Developing a dataset and using it with ML and NLP to schedule Indian supreme court cases.

Malhar Kajale[1], Rahul Motwani[2], Mohit Shahdadpuri[3], Nimish Chidrawar[4], Dr. Sujata Khedkar[5]
[1-5]Department of Computer Engineering, Vivekanand Education Society's Institute of Technology, Mumbai, India
[1]2020.malhar.kajale@ves.ac.in,[2]2020.rahul.motwani@ves.ac.in,[3]2020.mohit.shahdadpuri@ves.ac.in, [4]2020.nimish.chidrawar@ves.ac.in, [5]sujata.khedkar@ves.ac.in

Corresponding author: Rahul Motwani

***Abstract*— India is the largest democracy in the world and the judiciary is a prime pillar. However, there is a lack of unified, preliminary, legal datasets openly available in India to help research methods to expedite court case disposal. Legal data attributes of supreme court cases are available in isolation across various sources which require unconventional methods of extraction. The extraction and exploration of such data are discussed. India also has a very low judge-to-population ratio which is a major cause for the huge pendency of cases in the Supreme Court. In such a scenario, the quick and unbiased (not based on the type of case), regular disposal of the cases becomes critical. Such disposal of the cases is not availed using the current procedure of scheduling cases. In this study, machine learning has been leveraged to achieve the same by creating an automated schedule generation system. The case attributes of filing date, number of petitioners and respondents are used for a classification model to yield priority class labels for the incoming cases. The output of the classification model is passed to the case scheduling algorithm for determining a batch of court cases. The case statutes of each scheduled case are used along with a case type assigning algorithm and judge experience metrics based on cases handled per case type to assign the most experienced judges. Thus, this study reduces the entry barrier for upcoming researchers and showcases a Machine Learning based application of the data procured for efficient case disposal.**

Keywords—ML, NLP, NER, Web Scraping, Legal, Judicial

1. INTRODUCTION

The different data attributes of the Indian Supreme Court cases are present in isolation at various sources, making them unfit for further study of any kind. Making the data structured and unified will enable us to explore the use of machine learning in the judicial domain [1]. The existing studies across different domains show none [5] to least efforts [8] in describing the sources, attributes and procuring methodology of the data. Judicial data has to be extracted from various sources which is tedious and an unconventional approach is required. As a result, all the challenges that are faced in the dataset preparation phase remain unaddressed.

The data is analyzed for the count and type of cases based on their linguistics and accessibility. The different data fields are also explored in case they could lead to some more important data fields that could be used as important attributes. Techniques like web scraping [6] using Selenium WebDriver and Named Entity Recognition (NER) in Natural Language Processing [3, 4, 12] have been used to procure the data attributes. Two NER models have been trained using spaCy to identify and extract two key attributes - statutes and the judge names from the case's judgment text that is derived from the case PDFs using the PyPDF2 library.

The scheduling process of the Indian Supreme Court cases lacks an objective procedure. This leads to several cases being scheduled in a non-standard manner. This is in the disinterest of the Indian judiciary where an already huge number of cases are pending for their disposal. Building an equitable system for schedule generation will result in a more standard explanation for the case schedule and will constitute the minimal waiting time for justice delivery for each case type.

In the existing study [1], the case categories of all the incoming cases are considered as an important factor for scheduling them which leads to a bias as one is perceived as more critical than others based majorly on its category. But in actuality, the priority of the court cases is highly subjective and differs from case to case [2]. Thus, for scheduling a majority of the incoming cases, a transparent and explainable approach is preferable. In the proposed solution, an ML approach has been followed via a supervised machine learning classification model that has been used to classify the cases to be scheduled based on their filing date, number of petitioners and respondents. The classes of the cases so generated are consumed by a scheduling algorithm to build a cases' batch to be listed such that it contains most types of classes.

The scheduled cases are then assigned to the most experienced judges based on the case type of the case. The case statutes of the cases are utilized by a case type assigning algorithm to determine the case type. The case type is then looked up into the judge experience metrics for the most experienced judges for that case type and the same are assigned based on their availability. Also, the judge's allocation based on their experience will lead to better decision-making. This system serves as an appropriate foundation that could be easily extended to include more specific mechanisms so as to enable a more optimal case disposal methodology.

2. LITERATURE REVIEW

Manjeet Singh et al. [1], this paper aids in providing a basic understanding of the Indian legal system, which is a crucial tenet of democracy. It suggests a system for prioritizing matters so that they can be resolved more quickly and the judiciary can function more effectively.

Sivaranjani, et al. [2] the objective of this paper is to classify the cases and to predict the behavior of the court i.e whether the appeal case will be accepted or dismissed by the Supreme Court using CNN and HCNN, this paper gives the individual an ideology about how the case will be taken and treated by the court.

Jeong, M et al. [3], at large-scale equipment firms, maintenance staff must precisely and rapidly comprehend the maintenance equipment papers to accomplish the jobs. In order to overcome issues like the inclusion of words with semantically ambiguous terms as they may impair the maintenance performance of engineers, this work seeks to extract vital information or knowledge from these papers.

Nadeesha Perera et al. [4], named entity recognition and relation detection, is used in this study to identify relationships between proteins and medications, or between genes and diseases, and to consolidate extensive information on specific clinical or biomedical issues that are accessible for further analysis.

Bruno Mathis et al. [5], the open data programme for all court cases in France are covered in this essay. This document claims that there are 26 labels in court proceedings (rare and ubiquitous). In order to extract judgment data, this article examines various algorithms, including CRF, Spacy, Flair, and DeLft, and makes use of Kairntech's learning model assessment capabilities.

