# Chicago Kent Law School Project

*Chirac Prasad, IIT Student, Alexandre Gallo, IIT Student, Martin Berger, IIT Student, Antoine Gargot, IIT Student, Thomas Boot, IIT Student*

[1]IIT, Illinois Institute of Technology, 3300 S Federal St, Chicago, IL 60616

***Abstract –*** ***Nowadays, with the advance of data science, it is possible to extract new information out of large amount of data. Since a lot of court decisions relies on previous judgments and is often associated with similar case, it would be interesting to create a graph that enables us to see the similarities and the links between cases. The goal of this project is to realise a model that will enable us to predict the outcome of a trial. We will create a graph that enable to emphasise some legal aspects. We will focus on opinions (as nodes) and create links between them depending of the citations used ( edges). This method should provide us with a directed graph. In order to create such a graph, we will use the court listener data collected by the free law project.***

## I. PROBLEM OVERVIEW

In today's age , it is not easy for a lawyer to figure out how likely he may win a trial. The main goal of this project is to develop a tool that enables us to asses what are the similarities between trials to enable us to predict the outcome of a new one. Since judgment are often based on previous court decision, it will enable us to have a better idea of the possible outcome of a judgment.

## II. DATA

### I) Data Source

For our project, we used the CourtListener's API in order to find most of our needed information regarding the opinions and the citations.

The Courtlistberner API can be found at the following address: https://www.courtlistener.com/api/rest/v3/

Our data mainly consist of JSON files that we collected through the REST API. A typical JSON files consist of these main informations:

The resource url, the absolute url, the cluster url, the author, the creation date, the modification date, the opinion cited, the document in a plain text format, the document with html tag.

The entire data is huge, it is more than 33 GB when it is compressed. It contains about 4 millions opinions splitter between the different Federal Courts of the United States. It can't be processed on a regular computer. We plan to work with a sample of the data to start with and to develop the a first version of the model. Once the first version of the model will be developed, we will try to make it work on AWS. We will build our sample dataset with the supreme court dataset and the 9th court of appeal.

### II) Data Format

The data format consist of JSON files that contains informations about opinions made by different US courts.

You can find below a standard representation of one of the JSON files that we worked with.

```
{
  "resource_uri": "http://www.courtlistener.com/api/rest/v3/opinions/1087897/",
  "absolute_url": "/opinion/1087897/lopez-v-monterey-county/",
  "cluster": "http://www.courtlistener.com/api/rest/v3/clusters/1087897/",
  "author": null,
  "joined_by": [],
  "author_str": "",
  "per_curiam": false,
  "date_created": "2013-10-30T03:00:31.312681Z",
  "date_modified": "2017-12-06T01:11:11.815765Z",
  "type": "010combined",
  "sha1": "741a62c4dbe35939c8950a3639e697755f79420d",
  "page_count": null,
  "download_url": null,
  "local_path": null,
  "plain_text": “ENTIRE DOCUMENT IN PLAIN TEXT”,
  “html": “ENTIRE DOCUMENT WITH HTML TAGS”,
 html_lawbox”:,
 "html_columbia": null,
  “html_with_citations”:”Doc with html tags and special tags for citation",
 "extracted_by_ocr": false,
  "opinions_cited": [
   "http://www.courtlistener.com/api/rest/v3/opinions/108227/",
   "http://www.courtlistener.com/api/rest/v3/opinions/109263/",
   "http://www.courtlistener.com/api/rest/v3/opinions/109595/",
   "http://www.courtlistener.com/api/rest/v3/opinions/109927/",
   "http://www.courtlistener.com/api/rest/v3/opinions/110510/",
   "http://www.courtlistener.com/api/rest/v3/opinions/110688/",
   "http://www.courtlistener.com/api/rest/v3/opinions/110877/",
   "http://www.courtlistener.com/api/rest/v3/opinions/111098/",
   "http://www.courtlistener.com/api/rest/v3/opinions/112611/",
    "http://www.courtlistener.com/api/rest/v3/opinions/117968/",
    "http://www.courtlistener.com/api/rest/v3/opinions/2141794/"
  ]
}
```

In such a JSON file the main informations are:

The name of the file which represent the opinion number.

