



# Civera MassCourts

04.07.2021

---

Ammar Ahmad  
Sherry Courington  
Megan Mastorilli  
Shihab Karim  
Serra Jung

## Project Introduction

Civera is a software company seeking to liberate public data by making it easier to access and understand. The state of Massachusetts court system has a website, MassCourts.org, containing its public data. However, this website is unorganized and difficult to navigate, rendering it, and the data, useless to anyone trying to access it.

Civera has scraped the entire data from the official website into a database. They have designed an alternative website using that database of court data. However, to complete this project, they need the data to be in a standardized and clean form.

Hence, we were tasked with standardising the raw data from the MassCourts website into a new more readable dataset. Our main task is to standardize the database and perform analysis on the court data.

## Dataset

The data set is client's raw data for MA housing court dockets. It includes four tables that are stored in a MySQL database hosted on a server.

## Working with Data

The dataset that we have for this project is huge (in GBs). There are four tables each with millions of rows. We are using the following ways to interact with the data

- Connecting to the data server using MySQL workbench and writing queries on MySQL workbench.
- Connecting to the database server using python and reading the data using queries. We sometimes retrieve the result of the query in chunks since the size of the data is huge.
- Downloaded the tables as a csv file and reading the csv file using python.

## Tasks and Progress

Task	Status	Comments
Converting PHP regex to Python	Complete	Current code outputs an actor from a given match (Github existing code is

		hardcoded for the judge actor), might need to be modified/hardcoded for certain actors. Can modify to output action
Normalizing the database (fill missing action/actor values)	In Progress	
Analysis on Normalized Database	To do	

## Methodology for Normalizing the Database

Each Case in the raw dataset has different case actions that happened at different times. Moreover, each case has a case description. Using these case descriptions, we can infer the case action and the case actor. We were tasked to extract the case action (what happened) and case actor (who completed the action) for every entry.

The main task in normalizing the database is to fill in the missing action and actor values for the case actions. Each case action has a one line description from which we can extract or infer the case action and actor. Therefore, this becomes an information extraction task.

The client had done some of the extraction manually using regex. But since the data is so huge, we attempt to do this in an automated manner. To solve this problem, we first analyze the text and then try both supervised and unsupervised approaches based on our initial analysis of the data.

## EDA and Semantic Analysis of Action Descriptions

We analyzed the raw case descriptions and made the following important observations

- There are almost 30-40 million case actions for which we need to extract the actor and action information.
- Almost all case descriptions are 1-2 lines long. However, these lines are not in perfect english grammar and do not follow the same pattern.
- Some case descriptions did not even include the actor's name in the descriptions. E.g. in some case descriptions where the actor is "Court", the

word “Court” does not appear in the descriptions. This makes the problem more difficult.

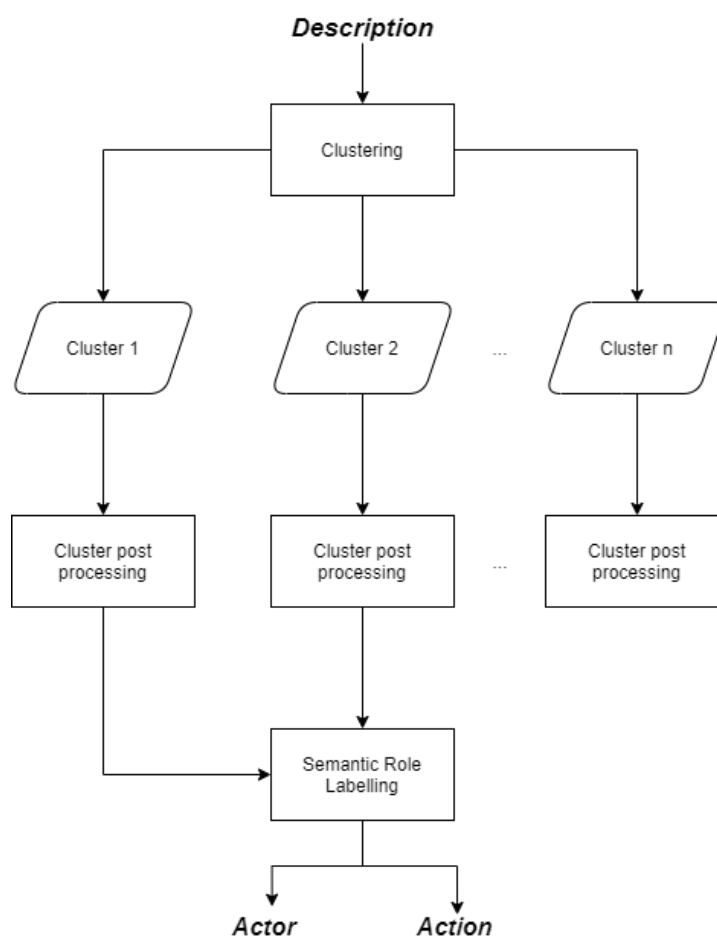
- Semantic Analysis done using NLP libraries like Spacy and NLTK was not very helpful in gaining insights into solving the extraction problem at hand because most case descriptions are written in passive voice sentences and generally NLP algorithms do not perform well on passive voice english sentences.