Chaimaa Lotfi et al. [6], this paper discusses advanced web scraping techniques which could help scholars and managers to learn how to mine online data most effectively.

Cartic Ramakrishnan et al. [7], the difficulties that developers of biomedical text mining or biocuration informatics systems have while trying to extract text from PDF files while taking into account the file's layout are covered in the study. The paper introduces the LA-PDFText system, which seeks to simplify accurate text extraction from research publications in PDF format for usage in text mining applications, in order to overcome this issue.

Shahmin Sharafat et al. [8], this research paper focuses on the extraction of entities such as dates, case numbers, reference cases, person names, respondent names etc. In order to automatically extract these entities, the primary requirement was to construct a dataset using legal judgments.

Gathika Ratnayaka et al. [9], this research paper is based on identifying the discourse relationships among the sentences in Court Case Transcripts. The information obtained helps the lawyers and the judges when performing their duties based on previous judgments.

Sahan Jayasinghe et al. [10], identifying critical sentences, facts and arguments in a legal case is a tedious task for legal professionals, in this research, they explore the usage of sentence embeddings for multi-class classification to identify critical sentences in a legal case, in the perspective of the main parties present in the case.

Xin Jin et al. [11], this paper propose an algorithm realizing the automatic loading and unloading operation in the storage of electric power materials under unmanned conditions with simulation experiments. It presents us with an efficient scheduling system that satisfies the needs of intelligent warehouse management.

Keet Sugathadasa et al. [12], this paper focuses on domain-specific semantic similarity. This project was created by the synergistic union of word2vec, a word embedding method that is used for semantic similarity calculation and lexicon-based (lexical) semantic similarity methods.

George Wanganga et al. [13], this paper presents a novel customer payment service request scheduling algorithm via matching request priority with the best personnel to handle the request based on data analytics through Machine Learning to improve SaMS-PSP's customer payment service requests processing speeds, personnel optimization, throughput, and low latency scheduling.

The structure and semantics of the legal text are very diverse. This diversity is even greater if the considered-legal PDFs belong to a huge timeline. To apply NER to such pdfs for data extraction, appropriate training data is required to train the NER model in terms of the count and diversity of the data. But it is a challenge to supply annotations in such a huge quantity using the conventional way to annotate and no studies discuss a way to complete the annotating procedure efficiently. This aspect is covered in our study by using a semi-automatic annotating procedure. There is a lack of concreteness in current studies to mention both the data sources and the data extraction methodologies for the set of legal data attributes.

Scheduling in India doesn't have any determined process. It is subjectively performed by different judges which leads to unjustifiable methods of schedule generation in the name of prioritization of certain types of cases. This leads to some types of cases perceived as less important which gets delayed justice. No particular research has been done on creating an automated schedule generation system for the judiciary by utilizing machine learning. Also, there is no mechanism that allows the optimized allotment of the judges based on their skills and experience so as to deliver better case clearance rates and judgment.

3. METHODOLOGY

1.1. Dataset preparation

1.1.1. Description and observation on the Initial Dataset

The initial dataset was obtained from justice hub [10] which was in CSV format consisting of 11158 rows, where each row represented a case and consisted of the following fields: bench of judges, diary number, case number, judgment by the judge, and an incomplete URL (Uniform Resource Locator) to the supreme court case judgment document open to public access on their website. After observation, it was concluded that the obtained data would not be sufficient for further research as important fields such as date of filing, date of hearings/listings, statutes used in the case, the number of petitioners, respondents and prosecution witnesses were not present.

1.1.2. Removal of irrelevant documents and transformation of attributes

Due to the linguistic diversity present in India, there were some documents written in regional languages but not in sufficient quantity to leverage them for research, a total of 10 such cases had to be removed from the dataset. A type of (incomplete) URL in the dataset was non-functional and had to be altered by trial and error so as to access the associated judgment document. The results of the above modifications were prepended with the supreme court website's subdomains for all the existing URLs and the existing field values acted as the subdirectory in the URLs to access the supreme court case judgment document.

Table 1: Document distribution in dataset

Total documents	Vernacular documents	Erroneous documents		Valid Documents
		Unsolvable	Resolvable	
11,158	10	32	7556	3560

1.1.3. Using existing attributes to extract more attributes and filling missing values

To obtain the required missing fields web scraping was performed [5] on the case information available on the supreme court website for the case. The date of filing, date of hearings/listing along with the number of petitioners and respondents were scraped by using selenium to automatically insert the diary number and year by using the existing diary number field of the dataset. A few clicks were simulated on the supreme court website in order to find the required legal attributes and were extracted using the HTML elements' XPath (that would be a consistent value for each case page).

1.1.4. Extracting more attributes via case PDF documents using Named Entity Recognition and RegEx

The structure & semantics of judgment documents were highly dissimilar [7, 9, 10] for applying NER models to find out statutes referenced in a case. To resolve this issue the final model was trained using text without any punctuation separated by single whitespaces. Also, for the text extraction process, the PDFs were opened in the 'non-strict' mode to deal with the differences in their header formats. The spaCy NER model was provided with custom annotations from the preprocessed text using semi-automatic regular expression based annotation generation. A batch size of 200 over 40,000 annotations was used for training the NER model. Similarly, 10,000 annotations for judge names were also generated to identify them from the judgment documents as there were missing values for the field in the original dataset [6]. These judge names were required as there were many missing fields for the attribute in the dataset.