The “opinion cited” value is also very important because it is where we can find all the the citation in the document. Those citations enables us to create links between the opinions. Furthermore, the “plain text” value contains all the informations that are in the original document. The “html ” version of the document is also a key element, since it contains the same document with tags on important parts.

## III) Programming Language, packages

We will mainly used Python to realise the project.

We wanted to realise most of the project in R but Python turned out to offer more solutions in terms of information scrapping and network analysis. In order to simplify the usage of our data we turned those JSON file into a CSV. However, we wanted to do it in R at first but because of a problem with the RJson package, we had to use python to do so.

The package that we used in Python:

- **networkx** for graph analysis
- **nltk** for language processing
- **pandas** for dataframe handling
- **sklearn** for K-Means and Silhouette curves
- **marplotlib** to plot the different data
- **numpy.linalg** to make gaussian blobs for clustering.
- **re** to analyse regular expressions
- **beautifulsoup** for html tag scrapping
- **json** for JSON handling and conversion
- **urllib2**
- **collections** to be able to use counter

These libraries enabled us to make great visualisation for our project.

For the natural language processing part we will use the NLTK.

We started to look how to use R with AWS with the Hadoop Package and shiny server but because we ended up coding most of the project in python, we will have to reconsider a new option to implement python on AWS.

# III. PRELIMINARY EXPERIMENTS

## I) Preprocessing

The dataset is split in different folders depending on which court the trial was related too.We decided to focus our work on the Supreme Court of the United States and the 9th Court of Appeal.

For both court, we used a python script to create a cvs sample file over 5000 files.

Looking into our samples, we noticed that the data was very inconsistent. Indeed, some of the data file had the opinion report with html tags in it and others didn’t. When they didn’t have the report with the html tags in it, they still had the court listener document as a plain text format.

At this moment we realised that the information extraction would be a real challenge and that we would have to create different algorithms depending on whether it would have the information as a plain text or with html tags.

## II) First Conclusions

The information extraction process is the corner stone of our project. Indeed, the reliability of our future analysis will depend on the quality of our information extraction algorithms. We will have to split the work in order to make sure that we cover the different possibilities.

# IV. EXPERIMENTS

## I) Selected approach

Knowing that the information extraction process is the corner stone of tour project, we decided to mainly focus on information extraction.

The goal we are aiming now is to scrap every JSON in the data and extract some specific information. We would like to make a structured synthesis of each opinion. For each opinion, we would like to extract: the case number, the date, the related citations, the name of each parties, the order of the case, name of the judge, which court it is happening in.

## II)Implementation

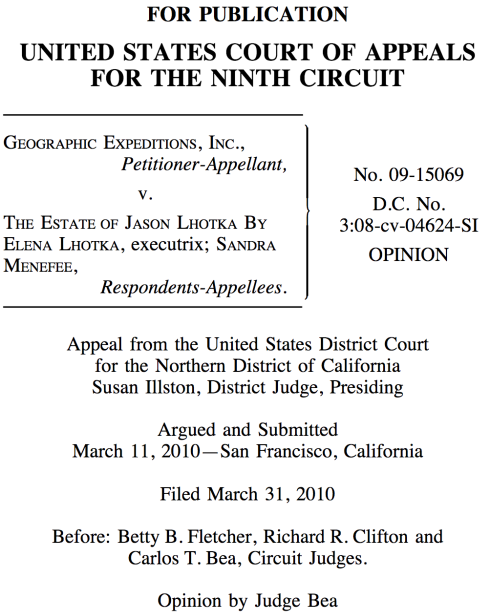
In order to extract the rest of the information that is missing, it is required from us to learn how to apply the text information retrieval course that we had.

