning rate" mentioned above is a flexible parameter which heavily influences the convergence of the algorithm. Larger learning rates make the algorithm take huge steps down the slope and it might jump across the minimum point thereby missing it. So, it is always good to stick to low learning rate such as 0.01. It can also be mathematically shown that gradient descent algorithm takes larger steps down the slope if the starting point is high above and takes baby steps as it reaches closer to the destination to be careful not to miss it and also be quick enough.

4.5.7 Stochastic Gradient Descent(SGD)

Stochastic gradient descent is a very popular and common algorithm used in various Machine Learning algorithms, most importantly forms the basis of Neural Networks. Gradient, in plain terms means slope or slant of a surface. So gradient descent literally means descending a slope to reach the lowest point on that surface. There are a few downsides of the gradient descent algorithm. We need to take a closer look at the amount of computation we make for each iteration of the algorithm.

Say we have 10,000 data points and 10 features. The sum of squared residuals consists of as many terms as there are data points, so 10000 terms in our case. We need to compute the derivative of this function with respect to each of the features, so in effect we will be doing 10000 * 10 = 100,000 computations per iteration. It is common to take 1000 iterations, in effect we have 100,000 * 1000 = 100000000 computations to complete the algorithm. That is pretty much an overhead and hence gradient descent is slow on huge data. Stochastic gradient descent comes to our rescue. "Stochastic", in plain terms means "random". It is also common to sample a small number of data points instead of just one point at each step and that is called "mini-batch" gradient descent. Mini-batch tries to strike a balance between the goodness of gradient descent and speed of SGD.

CHAPTER 5: EVALUATION AND RESULT

For the purpose of evaluation of the vectors that are built in the previous stages, we have followed a technique of visual evaluation using Tensorboard Projector Platform. This evaluation gives more understanding about the vectors. The tensorboard projector has sophisticated dimensionality reduction methods like UMAP, T-SNE, PCA to reduce the number of dimensions of the vector to 3 or 2 dimensional vector and visualize them in the 3 cartesian coordinate axis. In order to visualize them, we need to convert the result of the model to a tab separated file supported by the tensorboard that consist of vectors for each of the 562137 case file. We used the word2vec2tensor file provided by Tensorflow for the conversion process. Next, we need to prepare a file that contains the metadata of the vectors. The metadata contains the FULL CITATION, STATE, DAY, YEAR, MONTH, TOPIC of each of the case in the dataset as shown in the diagram. The metadata is a tab separated value file.