Afterward, the process of extracting the number of prosecution witnesses pertaining to criminal court cases was performed on each individual judgment in the dataset. A regex was created to identify and find all the references of unique prosecution witnesses in a document.be sufficient for further research as important fields such as date of filing, date of hearings/listings, statutes used in the case, the number of petitioners, respondents and prosecution witnesses were not present.

Figure 1 represents the entire methodology followed for the preparation of the legal dataset. The results of these methods were compiled together with the existing dataset for the formation of a dataset that could be utilized for developing a system for scheduling [11, 13] court cases

.

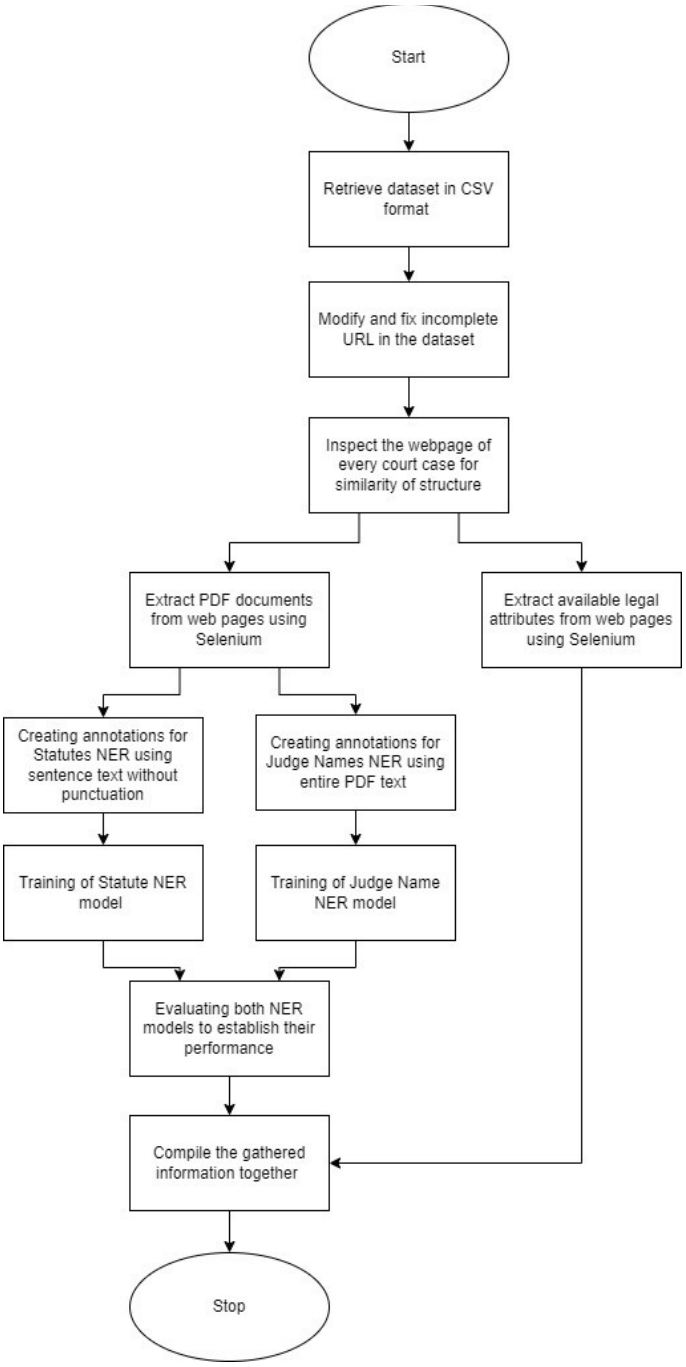

Figure 1: Approach to preparation of preliminary dataset

1.2 Building the case scheduling system

The software architecture of the solution has two modules – the 'Case scheduling module' and the 'Judge allocation module'. The first module achieves the functionality of scheduling the court cases, while the other module assigns judges to the scheduled cases. Both the modules are functionally decoupled from each other, hence were parallelly implemented.

1.2.1 Module 1 - Case scheduling

To increase the accuracy and other evaluation measures for any model it is of utmost importance to have a large amount of data to reinforce patterns that are to be recognized and predicted. The initial number of rows in the dataset were 10115.

The listing dates attribute was split up and each generated element was added as a filing date in the initial dataset creating a new tuple in the dataset with the remaining input features as the original case tuple. This process resulted in expanding the dataset to 40370 rows which was further divided into training and testing sets in the ratio of 4:1 using scikit-learn.

Due to the absence of correlation in the data obtained it was infeasible to train a regression model. Thus, a classification model with a suitable output class label had to be decided. After much deliberation, it was decided to set the range of the output label to range from 1 to 5. This range would propose that the Supreme Court of India would hear a minimum of five cases per day. The decided output label was derived using the existing input feature such as filing dates, number of petitioners and respondents. Filing dates were converted to numeric form by multiplying the numeric year by 365, the numeric month by 12 and adding these two values with the day of the month. The date attribute in integer form would be inverted by multiplying it by -1 so that an older date has a higher numeric value than a newer date giving the older dates more importance while scheduling. All the required and created input features were normalized using min-max scaling to range between 0 and 1 after analysis of the variations in data values. The output label was calculated by adding all the input features and then normalizing the sum to range from 0 to 5 in floating point number which was then ceiled to range from 1 to 5 natural numbers, refer figure 2. After the formation of the required data, a logistic regression classification model was trained using scikit-learn with input features such as calculated filing date, number of petitioners and respondents and the output variable being the calculated label from 1 to 5.