In light of the above observations, we decided to try both unsupervised and supervised approaches to solve this task.

## Unsupervised Approach

In the unsupervised approach, we first cluster the text into different clusters or categories and then apply some approach to extract the actor and action from descriptions.

We are attempting to design the following pipeline



According to this pipeline we perform the following steps

1. Clustering the descriptions to different clusters
2. Cluster post processing
3. Semantic Role Labelling

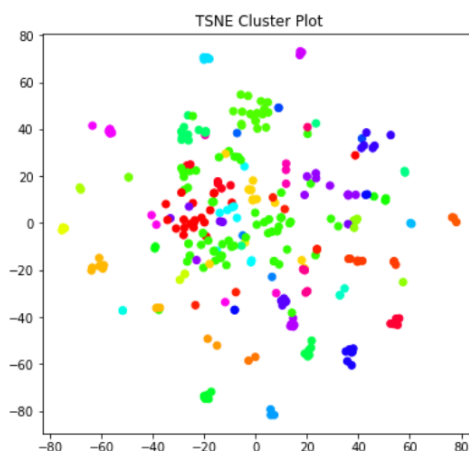
## Clustering

Our intuition is that all the case actions of the same kind should fall into the same cluster. Also we think that there will be many clusters which will be similar semantically, so we can use the same approach to extract the case actions from them.

To perform clustering on these text descriptions, we have to convert the text into a form which can be clustered using different clustering algorithms such as K Means clustering, Hierarchical Clustering and Community Detection Clustering. Each text description has to be converted to a set of numbers. So we tried the following methods to perform this conversion

- TFIDF Features

These are features that are generated according to the count of each word in the complete dataset.



- Glove Embeddings

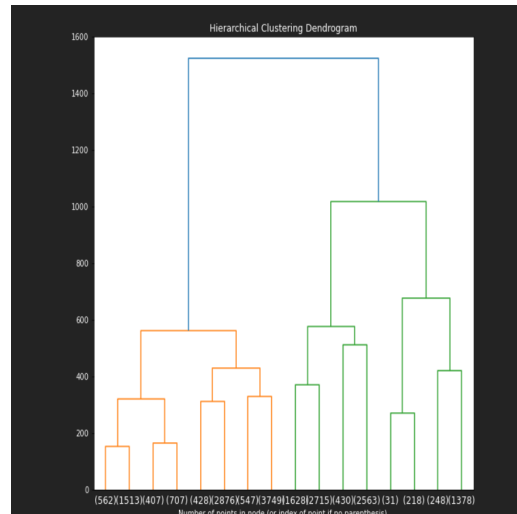
Pre trained glove embeddings used to convert each text description into its equivalent representation.

- Fasttext Embeddings

These embeddings were trained by Facebook, We use these pretrained embeddings as well.

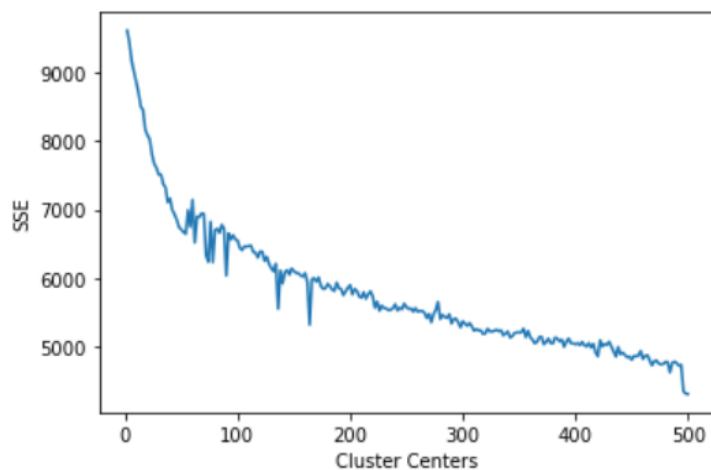
- POS (Parts of Speech) Features

These representations were made by converting each word into its part of speech.