For this task, we will most probably work with the NLTK, Re and Beautiful Soup4 python library. Indeed NLTK offers a wide range of function for text processing and tagging.

# IV. EXPERIMENTS & RESULTS

## I) Plain text scrapping

Plain text scrapping is very challenging due to the instructed nature of the documents. The first step consisted in analysing a large amount of report and to find where the key information was in the reports. As we can see below, the first page of each report contains a lot of the information that we would like to extract.



FOR PUBLICATION

UNITED STATES COURT OF APPEALS
FOR THE NINTH CIRCUIT

GEOGRAPHIC EXPEDITIONS, INC.,
*Petitioner-Appellant,*
v.
THE ESTATE OF JASON LHOTKA BY ELENA LHOTKA, executrix; SANDRA MENEFEE,
*Respondents-Appellees.*

No. 09-15069
D.C. No. 3:08-cv-04624-SI
OPINION

Appeal from the United States District Court
for the Northern District of California
Susan Illston, District Judge, Presiding

Argued and Submitted
March 11, 2010—San Francisco, California

Filed March 31, 2010

Before: Betty B. Fletcher, Richard R. Clifton and
Carlos T. Bea, Circuit Judges.

Opinion by Judge Bea

**Plain text version of the above document:**

*' FOR PUBLICATION\n UNITED STATES COURT OF APPEALS\n FOR THE NINTH CIRCUIT \n\nGEOGRAPHIC EXPEDITIONS, INC., \uf8fc\n Petitioner-Appellant,\n No. 09-15069\n v.\nTHE ESTATE OF JASON LHOTKA BY \uf8fd D.C. No.\n 3:08-cv-04624-SI\nELENA LHOTKA, executrix; SANDRA\n OPINION \nMENEFEE,\n Respondents-Appellees.\n \uf8fe\n Appeal from the United States District Court \n for the Northern District of California\n Susan Illston, District Judge, Presiding\n\n Argued and Submitted\n March 11, 2010—San Francisco, California\n\n Filed March 31, 2010\n\n Before: Betty B. Fletcher, Richard R. Clifton and\n Carlos T. Bea, Circuit Judges.\n\n Opinion by Judge Bea\n\n\n\n\n*

For each document, the first page often has the same format so we focused our analysis through each line of the document. We decided to narrow down to each line because we could target more efficiently each information.

On the first sentence, we focused on detecting the court of appeal for which each document belongs to. This has been done using by tokenising the first sentence and detect if one of the following word was in it: *["FIRST", "SECOND","THIRD","FOURTH","FIFTH","SIXTH","SEVENTH","EIGTH","NINTH" ]*

We also noticed that the opinion number of the document was often in the second sentence. Therefore we analysed the second sentence tried to detect the following regular expression: *r"[0-9][0-9]-[0-9][0-9][0-9][0-9][0-9]"*

In the sentence number four, we can often detect the important dates related to the case. We extracted the dates by detecting the following regular expression: *r"((January|February|March|April|May|June|July|August|September|October|November|December) [0-9]+, [0-9][0-9][0-9][0-9])"*

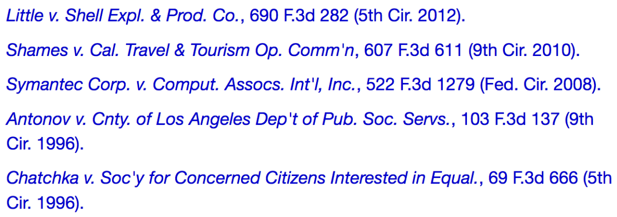
We can also notice that their are often the participating judges in the same sentence. Therefore, we detected the word "*Before:*" in the sentence and took the end of it to have the list of the people in it.

To finish with the first page analysis, we analysed the fifth sentence and were able to detect the name of the opinion judge. We did so by detecting the word *"Opinion by: "* and taking the end of the sentence.