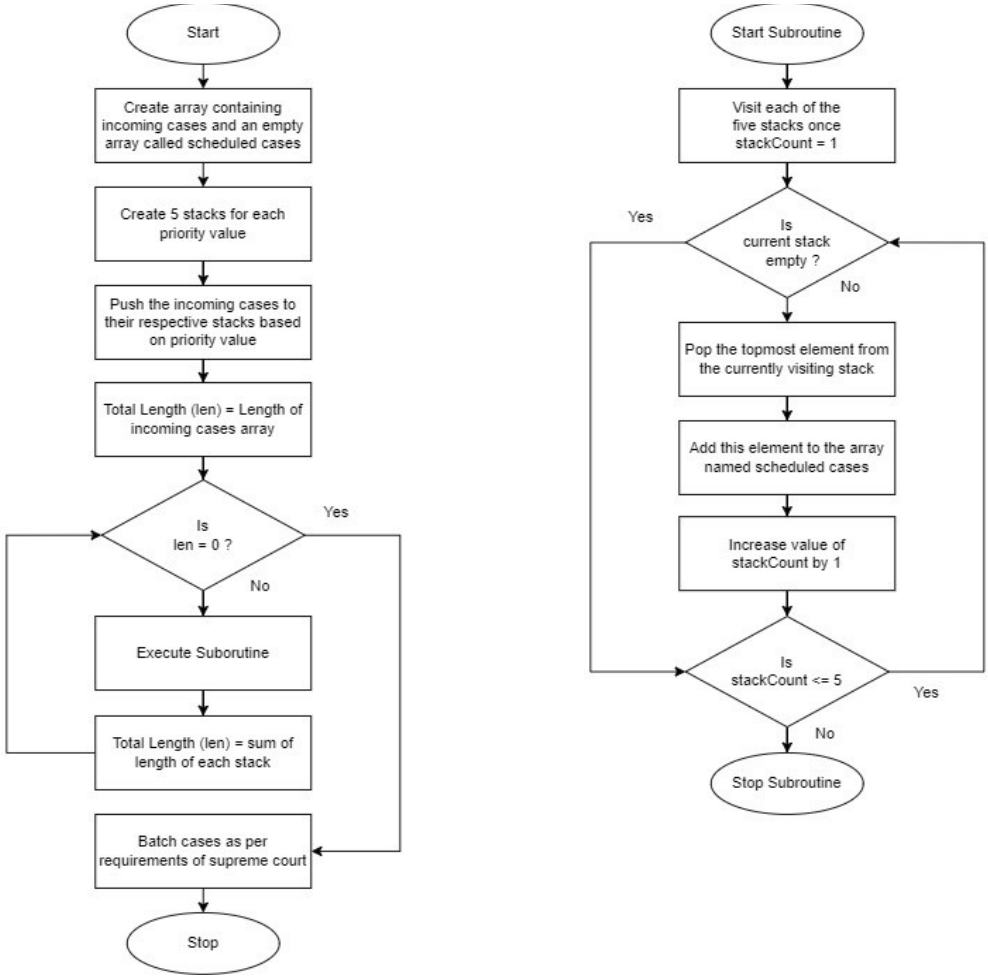

Figure 2: Case scheduling algorithm

The essential logic behind labeling the output from 1 to 5 was that on a single day, a batch of five cases would be generated which would ideally contain all the unique output labels from 1 to 5. A case with an output label of 5 would have higher priority that is during the selection of a batch of five cases an output label of 5 would be given the first priority of selection. While selecting batches an attempt is made to select all unique output labels but scenarios such as the absence of the required output label and the total number of cases in a group to be scheduled is not divisible by the batch size of five are also handled by the scheduling algorithm. If a required priority number is not found for a slot in a batch of five then the next closest lower priority value is found in a group of arrived cases. If there is an overflow of cases to be scheduled then the left-out cases would get an increased priority by a value of one in the next iteration of scheduling taking place.

1.2.1 *Module 2 - Judge allocation*

The attributes of case and case statutes were considered. Another dataset was built by assigning all the possible statutes to their respective case categories following a one-to-one relationship. This assignment was done after studying each statute individually to maintain accuracy throughout the process. The case categories used were referenced from the Indian Supreme Court Case Category Website. The result was a CSV dataset with 18 columns representing 18 case categories and in each column, there were statutes belonging to the same case category.

An algorithm was devised and developed that takes the statutes of each case and for each statute present in the case, it determines the case category by searching for that statute in the case category CSV. While searching for the categories of all the statutes of a case, the counts for those are maintained in a list having 18 indices. The indices of the list were in turn corresponding with the names of the case categories in a one-to-one relationship with the case category name holding list . After all the statutes are searched for a case, the final counts in the list for each category are considered and the maximum count-bearing index is used to select the category as the final case category for the case. This procedure was repeated for each case in the dataset. Thus, 5067 total cases out of 10,463 case records were initially classified using the above-discussed procedure into 18 case categories. The rest of the scenarios needed separate handling.

There were scenarios where a case record contained erroneous non-statutes present in the list of statutes, due to the inaccuracies of the NER model or the PDF text extractor. If these occurrences for a case were in addition to the correct statutes in the list, then the non-erroneous statutes were used to try and assign a case category to the case ignoring the erroneous non-statutes. Else, if there were only erroneous non-statutes for a case, then this was handled by marking the case category as 'not found' after searching the entire case category CSV for the assumed statute's case category. The ‘not found’ values were handled further by eliminating the respective case records from the dataset.

Also, there were scenarios for which there were no statutes identified by the NER model, again due to its inaccuracies, or there might not be any statute present in the case pdf. Those were assigned a ‘null’ value temporarily. Later, the resolution of 805 null valued records out of 3713 such records, which were having their general case categories (criminal and civil) appended in the ‘subdirectory of the judgment PDF’ field of the initial CSV dataset obtained from was done. The rest of the null-valued records were eliminated.

