British Columbia Judicial Decisions Analysis Project plan

Group members: Niki Hajmoshir, Ilana Zimmerman, Ravi Gill

Description

As of today there is currently no reliable data on what is the amount of compensation a court awards to injured people in British Columbia. When a negligence case wins, the court awards what are called 'damages'. These damages are the financial amount the injured person is paid from the person who injured them. In this project we are trying to analyze B.C. court negligence cases from the past 20 years and gather some metadata such as whether the damage awards have gone up or assess Length of the decisions made.

Datasets

There are two data sources available for all relevant B.C. court cases. The two sources are LexisNexis and WestLaw Connect. Data is in .rtf format. This data source consists of the full text of 8170 cases. These 8170 cases consist of 2195 cases from WestLaw; and 5975 cases from LexisNexis. This data source is roughly 50,000 pages long. Each case consists of a case summary, Regulations and Rules, counsel, Reasons for Judgment and a conclusion.

Expected Deliverables

Overall Deliverables

In general, the question our project hopes to answer is how negligence payouts have been changing since the year 2000 in BC. Our core deliverable for the project is to pull out relevant fields from a collection of negligence cases. Relevant fields include things like the amount of damages awarded and what type of damage was awarded. It will also include whether the case found there to be contributory negligence. The full list of possible fields to extract can be found below.

Potential Deliverables

Depending on how challenging the dataset is to work with there is potential to improve upon our single core deliverable. We can potentially do some analysis of the data we have pulled out

such as tracking how damages change year over year in things such as payout amounts, successful cases, total cases, etc.

Another potential deliverable is to be able to provide a user-friendly application that is able to parse legal documents that are in a similar format to extract the same information for future cases. This would be very useful for use in the future when there are more cases that are publicly reported. This would allow a non-technical user to continue to track how negligence cases in BC change over time.

Lastly, given the time it is also possible to create a web interface for the data we extract. The web interface would allow for individuals to be able to search and filter through the cases.

As the capstone project proceeds we will keep in touch with our capstone partner to determine which of these potential deliverables is most useful for them. This allows us to keep an open mind about what to do after our initial data extraction problem has been dealt with.

Information Extraction Fields

Below is a comprehensive list of fields we will attempt to extract from the case data. Currently we do not know which of these items will be able to be pulled out of the data with minimal error but our plan is to try to cover as many of these fields as possible.

Per case we will extract:

- Case Number (String)
- Case Name (String)
- Written Decision? (Y/N)
- Plaintiff Wins? (Y/N)
- Multiple Defendants? (Y/N - could be changed to an integer)
- $ Damages total before contributory negligence (float)
- $ Non-pecuniary Damages (float)
- $ Pecuniary Damages Total (float)
- $ Special Damages Pecuniary (float)
- $ Future Care Costs (General Damages) (float)
- $ General Damages (float)
- $ Punitive Damages (float)

- $ Aggravated Damages (float)
- Contributory Negligence Raised ? (Y/N)
- Contributory Negligence Successful ? (Y/N)
- % Reduction due to contributory negligence (float)
- $ Reduction due to contributory negligence (float)
- $ Final Award after contributory negligence (float)
- Judge Name (String)
- Decision Length in paragraphs (int)
- Registry (String)

Methods

Data Handling

We first converted all .docx files to .txt files for ease of manipulation in Python. This includes all text and tables present in the original files. Tables from the original document will be stored outside of the .txt file for reference. Each file consists of roughly 50 individual negligence cases, separated by an “End of Document” marker. We plan to store the raw text files on our local machine and have pushed a zip file of the data to Github. Due to the limited volume of data, further database storage/management is not a concern. As part of the final deliverable we will likely export a CSV file with all relevant fields filled in. To the best of our knowledge, we will use the following Python libraries: docx, pandas, numpy, re, matplotlib/altair, sklearn, and nltk.

Static Information Extraction

To start the information extraction process we are picking up on patterns present in all files and individual documents using regular expressions. Many of the fields listed above are easy to extract however, due to variation in structure of the case files, a simple regular expression will not work for extracting the damages awarded in each trial. To verify accuracy of the text extracted for the more straight-forwarded fields of interest (i.e. Registry, Judge Name, Case Number, Decision Length, and Case Name) , we will write tests to ensure the result adheres to the expected format and manually review a handful of results relative to the original documents.

Identifying Damages Awarded

To locate the damages awarded for each case, if any, we will build a simple classifier. First we will need to explore the texts and identify features that will likely differentiate between final damages, partial damages, and other numerical values mentioned. A few features that come to mind are: context in terms of n-grams before and after the dollar value, location in the text of the value is mentioned, and whether or not it is the last value mentioned. After identifying key features related to damages awarded (dollar values), we plan to hand-annotate 100+ results for training. To start we plan to use a tree-based classifier such as a simple Decision Tree or XGBoost. To further explore the structure of damages awarded and other categories of damages (non-pecuniary, specific, and general), we may explore topic modeling using Latent Dirichlet Allocation using the context surrounding the different values.

Code Quality Assurance

For testing our code and performing code reviews, we will have a streamlined procedure in place to ensure code is working as intended prior to finalizing. Although we are each starting with our own experimental Jupyter Notebooks for exploring the texts and information extraction ideas, we will piece together an organized and commented notebook composed of code that is fully functional. This code will have been successfully applied to all documents (over 8,000), which is a test in itself, but will also contain appropriate asserts and will be reviewed by all members of the team via a Github pull-request prior to merging to the master branch of our repository.

Schedule

Meeting schedules

The group will talk everyday at the end of the day for 30 minutes to give updates on work progress and the task that has been checked off. The group will talk to the capstone supervisor, Julian Brooke, on Wednesdays from 10-11 am. The group and the subject matter expert, Lachlan Caunt, will talk on Fridays 10-11 am.

Work schedule

The team's word schedule is designed in a way that most of the information extraction should be done by week 3. Week 4 is to place all team's code together and try to create an interface

where users can easily visualize graphs and charts. Week 5 schedule is not yet determined since the paste of work is not obvious, however if everything goes according to plan we will dedicate that week to extract more useful data and improve the interface. Below is a detailed schedule for each week and assigned tasks for each team member. In week 6 and 7 as suggested by our mentor we will be focusing on writing the final report and preparing our presentation.

Table of schedules for each group member for next 5 weeks.

Group member	**Week1 (May 4 - 11)**	**Week2 (May 11 - 18)**	**Week3 (May 18 - 25)**	**Week4 (May 25 - Jun1)**	**Week5 (Jun1 - 8)**
Goal of the week	1- Finishing the project plan 2- Turning data from .doc to .txt	1- Extract static information 2- Extract damages	- Try decision tree approach for damage extraction	- placing everything together- start on interface, graphs, charts	TBD
Ravi	Expected Deliverables .Doc to .txt	static IE	(all will work on this)	placing everything together	TBD
Ilana	Methods	static IE	(all will work on this)	interface	TBD
Niki	Description Dataset schedules	static IE	(all will work on this)	Graphs and visualization	TBD