In the third sentence, we could supposedly detect the petitioner and the respondent but it is a complicated task because there are a lots of dots for abbreviation that often disturb de creation of a pattern. Furthermore, the name of a petitioner or a respondent can often have several different patterns which complicate the task.

We also tried to extract the verdict out of each document. We notice that the verdict was often located in the last sentence of the document so we parsed the last document and looked for the following regular expressions: *r"(AFFIRMED|REVERSED|REMANDED|REVERSED IN PART)"*.

One of the main goal was also to detect each situation in a document. However they can take several different shape:



*Little v. Shell Expl. & Prod. Co.*, 690 F.3d 282 (5th Cir. 2012).

*Shames v. Cal. Travel & Tourism Op. Comm'n*, 607 F.3d 611 (9th Cir. 2010).

*Symantec Corp. v. Comput. Assocs. Int'l, Inc.*, 522 F.3d 1279 (Fed. Cir. 2008).

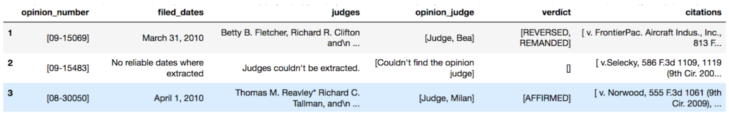
*Antonov v. Cnty. of Los Angeles Dep't of Pub. Soc. Servs.*, 103 F.3d 137 (9th Cir. 1996).

*Chatchka v. Soc'y for Concerned Citizens Interested in Equal.*, 69 F.3d 666 (5th Cir. 1996).

Therefore, we scrapped the entire document in order to detect the following regular expression:

'(  v\.([0-9]|\-|\,|  |[A-Z]|[a-z]|\.|\')*(\([0-9]+(th|d|  )+Cir. [0-9]+\)))'

## II) Plain text scrapping results



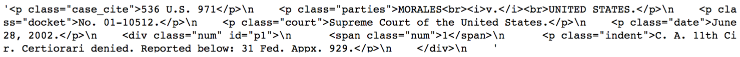
| | opinion_number | filed_dates | judges | opinion_judge | verdict | citations |
|---|---|---|---|---|---|---|
| 1 | [09-15069] | March 31, 2010 | Betty B. Fletcher, Richard R. Clifton and\n ... | [Judge, Bea] | [REVERSED, REMANDED] | [ v. FrontierPac. Aircraft Indus., Inc., 813 F... |
| 2 | [09-15483] | No reliable dates where extracted | Judges couldn't be extracted. | [Couldn't find the opinion judge] | [] | [ v.Selecky, 586 F.3d 1109, 1119 (9th Cir. 200... |
| 3 | [08-30050] | April 1, 2010 | Thomas M. Reavley* Richard C. Tallman, and\n ... | [Judge, Milan] | [AFFIRMED] | [ v. Norwood, 555 F.3d 1061 (9th Cir. 2009), ... |

We can notice that the searching process is quiet efficient when the format of the document is "standard". However for a lot of documents, it couldn't find the totality of the information. This is due to the fact that the structure of the document can be altered just by one dot.

This algorithm is yet to be improved by making it more generalised to all sorts of documents.

## III) HTML with citation scrapping

In order to scrap the data in the document version with html tags, we scraped the HTML for each opinions in order to create a new data frame based on HTML class.