There were also some scenarios in which the category count-holding list for a case contained the same values at two or more indices and that value was the maximum amongst all values present at other indices. For such cases, we cannot determine the case category of the case based on the algorithm written. These records were temporarily handled by assigning a tied’ value as the case category. These ‘tied’ values were later handled by using the probabilities of all 18 case categories. The probabilities for each case category were deduced from the final list holding the count of all the cases in each case category. Following the initial iteration to identify the tied cases and determine the probability of each case category, the algorithm was separately implemented after some changes in the second iteration for those tied cases. Thus, the tied cases were resolved in the second iteration by multiplying each tied category count in the list with the respective case category probability obtained from the results of the initial iteration.

In this way, an output CSV with case and case category columns was obtained. This CSV is considered along with the CSV dataset of case and judge names. And then the judge names are mapped to the case categories in a many-to-many relationship. Then, a CSV dataset with case categories and a judge's name as columns was prepared. This dataset has the experience of each judge per case category based on the cases heard, which can be leveraged for assigning the judges to fresh incoming cases based on the type of case and the availability of judges.

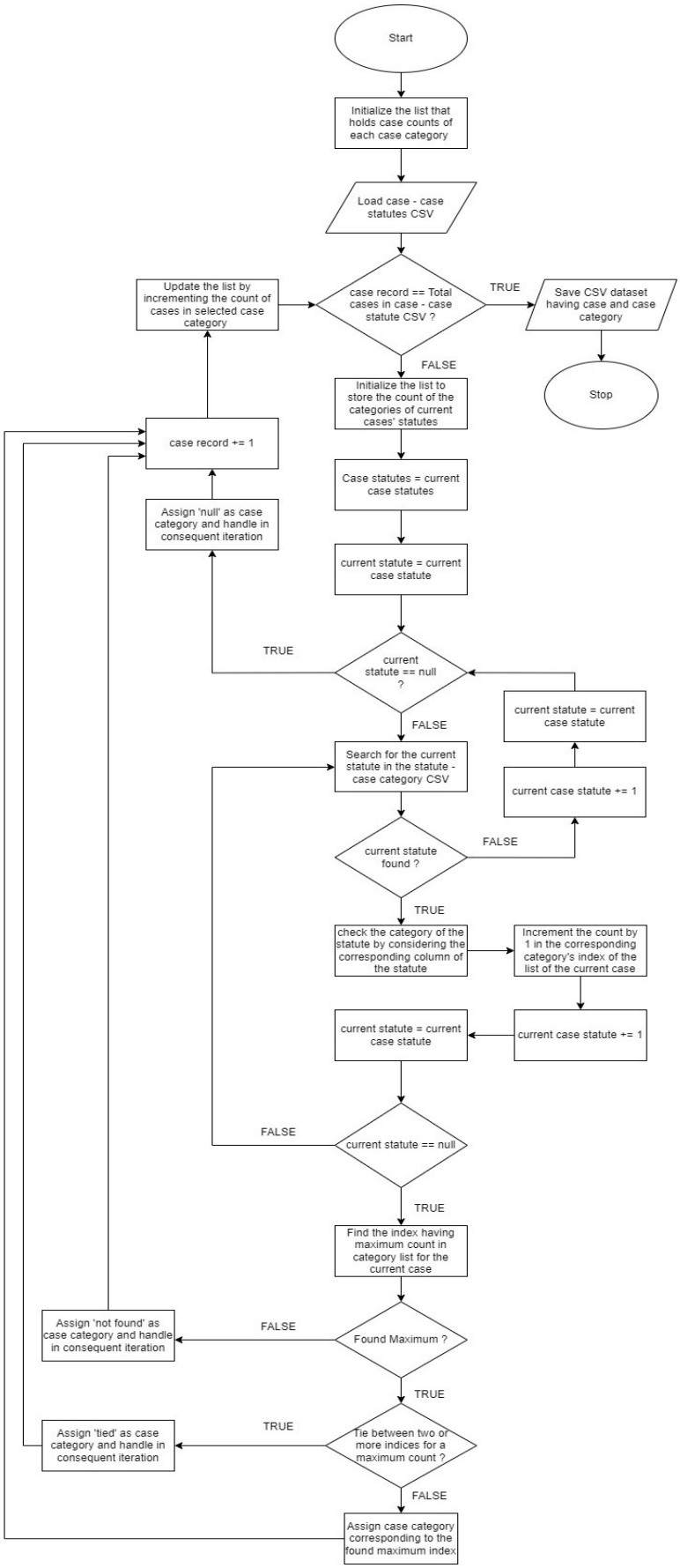

Figure 3: Algorithm for mapping case to case category

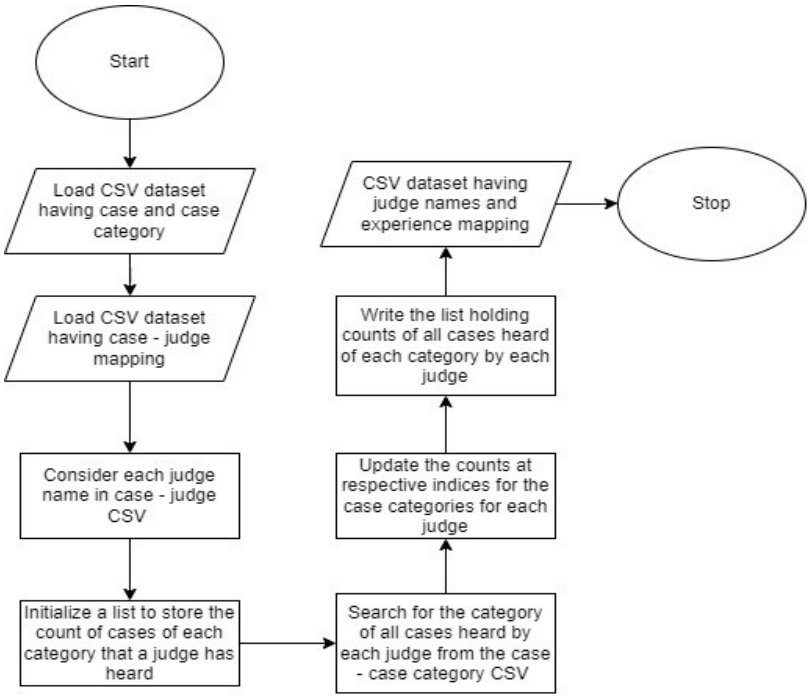

Figure 4: Algorithm for creating experience dataset of judge according to the case categories

4. EXPERIMENTATION AND RESULTS

4.1 Creation of regular expressions for accurate annotations

After observation, it was concluded that judge names for a particular judgment were always found at the end of the document preceded by "J." as a prefix. The regex for the same is as follows.

j\.?(I.)?\s*[\[([a-z\s*\.]*[\])]

Determining statutes from an entire judgment resulted in the formation of a huge regex which also subsequently gave incorrect annotations. The regex for identifying statutes from a single sentence gave better annotations and fewer edge cases. The regex for the identification of statutes from a sentence is as follows.

([A-Z]+[a-z]*\s(and\s|with\s|of\s|for\s)*)+(\(.+\)\s)*([A-Z]+[a-z]*\s(and\s|with\s|of\s|for\s)*)*Act

4.2 Experimentation with Statute Identification models

Table 2: Performance measures of the Statute Identification Model using whole document at once

Performance measure	Value
F-score	0.769
Precision	0.704
Recall	0.846

A model identifying statutes from individual sentences instead of the whole document was developed. The performance measures improved significantly. Although it performed fairly, it was misinterpreting some phrases as statutes. This was due to the punctuation marks present in the sentences identified by the model.

Table 3: Performance measures of the statute identification model using individual sentences

Performance measure	Value
F-score	0.989
Precision	0.987
Recall	0.990

The final model was trained with text separated by white spaces, without any punctuation. Statutes were being recognized by the model accurately.