### Choosing the optimum number of clusters

One of the typically large challenges in clustering is choosing the optimum number of clusters. We utilized the traditional method referred to as the elbow method. With this method we tried a numerical range of clusters and plotted the Sum Squared Error, SSE (*the difference between each observation and the mean of the cluster*) generated by each cluster on a graph. The elbow, or the break in the graph, indicates the point where it is no longer advantageous to create additional clusters. An illustration is shown below:



### Cluster Post Processing



We observed that our clustering using different methods was very good but not perfect. So to improve the clustering results, we will add a cluster post processing step. In this step, we take all sentences in a cluster and exclude the sentences from the cluster that are different from the majority of sentences because theoretically we should have similar sentences in a cluster.

## Extraction using Semantic Role Labelling

We saw that most clusters were in passive voice and identified some common passive voice patterns. To extract action and actor from these clusters of sentences, we are going to use an NLP approach called Semantic Role Labelling. We are going to use a state of the art pretrained machine learning model for this task.

## Supervised Approach

In the supervised approach, we tried to perform classification on distinct case actions and case actors. We attempted to design models that predict what case action and actor each description belongs to.

To do this we again had to convert the text descriptions to numbers using TFIDF features. We tried the following models

1. Linear Singular Value Decomposition on TFIDF Features
2. Multinomial Naive Bayes on TFIDF Features
3. Random Forest on TFIDF Features

## Imbalanced Data Problem

One big challenge in training supervised models is the imbalanced dataset that we have. This means that there are very few examples of some case actions and actors while a very large number of examples for other case actions and actors. This imbalance is inevitable given the nature of the data. We first attempted to solve this problem by creating custom balanced training set for our models, where for each distinct case action, we took 10% of its rows to create a training set. Assuming the remaining unclassified data had similar ratios of common and uncommon actions, we thought this would work. Another way we attempted to solve this data imbalance was by taking exactly 10 rows from each distinct case action to create a training set. However, both of these methods continued to result in a low accuracy score on all three models.

## Text Summarization by fine tuning a Transformer Model

Transformers are huge machine learning models that are trained for working with language data. They are usually trained on a huge data set for learning representations of a language and then fine tuned (further trained on a small dataset) for a particular task. Because of the data imbalance problem and the fact that it is difficult to know the distinct

actions that exist, we change the problem from a classification task to a text summarization task. In a text summarization task, a model is trained to summarize a piece of text in few words. In our case, the description would be the input text and the output would be its summary aka its action. By framing our task in this way, we might be able to leverage the general language understanding abilities of transformers.

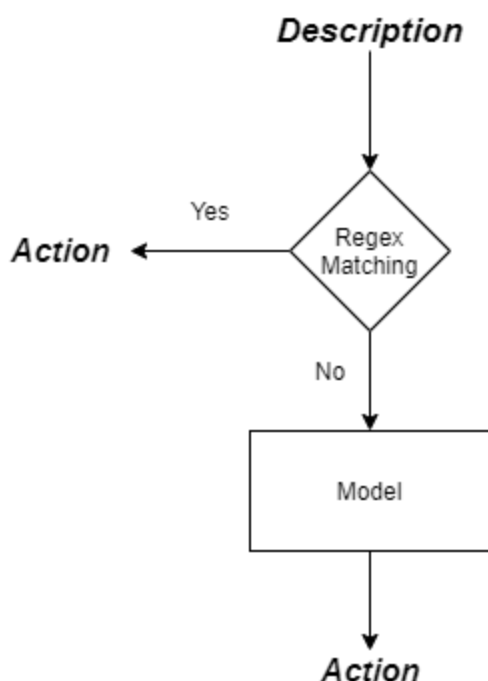
For this task, we use T5 - Text to Text Transformer from Google. We fine tune it on approximately 400K examples. After fine tuning we input the raw descriptions, and it outputs the action.

## Regex and Pattern Matching

For some descriptions, it is easier to just match regex patterns to extract actions. These actions and actors are straightforward. We were also provided with some regex patterns that were already written for this task in PHP code. We converted those to python regex.

## Combined Approach

After trying all the above approaches, we came to the conclusion that a combined approach involving the regex and the supervised approach of a fine tuned text summarization transformer works the best. A description is first matched with regex patterns and if a pattern does not match then the description is given to the trained model and the model outputs the action.





## Limitations and Challenges

The limitations and challenges faced are described below:

- The immense size of raw data makes it difficult to try different approaches since it takes a very long time to process the complete dataset. To deal with this, we have tried using our approaches on smaller subsets of the data. However, since we sometimes are unable to perform a technique on all the data, there is a certain amount of guesswork involved which is eliminated when we test our approach on the whole dataset. But that testing takes a long time.