Executive Summary: Predicting US Supreme Court Decisions Using NLP by Jørgen Leiros

Jørgen Maurstad Leiros, as part of his MSc in Business Administration and Data Science, conducted a **Natural Language Processing (NLP) and machine learning study** to predict the **outcomes of US Supreme Court cases** based on **textual case descriptions**. His research assessed whether **modern NLP techniques can assist legal professionals in forecasting judicial decisions**, with potential implications for legal strategy and case selection.

Key Focus Areas:

1. **Problem Statement and Dataset** –
    - The study analyzed **3,303 Supreme Court rulings from 1978-2021**, sourced from the **Oyez database**, focusing on case facts and verdicts.
    - The research explored whether **NLP-based models can predict case outcomes** to assist law firms and policymakers.
2. **NLP Techniques and Model Selection** –
    - **Text Preprocessing:** Data cleaning, tokenization, stopword removal, and feature engineering were applied.
    - **Text Vectorization Approaches:**
        - **Bag-of-Words (BoW)** – Used in the baseline Naïve Bayes model.
        - **TF-IDF (Term Frequency-Inverse Document Frequency)** – Applied in the **Support Vector Machine (SVM)** model.
        - **Word2Vec embeddings** – Used for **LSTM (Long Short-Term Memory) models**.
        - **BERT (Bidirectional Encoder Representations from Transformers)** – Fine-tuned for legal text classification.
3. **Model Performance and Findings** –
    - **Naïve Bayes (Baseline Model)** performed best with **65% accuracy**, outperforming even more complex models.
    - **SVM, LSTM, and BERT models** failed to generalize well due to class imbalance and case description limitations.
    - **Key challenge:** The dataset's label (first-party winner) did not consistently indicate conviction/acquittal, **introducing ambiguity** in model predictions.

4. **Limitations and Challenges** –
    - **Class imbalance and data inconsistencies** reduced model accuracy.
    - **Supreme Court cases often involve complex, constitutional matters**, making predictions inherently difficult.
    - **Overfitting issues** were present in the deep learning models, suggesting a need for **better-structured datasets**.
5. **Ethical Considerations and Future Work** –
    - **Bias in judicial decisions** may be reflected in model predictions, raising concerns about fairness.
    - **Transparency and explainability** are crucial for legal AI applications.
    - Future research should focus on **restructuring the dataset** (e.g., defining cases as "convicted" vs. "not convicted") and **including sentiment analysis from legal arguments**.

Key Takeaways:

- **NLP models currently lack sufficient accuracy** for predicting Supreme Court decisions based on factual descriptions.
- **Dataset structure and legal text complexity** present major challenges in using AI for legal decision-making.
- **Future improvements should focus on better data labeling and hybrid AI models** to improve predictive accuracy.

Through this research, Jørgen has demonstrated expertise in **NLP, machine learning, and legal analytics**, contributing insights into **AI's potential and limitations in judicial decision-making**.

Abstract

The topic of this project is to detect Supreme Court case outcomes based on factual descriptions of the case. The problem addressed is whether or not natural language processing techniques can aid in decision making in a legal context. The concepts covered in this project include balancing techniques, vectorization of strings, and NLP models, looking at Naive Bayes, Support Vector Machines, Long-Short Term Memory and BERT. The paper used a dataset containing all Supreme Court decisions from 1782 until 2021 containing short

descriptions of the cases as well as the outcome. In evaluating the models, the emphasis lay on accuracy while also taking recall and precision into consideration. The results were quite poor with the simple baseline model in Naive Bayes being the best one having an accuracy of 0,65. The study finally concludes that using NLP in this context is immature, and needs further refinement. In this regard, our suggestion is to start with building a dataset that has a better structure, where a target variable could be convicted/unconvicted instead of first or second party winner.

1 Introduction and Motivation

The presence of computers in courtrooms has been regular since as early as 1952, where it was used in a California federal bribery case (Harris, 1967). Although the technology was simply assisting in a statistical analysis in this case, there were already widespread theories of the inevitable relationship between computers and law, in both administration and practice. The application of Artificial intelligence (AI) in law has since the mid-20th century been prominent, and ideas from the computer science field have been applied to law in parallel with technological advancements (Surden, 2019). Since around 2000, the focus has increasingly shifted towards machine-learning approaches, mirroring trends in the broader field of artificial intelligence. This shift has facilitated the rise of legal-tech startups that employ machine learning to enhance the efficiency and effectiveness of legal services.