Table 4: Performance measures of the statute identification model using individual sentences, without punctuation

Performance measure	Value
F-score	0.993
Precision	0.992
Recall	0.994

4.3 Judge Name Recognition Results

Table 5: Judge names extraction using regex

Index No.	Judge Names
1	[‘ABHAY MANOHAR SAPRE’, ‘INDU MALHOTRA’]
2	[‘L. NAGESWARA RAO’, ‘HEMANT GUPTA’]
..	..
..	..
62	[‘ji(supra’,’ji(supra’,’MOHAN M. SHANTANAGOUDAR’]

Table 6: Judge names extraction using NER model

Index No.	Judge Names
1	[‘ABHAY MANOHAR SAPRE’, ‘INDU MALHOTRA’]
2	[‘L. NAGESWARA RAO’, ‘HEMANT GUPTA’]
..	..
..	..
62	[’MOHAN M. SHANTANAGOUDAR’]

4.4 Statute Results

Table 7: Statute NER annotations using regular expressions

Index No.	Sentence	Identified Statutes
1	'Charges were framed against … Tamil Nadu Prohibition of Harassment of Women Act'	['Tamil Nadu Prohibition of Harassment of Women Act']
2	'It is submitted that all the ingredients necessary for conviction … Evidence Act 1872 were …'	['Evidence Act']
3	'The High Court maintained the conviction under … Indian Evidence Act 1872'	['Indian Evidence Act']
..	..	..

Table 8: Statute extraction using NER model

Index No.	Statutes
1	Tamil Nadu Prohibition of Harrasment of Women
2	Central Government Municipalities and Panchayati Raj
3	Evidence#Indian Evidence#Criminal Law Second Amendment#Indian Evidence
..	..

4.5 NER Model Recognition Results

Table 9: Statute identifying NER model performance measures

		Performance Measures		
		F-score	Precision	Recall
NER Model Name	Statute	0.999	0.998	0.999
	Judge	0.996	0.996	0.990

4.6 Mapping case to case category – separate handling after iteration one

Table 10: Results of separate handling of cases after iteration one

Temporary values	Count	Action
Tied	344	Resolved
Null	805	Resolved
	2908	Eliminated
Not Found	1279	Eliminated
Total	5336	

4.7 Mapping case to case category – final categorization after iteration two

Table 11: Count of cases in each category after final iteration

Case Category	Count
Land acquisition, requisition, rent act and eviction act matters	415
Direct taxes matter and indirect taxes matters	509
Labor and service matters and matters relating to judiciary	790
Academic matters and admission/transfer to engineering and medical colleges	168
Letter petition and PIL matters	1
Election matters	67
Company law, MRTP, TRAI, SEBI, IDRAI and RBI and mercantile laws, commercial transactions Including banking & consumer protection	543
Arbitration and compensation matters	216
Criminal matters and state excise-trading In liquor-privileges, licenses-distilleries breweries	772
Appeal against orders of statutory bodies	77
Contempt of court matters	50
Ordinary civil, family law and personal law matters	645
Religious and charitable endowments	34
Matters related to appointments of constitutional functionaries, statutory bodies and of the other law officers	72
Simple money and mortgage matters	10
Land laws and agricultural tenancies	113
Matters relating to leases, govt. contracts and contracts by local bodies and RTI	95
Miscellaneous	490
Total categorized cases	5067

4.8 Exploratory data analysis

Exploratory data analysis was conducted on the dataset to analyze covariance and correlation between data attributes in order to fit a machine learning model for the next listing date prediction.