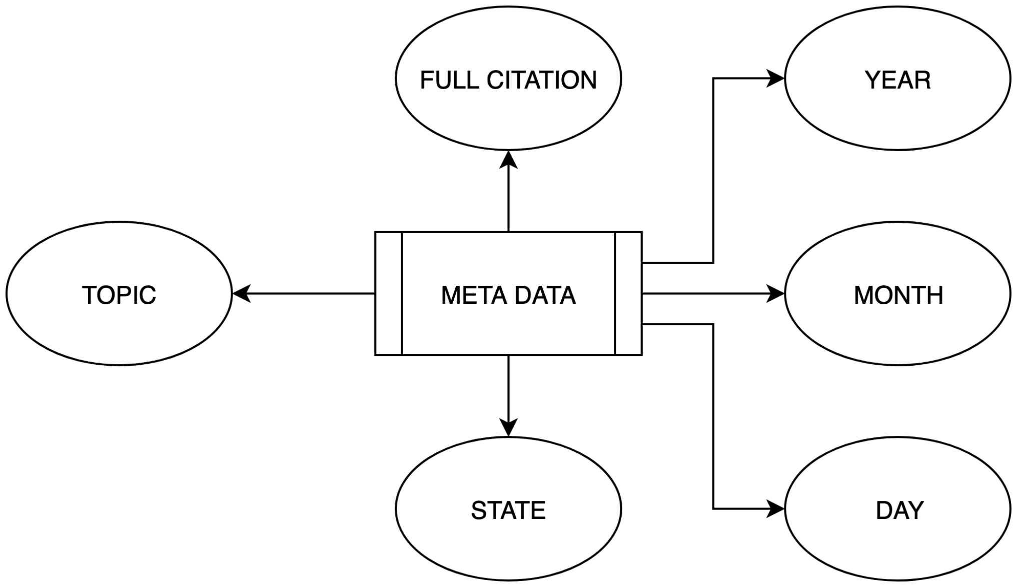
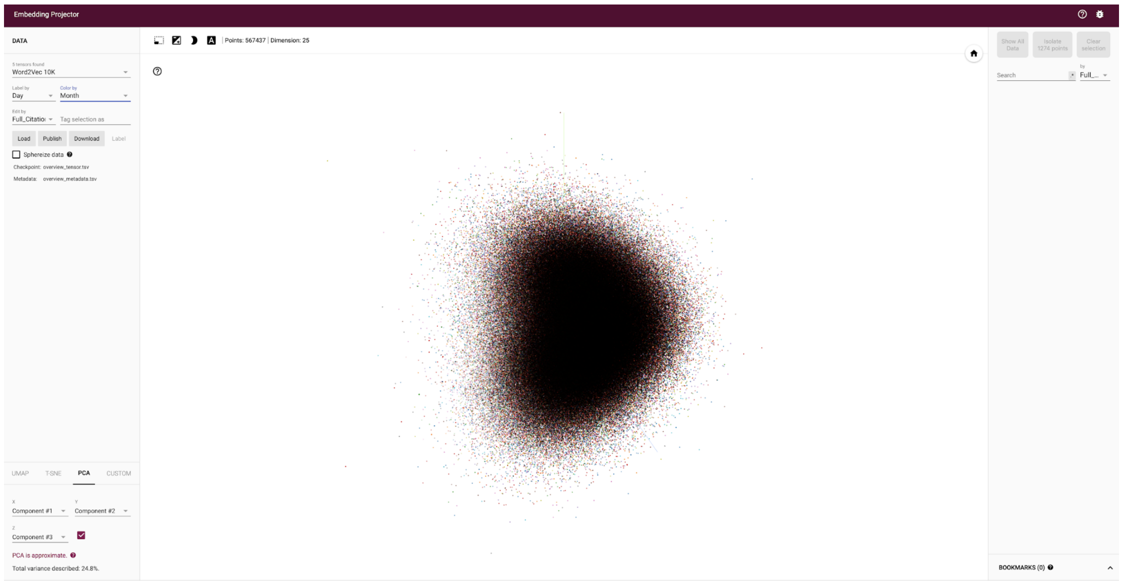


Figure 5.1: Metadata of the Vector.

We use the tensor file and the metadata file in the visualization on tensorboard. We open the tensorboard projector in this link `https://projector.tensorflow.org`. Click on the 'Load' button the left and upload the tensor and metadata file. After the parsing process is complete, we will be able to see a 3 dimensional plot as shown in the figure 5.2 below.



Points: 567437 | Dimension: 25

5 tensors found: Word2Vec 10K

Label by: Day
Color by: Month
Edit by: Full_Citatio
Tag selection as

☐ Sphereize data

Checkpoint: overview_tensor.tsv
Metadata: overview_metadata.tsv

X: Component #1
Y: Component #2
Z: Component #3 ✓

PCA is approximate.
Total variance described: 24.8%.

Figure 5.2: TensorFlow 3D plot.

5.1 Features in the Analysis Platform

- Color: We have the 'Label By' option on the left window that enables us to differentiate the cases based on the state, day, month. Selecting a particular label will color the data points in the plot based on the property. For example, once we select the 'Label by' to 'STATE', we will have all the 50 states colored in 50 colors. Given the human eye ability to differentiate 50 colors, it is difficult to segregate the states by glance. However, the plot contains 50 colors corresponding to 50 states.

- Click and Explore: We can click on any of the datapoint in the plot. This will highlight only a particular data point and show all the metadata associated with this data point on the right top corner. We will be able to see the FULL CITATION, STATE, DAY, MONTH, YEAR of the case.

- Similarity Measure: Once, we select a particular datapoint, an option window opens on the right. We can isolate the closest points to the selected data point. We have the freedom to also specify the number of close points to be considered. The close points are selected based on two distance measuring metrics. I. Cosine Similarity : is a measure of similarity between two non-zero vectors of an inner product space that measures the cosine of the angle between them. Cosine Similarity is generally used as a metric for measuring distance when the magnitude of the vectors does not matter. II. Euclidean Distance : is a measure of direct straight line distance between the two data points.