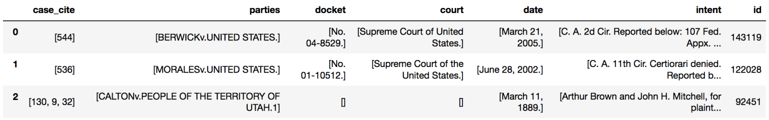
```
'<p class="case_cite">536 U.S. 971</p>\n    <p class="parties">MORALES<br><i>v.</i><br>UNITED STATES.</p>\n    <p cla
ss="docket">No. 01-10512.</p>\n    <p class="court">Supreme Court of the United States.</p>\n    <p class="date">June
28, 2002.</p>\n    <div class="num" id="p1">\n      <span class="num">1</span>\n      <p class="indent">C. A. 11th Ci
r. Certiorari denied. Reported below: 31 Fed. Appx. 929.</p>\n    </div>\n    '
```

**Example of a html document:**

We used the BeautifulSoup function findAll() to collect the different informations that we were looking for. The HTML class that we collected are the followings: *case_cite, parties, docket, court, date, indent.*

Once we scrapped all these information, we turned it into a data frame.

## IV)HTML with citation scrapping results



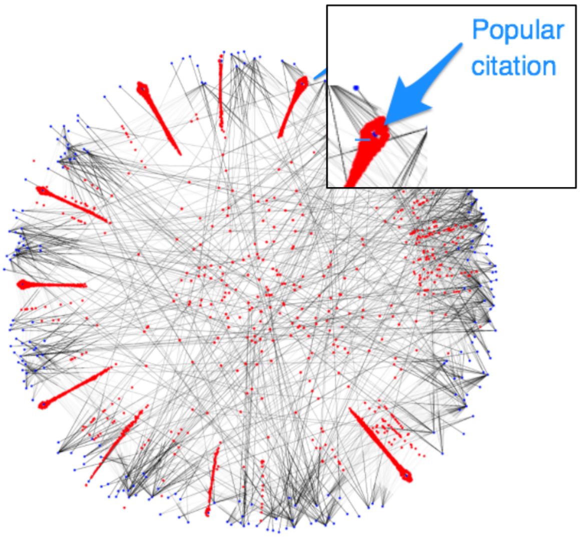
| | case_cite | parties | docket | court | date | intent | id |
|---|---|---|---|---|---|---|---|
| 0 | [544] | [BERWICKv.UNITED STATES.] | [No. 04-8529.] | [Supreme Court of United States.] | [March 21, 2005.] | [C. A. 2d Cir. Reported below: 107 Fed. Appx. ... | 143119 |
| 1 | [536] | [MORALESv.UNITED STATES.] | [No. 01-10512.] | [Supreme Court of the United States.] | [June 28, 2002.] | [C. A. 11th Cir. Certiorari denied. Reported b... | 122028 |
| 2 | [130, 9, 32] | [CALTONv.PEOPLE OF THE TERRITORY OF UTAH.1] | [] | [] | [March 11, 1889.] | [Arthur Brown and John H. Mitchell, for plaint... | 92451 |

As we can see above, with this technique we managed to get accurate results. However, sometimes, the HTML tags were empty or did not contain the right information. This inconsistency in the HTML tag is a source of noise in our information extraction process. It is therefor important to cross validate the results that we obtained using the HTML scrapping method with the plain text scrapping method.

# V. GRAPH ANALYSIS

## I) Generating a graph

For this part, we collected all the citations that we had for every cases. Each opinion will be represented by a node. Whenever an opinion cite another one, we will draw a directed edge between going from the opinion to the cited node. Every node in red will be nodes that are opinions in our dataset. However, all the dark nodes, will represent a opinion that is cited but that is not in the data base.

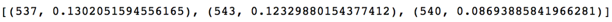


## II)Graph Analysis

We can notice that the most of the blue nodes are on the outskirt of the graph. Furthermore, we can also notice that some of the node are widely cited in the opinions in the dataset. When there is a large agglomeration of dark edge around a blue node, we can see that this node is fairly important. However when a lot of red dots end up around the same blue node, this node becomes extremely important.

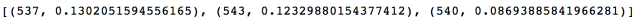
While studying the graph, we noticed that the closeness centrality and the in degree centrality had the same values.It is due to the fact that our graph is a directed one. Here we can see the nodes that have the highest in degree centrality and the one that have the highest closeness centrality. They represent the cases that are the most cited.