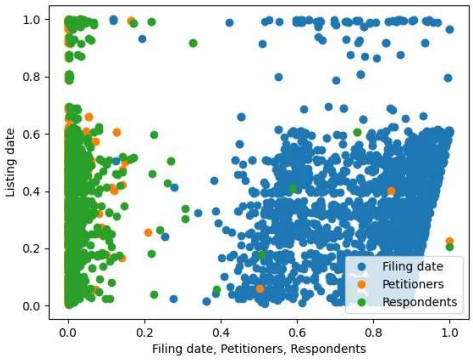

Figure 5: listing date vs filing date, count of petitioners and respondents

The initial data plot as seen in Fig 1 does not depict any sort of relation between data attributes and no satisfactory line of regression which would support a positive correlation could be drawn through it. Here the predicted output variable was the next listing date and the input features were attributes such as listing date, number of petitioners and respondents. From these observations, it was clear that fitting a regression model would obtain poor results.

After the elimination of regression, the next logical option was to try using a classification model for training. A suitable label for the usage of classification was absent and had to be derived in an unbiased manner by excluding attributes such as statute-based case category

For accurate predictions, the training and testing data had to be scaled down in order to maintain the relation between the input features and output variables over a closely similar range of values. Min-max scaling was used to normalize the training and testing data to range between 0 and 1 in floating point values as seen in Fig 1.

Outliers had to be removed from the dataset and the same were identified through their positions in a graph against the hearing date. These outliers can be seen in Fig 6, 7 and 8.

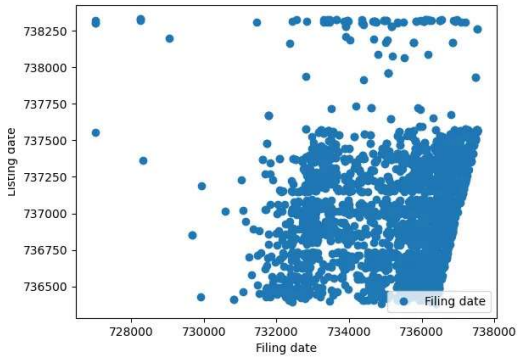

Figure 6: Identification of outliers in listing date vs filing date

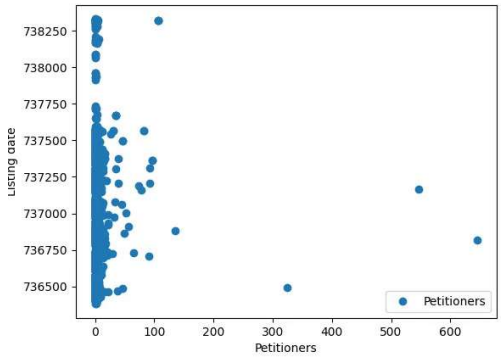

Figure 7: Identification of outliers in listing date vs count of petitioners

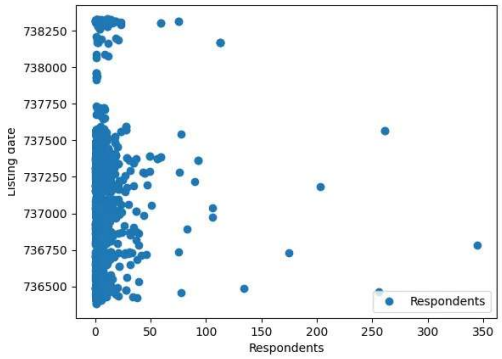

Figure 8: Identification of outliers in listing date vs count of respondents

4.9 Priority assignment to cases

After the assignment of priority values to cases was completed the distribution of cases for each priority value can be seen in Table 11.

Table 12: distribution of cases for each priority value

	Priority Labels				
	5	4	3	2	1
Case Count	2008	2794	10926	16539	8102

4.10 Performance measures of supervised model

The confusion matrix of the logistic regression supervised machine learning model for predicting the priority labels is shown in Fig 9.

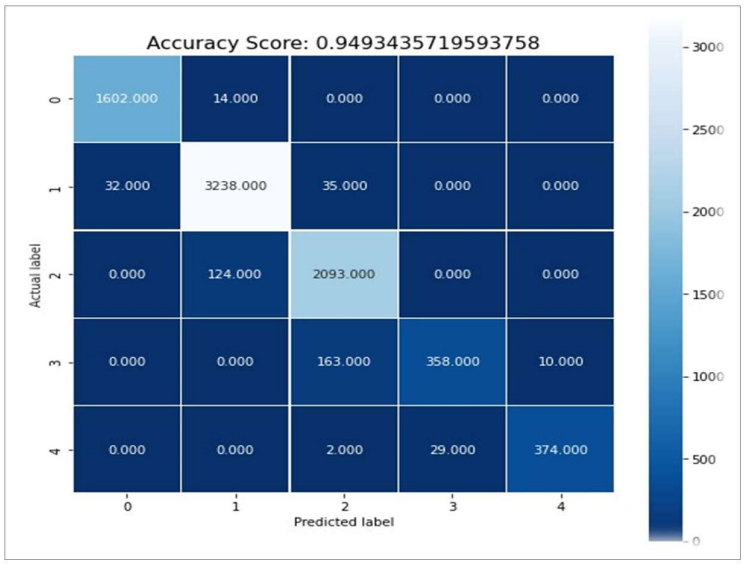

Figure 9: Confusion matrix for logistic regression model

4.11 Precision, recall, F-score and support for each priority label

The performance measures such as precision, recall, F-score and support for each priority label are displayed in Table 12.