- Dimensionality Reduction Techniques: We have the popular dimensionality reduction techniques given as feature in the bottom left corner. They are UMAP, T-SNE, PCA. The default method is PCA with 3 vectors.
- Bookmark Analysis: If a researcher conducts a particular analysis, we could save the visuals and the profile used for analysis by using the bookmark option on the bottom right option.

The following figures 5.3, 5.4, 5.4 shows the output of the tensorboard for various number of dimensions.

- 50 Vectors

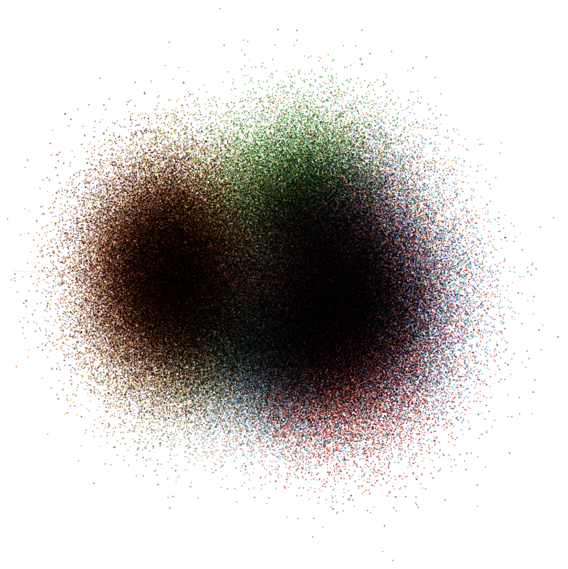


Figure 5.3: 50 Vectors.

- 100 Dimensions

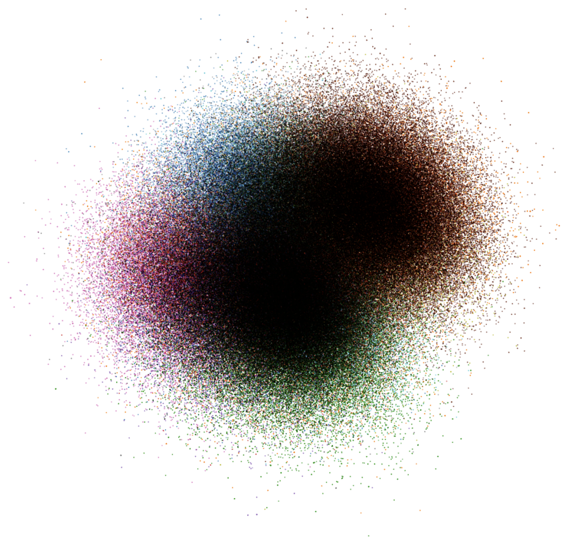


Figure 5.4: 100 Dimensions.

- 300 Dimensions

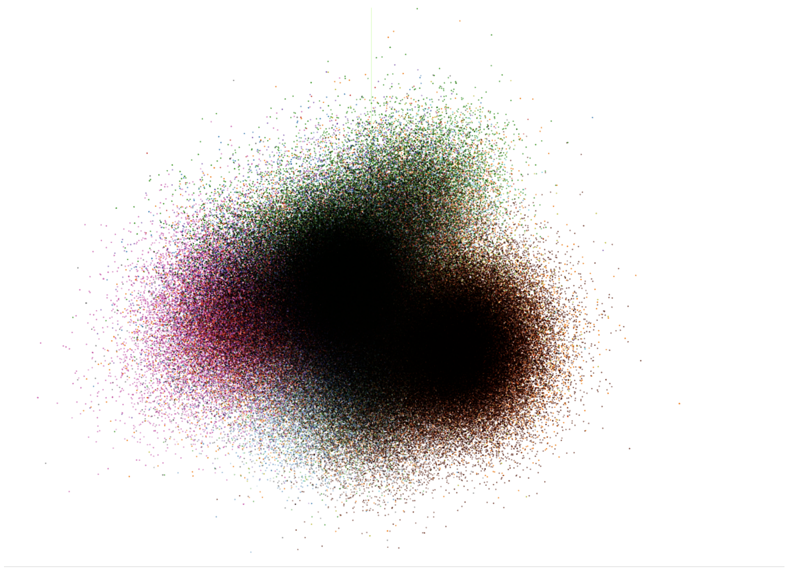


Figure 5.5: 300 Dimensions.