<u>In degree centrality:</u>



```
[(537, 0.1302051594556165), (543, 0.12329880154377412), (540, 0.08693885841966281)]
```

<u>Closeness centrality:</u>



```
[(537, 0.1302051594556165), (543, 0.12329880154377412), (540, 0.08693885841966281)]
```

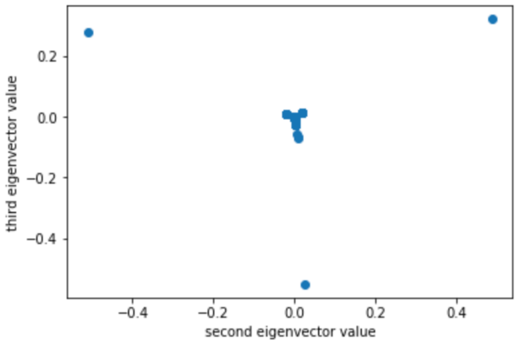
We also computed the nodes that have the highest out.degree centrality in order to determine which cases are citing the most the other cases.

Out degree centrality:

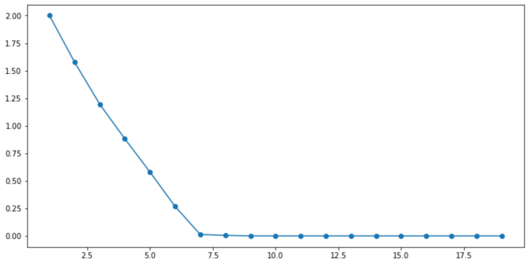
```
[('92451', 0.0006093845216331506), ('99504', 0.0006093845216331506), ('85081', 0.0006093845216331506)]
```

## III) Cluster Analysis

We calculated the directed laplacian matrix of the case cited in the graph and we could notice the different clusters possible in our graph. This enabled us to visualise our citations on a 2d plan. We can notice that there are several cluster possible. Probably 5 or 6.

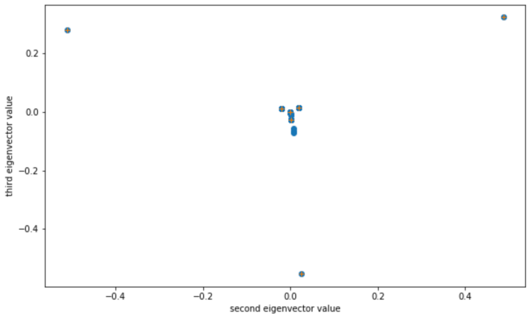


We still did an Eclust analysis to see what would be the appropriate amount of cluster in our graph.



As we can see, the appropriate amount of cluster in the graph given by the Eclust analysis is 7( we look at the elbow part of the curve). Therefor, it would be possible to group all the citation into 7 clusters.

We than applied a K-Means Clusering algorithm with k  to 7  based on the 2 dimensional representation that we had above:



As we can see on the K-Means representation, we clearly have 7 clusters well defined.

The fact that we are able to clearly identify clusters of citations would help us to build a solid predictive model for the next stage of the project.

## VI. UNCONCLUSIVE TRIES

During our data scrapping part, we also tried other different methods that were inconclusive. We tried to use the nltk package and do some name entity tagging in the documents but it wasn't tagging items properly and we couldn't clearly define to which individual the names related to.

## VII. CONCLUSION

This is a project with high potential. It was a very challenging project that enabled us to discover the real challenges that can face a data scientist and how complex can information extraction become.

In order to reach our final goal ”predict the outcome of a trial” we still need to improve the information extraction process.

We could do different type of analysis and start to look into different type of correlations  in our data. May be look if there is a correlation between the verdict and the judge or the month of the year .

We could also perform LSA over the order of a new case in order to detect which cases have the closest order to it.

Many improvement can be added to our work and I strongly think about following up on it during my spare time .