Table 13: precision, recall, F-score and support for each priority label

		Performance measures			
		Precision	Recall	F-score	Support
Priority Labels	1	0.98	0.99	0.99	1616
	2	0.96	0.98	0.97	3305
	3	0.91	0.94	0.93	2217
	4	0.93	0.67	0.78	531
	5	0.97	0.92	0.95	405

5. CONCLUSION

The paper displays how the problem of the lack of open datasets for applying machine learning algorithms in relation to the judicial domain is tackled using unconventional methods. An unbiased planning method for the fast and efficient disposal of court cases in India is also discussed. The proposed solution deals with both the assignment of judges to cases and also schedule generation based on the freshly arriving cases and backlog. Case categories are created and assigned to cases using an innovatively developed algorithm which helps in judge allocation performed by calculating experience metrics for each judge and maintaining a database for the same. The scheduling was performed firstly by assigning priority labels to cases by training a supervised machine learning model and then by utilizing the predicted priority values to generate a timetable using an effectively constructed algorithm. In the future, the solution will be enhanced to automatically handle instances by judging urgency.

REFERENCES

[1] Singh, M. (2018). Indian Judicial System Overview and a Approach for Automatic Roster Preparation and Case Scheduling for Faster Case Solving (Need of: E-Courts). 2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN). doi:10.1109/icacccn.2018.8748721

[2] Sivaranjani, N., Jayabharathy, J., & Teja, P. C. (2021). Predicting the supreme court decision on appeal cases using hierarchical convolutional neural network. International Journal of Speech Technology, 24(3), 643–650. Springer Science and Business Media LLC. https://doi.org/10.1007/s10772-021-09820-4

[3] Jeong, M., Suh, H., Lee, H., & Lee, J. H. (2022). A Named Entity and Relationship Extraction Method from Trouble-Shooting Documents in Korean. Applied Sciences, 12(23), 11971. MDPI AG. https://doi.org/10.3390/app122311971

[4] N. Perera, M. Dehmer, F. Emmert-Streib, Named entity recognition and relation detection for biomedical information extraction, Frontiers in Cell and Developmental Biology 8 (2020). doi:10.3389/fcell.2020.00673. URL https://www.frontiersin.org/articles/10.3389/fcell.2020.00673

[5] Mathis, B. (2022). Extracting Proceedings Data from Court Cases with Machine Learning. Stats, 5(4), 1305–1320. MDPI AG. https://doi.org/10.3390/stats5040079

[6] Lotfi, C., Srinivasan, S., Ertz, M., & Latrous, I. (2021). Web Scraping Techniques and Applications: A Literature Review. SCRS CONFERENCE PROCEEDINGS ON INTELLIGENT SYSTEMS (pp. 381–394). Soft Computing Research Society. https://doi.org/10.52458/978-93-91842-08-6-38

[7] Ramakrishnan, C., Patnia, A., Hovy, E., & Burns, G. A. (2012). Layout-aware text extraction from full-text PDF of scientific articles. Source Code for Biology and Medicine ,7(1). Springer Science and Business Media LLC. https://doi.org/10.1186/1751-0473-7-7

[8] Sharafat, S., Nasar, Z., & Jaffry, S. W. (2019). Legal Data Mining from Civil Judgments. Communications in Computer and Information Science (pp. 426–436). Springer Singapore. https://doi.org/10.1007/978-981-13-6052-7_37

[9] Ratnayaka, G., Rupasinghe, T., de Silva, N., Warushavithana, M., Gamage, V., & Perera, A. S. (2018). Identifying Relationships Among Sentences in Court Case Transcripts Using Discourse Relations. 2018 18th International Conference on Advances in ICT for Emerging Regions (ICTer), 13-20. IEEE. https://doi.org/10.1109/icter.2018.8615485

[10] Jayasinghe, S., Rambukkanage, L., Silva, A., Silva, N. de, & Perera, A. S. (2021). Critical Sentence Identification in Legal Cases Using Multi-Class Classification. 2021 IEEE 16th International Conference on Industrial and Information Systems (ICIIS) (pp. 146–151). doi:10.1109/ICIIS53135.2021.9660657

[11] Jin, X., & Yu, L. (2022). Research and implementation of high priority scheduling algorithm based on intelligent storage of power materials. Energy Reports, 8, 398–405. Elsevier BV. https://doi.org/10.1016/j.egyr.2022.03.126

[12] Sugathadasa, K., Ayesha, B., de Silva, N., Perera, A. S., Jayawardana, V., Lakmal, D., & Perera, M. (2017). Synergistic Union of Word2Vec and Lexicon for Domain Specific Semantic Similarity. arXiv. https://doi.org/10.48550/ARXIV.1706.01967

[13] Wanganga, G., & Qu, Y. (2020). An Auto Optimized Payment Service Requests Scheduling Algorithm via Data Analytics through Machine Learning. 2020 International Conference on Computational Science and Computational Intelligence (CSCI) (pp. 1498–1502). doi:10.1109/CSCI51800.2020.00277

[14] [dataset] Arpit Jain, Anubhav Mishra (2020). Supreme Court Cases [2010 – 2020]. https://justicehub.in/dataset/supreme-court-cases-2010-2020/resource/a444c230-fce7-4166-8810-48e888da23e4.