5.2 Sample Analysis

We have selected the 25 dimension vector that is visually more convincing and used it in the sample analysis. The following are some of the interpretations that was performed on seeing the visual output of the 25 dimension vectors.

- The cases that have similar topic names are clustered together
- The euclidean distance and cosine similarity helps to identify the cases having same word distribution.
- It becomes easy to identify any case that might have similar word distribution in Overview text across millions of cases within a fraction of a second.
- We can apply filters on the cases and identify the cases that happened in any month or date across the United States to compute their similarity.
- We identified cases that are far in spatial distribution and figured the reason for this long distance in space.

For researchers interested in reproducing the vectors and view them visually, the vectors are located in the webpage of the authors of this project.

The vector file and the metadata file generated during this stage is located in the project repository.

CHAPTER 6: ADVANCED MACHINE LEARNING

In this section, we have explained a hypotheical concept to compute the embedding for the judges using the datasets prepared in this research. Throughout the process of extracting the data from the source, performing data mining on top of the data, building a clean dataset and performing machine learning we have built multiple datasets in the voyage. To summarize the datasets that have been built during the process, the below diagram represents the relationship between multiple datasets that we have built.

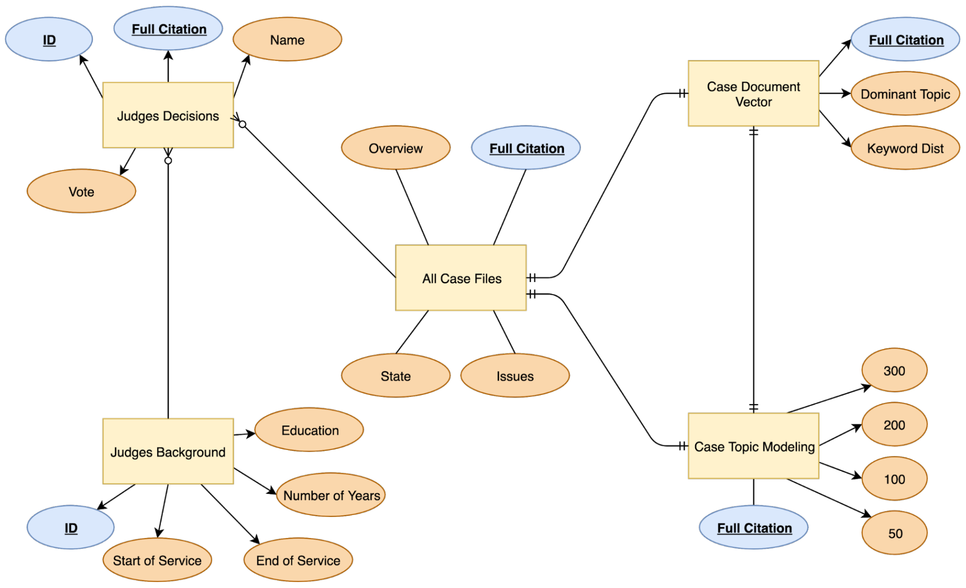


Figure 6.1: Advanced Machine Learning.

In this section, we have explained one method of computing an embedding for the judges involved in the Supreme Court based on the cases they have handled during their service period mentioned in the Roaster File.

Since we have a rich source of data from multiple sources, it becomes easy for us to link between multiple datasets as shown in the image. We propose a method to construct the embedding for the judges in the Supreme Court Dataset.

As we have the cleaned dataset, we could build a supervised machine learning task to predict if a judge has given a decision to consur, dissent or opinion for a particular case. In this task, ut primary goal will not be to develop the most accurate neural network, but to generate the best embedding. The prediction task is just the method through which we train our network to make the embeddings. At the end of training, we are not going to be testing our model on new data, so we don't need to evaluate the performance. Instead of testing on new data, we'll look at the embeddings themselves to see if judges that we think are similar have embeddings that are close to each other.

If we kept a separate validation / testing set, then we would be limiting the amount of data that our network can use to train. This would result in less accurate embeddings. Normally with any supervised model, we need to be concerned about overfitting, but again, because we do not need our model to generalize to new data and our goal is the embeddings, we will make our model as effective as possible by using all the data for training. In general, always have a separate validation and testing set (or use cross validation) and make sure to regularize your model to prevent overfitting.

The next step is the most technically complicated but thankfully fairly simple with Keras. We are going to construct the neural network that learns the entity embeddings. The input to this network is the (judges background, case details, vectors of case, topic) as integers, and the output will be a prediction of whether judge gave a decision of opinion, concur or dissent. However, we're not actually interested in the prediction except as the device used to train the network by comparison to the label. What we are after is at the heart of the network: the embedding layers, one for the judges and one for the link each of which maps the input entity to a 50 dimensional vector. The layers of our network are as follows:

- Input: parallel inputs for the case details, judges info, topic modelling and case vectors.

- Embedding: parallel embeddings for the case details, judges info, topic modelling and case vectors.

- Dot: computes the dot product between the embeddings to merge them together
- Reshape: utility layer needed to correct the shape of the dot product
- (Optional) Dense: fully connected layer with sigmoid activation to generate output for classification

After converting the inputs to an embedding, we need a way to combine the embeddings into a single number. For this we can use the dot product which does element-wise multiplication of numbers in the vectors and then sums the result to a single number. This raw number (after reshaping) is then the output of the model for the case of regression. Using this method for combining the embeddings means we are trying to make the network learn similar embeddings for books that link to similar judges.

Once the embedding is prepared, the weights in the neural network represent the information regarding the judges. We extract this weight and make of it in the visualization. We could build similar visualization of the judges in the n dimensional space using the tensorboard platform.

CHAPTER 7: CONCLUSIONS

The new dataset prepared in this projects supports numerous promising avenue of research. By using the automated Data Mining technique, we have synthesized inexpensively, quickly and reliable data for cases happened from 1953 to 2014 across 50 states. We have explained about the data collection process that produces the most reliable outputs than hand-coding. Also, we have demonstrated an interesting analysis on the dataset that was prepared to analyze the trend in the number of cases that happen over months, the number of cases across each regional reported and the number of cases across each state. Finally, we have applied two cutting edge concepts in Natural Language Processing like the topic modelling and embedding for each case that makes the study more interesting. The embedding prepared, reduces the limitations of traditional encoding methods and can be efficiently used for the purposes such as finding the nearest neighbour, input into another model, and visualizations. Also, we have proposed a method to identify the judges similar to each other based on the cases they have handled and their background information.

Even Though, we have not worked on burdensome multivariate analysis of the data, our initial investigations prove that variation across states might be partly explained by institutional factors such as judicial selection mechanism. We suspect that multivariate time-series-cross-sectional analysis will reveal the influence of social, political, economic, and demographic factors.

Our ultimate goal is to open a domain of research in the State Politics by interconnecting with various data related to judges, state legislature, governors to map a common space of ideological positions in the American States. By interconnecting data on these various political actors, as well as data from national political institutions, we can begin to track the evolution of policy making across institutions and over time to better understand how institutions shape behavior, and how behavior influences policy creation.

REFERENCES

[1] C. Warren, *The Supreme Court in United States History.* Cosmic Inc, 2011.

[2] P. Brace, K. Sims-Butler, K. Arceneaux, and M. Johnson, “Public opinion in the american states: New perspectives using national survey data,” *American Journal of Political Science*, pp. 173–189, 2002.

[3] L. Langer, *Judicial review in state supreme courts: A comparative study.* SUNY Press, 2002.

[4] H. J. Spaeth and J. A. Segal, “The us supreme court judicial data base: Providing new insights into the court,” *Judicature*, vol. 83, p. 228, 1999.

[5] P. Brace and M. G. Hall, “State supreme court data project,” *Data file. URL: http://www. ruf. rice. edu/pbrace/statecourt*, 2007.

[6] P. Brace and B. D. Boyea, “State public opinion, the death penalty, and the practice of electing judges,” *American Journal of Political Science*, vol. 52, no. 2, pp. 360–372, 2008.

[7] B. D. Boyea and V. A. Farrar-Myers, “Leadership and election litigation in state supreme courts,” *State and Local Government Review*, vol. 43, no. 1, pp. 17–31, 2011.

[8] R. L. Vining Jr and T. Wilhelm, “Measuring case salience in state courts of last resort,” *Political Research Quarterly*, vol. 64, no. 3, pp. 559–572, 2011.

[9] C. W. Bonneau, “Impartial judges? race, institutional context, and us state supreme courts,” *State Politics & Policy Quarterly*, vol. 9, no. 4, pp. 381–403, 2009.

[10] P. Brace, M. G. Hall, and L. Langer, “Placing state supreme courts in state politics,” *State Politics & Policy Quarterly*, vol. 1, no. 1, pp. 81–110, 2001.

[11] H. M. Kritzer, P. Brace, M. G. Hall, and B. T. Boyea, “The business of state supreme courts, revisited,” *Journal of Empirical Legal Studies*, vol. 4, no. 2, pp. 427–439, 2007.

[12] P. Brace and M. G. Hall, “Integrated models of judicial dissent,” *The Journal of Politics*, vol. 55, no. 4, pp. 914–935, 1993.

[13] R. P. Caldarone, B. Canes-Wrone, and T. S. Clark, “Partisan labels and democratic accountability: An analysis of state supreme court abortion decisions,” *The Journal of Politics*, vol. 71, no. 2, pp. 560–573, 2009.

[14] D. W. Romero and F. Sanders Romero, “Precedent, parity, and racial discrimination: A federal/state comparison of the impact of brown v. board of education,” *Law & society review*, vol. 37, no. 4, pp. 809–826, 2003.

[15] M. E. Hall and J. H. Windett, “New data on state supreme court cases,” *State Politics & Policy Quarterly*, vol. 13, no. 4, pp. 427–445, 2013.

[16] D. J. Hand, “Data mining,” *Encyclopedia of Environmetrics*, vol. 2, 2006.

[17] C. C. Aggarwal and C. Zhai, *Mining text data.* Springer Science & Business Media, 2012.

[18] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[19] Y. Bengio, Y. LeCun, *et al.*, “Scaling learning algorithms towards ai,” *Large-scale kernel machines*, vol. 34, no. 5, pp. 1–41, 2007.

[20] Y. Bengio, A. Courville, and P. Vincent, “Representation learning: A review and new perspectives,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[21] Y. Bengio, “Deep learning of representations: Looking forward,” in *International Conference on Statistical Language and Speech Processing*, pp. 1–37, Springer, 2013.

[22] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent dirichlet allocation,” *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[23] A. F. Smith and G. O. Roberts, “Bayesian computation via the gibbs sampler and related markov chain monte carlo methods,” *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 55, no. 1, pp. 3–23, 1993.

[24] A. M. Dai, C. Olah, and Q. V. Le, “Document embedding with paragraph vectors,” *arXiv preprint arXiv:1507.07998*, 2015.

APPENDIX A: DATA ACCESS

The centralized dataset developed for all the states in the United States can be accessed at the personal webpage of Dr.Jason Windett, Department of Political Science, university of North Carolina, Charlotte - `https://jasonwindett.com` or in my homepage accessible at `https://kaddynator.github.io`

We have multiple files for the project location in the web repository mentioned above.

1. Supreme Court Dataset
2. Data Description File
3. TSV Vector File of the cases containing Overview text
4. Metadata file of the cases containing Overview text

APPENDIX B: DATA DESCRIPTION

This section contains the description for each of the column in the Dataset.

Table B.1: Data Description

Data Column Name	Description
state	Name of the State where the case was conducted
casecount	Number of words in the Case Document
caseword	Number of words in the case Document
caseparty1	Case Party 1 in the case
caseparty2	Case Party 2 in the case
court	The court in which the case happened
fullcitation	The Full citation number of the case
statereported	The State Reporter number of the case
regionalreporter	The Regional Reporter number of the case
lexisnumber	The Lexis number of the case
month	The month on which the case happened
day	The day on which the case happened
year	The year on which the case happened
judges	The judges involved in the case
subsequenthistory	The history of the case, before the case arrived the current court
priorcourt	The court in which the case was previously heard
priorjudge	The judges involved in which the case if it was previously heard
proceduralposture	The procedural posture text body
overview	The overview text body
outcome	the outcome text body
opinionwriter	the opinion writer of the case
dissentwriter	The dissent writer of the case
issues1,..	The issues identified in the case