The process leading up to a US Supreme Court hearing is a complex and resource-intensive affair. Typically, a case starts at a federal district court or state trial court, and can from there be further appealed to a higher court (Judicial Learning Center, 2012). The final appeal is to the US Supreme Court, who only accepts between 100-150 of the more than 7000 cases they receive annually. The cases that get chosen are also supposed to be of importance to as many people as possible, such as cases that are answering important legal disputes or are directly influencing the constitution.

This paper employs a variety of models within the field of Natural Language Processing (NLP) and machine learning to predict the outcome of a case based on the case facts. Successful implementation of a predictive model could result in significant improvements in the strategic decision-making process of law firms, for instance in deciding whether or not to appeal a case. The model's predictions could save valuable time and resources for both firms

and private clients, in addition to reducing the amount of cases presented to the US Supreme Court, leading us to the following research question:

"Can modern NLP techniques assist in predicting U.S. Supreme Court outcomes based on short, objective descriptions of the case?"

2 Related Works

Several publications have recently focused on related topics within the field of using NLP to unveil patterns predicting judicial decisions. In a notable study, Alteras et al. (2016) compared Support Vector Machines (SVMs) and Convolutional Neural Networks (CNNs) in predicting the outcomes of European Convention on Human Rights (ECHR) cases, with the CNN achieving a commendable accuracy of 82 %. It should be noted that this study had highly detailed case documents which they were able to connect to the specific article under which the case was filed.

More related to our dataset and case, a study from the Philippines used NLP to predict decisions in the Philippine Supreme Court (Virtucio et al., 2018). Based on the attributes of their dataset, they categorised all data points as either violations or non-violations. This classification was grounded in the assumption that these categories corresponded directly to the Supreme Court decisions of "affirmed" or "reversed". To deal with data imbalance, they utilised an undersampling approach before training the model. The study explored various feature extraction methods such as various length n-grams and topic modelling. They found that unstemmed bigram inputs provided the best accuracy with a random forest classifier achieving a score of 0.59, barely surpassing the effectiveness of random guessing.

A group of students at the Southern Methodist University have performed a similar classification task as our own, utilising the same dataset. Lockard et al. (2023) apply a LSTM classifier on court decisions from 2000-2019. The model reaches an accuracy of 81,1%, but unfortunately, their dataset was highly imbalanced, explaining the good accuracy score. Looking at the F1-score of 0.324 and an AUC of 0.68, the results indicate that the model requires further refinement and validation before they can be reliably applied in a judicial decision-making process.

A review of these studies suggests that predicting the outcome of US Supreme Court cases may prove challenging. In this paper we aim to build on these papers using similar models

and techniques while addressing identified shortcomings, particularly when it comes to class imbalance.

3 Methodology

3.1 Data description:

The dataset for this project comprises 3,303 Supreme Court rulings from the United States, spanning from 1978 to 2021. The data is sourced from the Oyez database, which serves as an archive of Supreme Court resources put together by Cornell's Legal Information Institute, Justia and Chicago-Kent College of Law (Oyez, n.d). The csv file contains in total 16 columns describing the case (Appendix A). The most interesting data forming the centre of our analysis is the textual description of the case in the "fact" column and the outcome of the trial in the "first_party_winner" column. Generally speaking, the first party of a case in the Supreme Court is the part that has appealed the decision of a lower court, meaning that the originally criminally accused can be either first or second party, depending on the context. What we find especially interesting about this particular dataset, is that the target value is derived from the majority vote of judges, indirectly reflecting actual human behaviour.

We performed an exploratory data analysis to identify potential issues which could have an impact on our predictions. This process included checking for missing values, data imbalance and other anomalies. There were just 15 missing values in the target variable and none in the fact column. The column with the highest number of missing entries was "issue_area," which had 142 unfilled values (Appendix A). The fact column was rather short with a mean of 1112 characters and a standard deviation of 531, seeing instances as short as 26, and as long as 6201. (Appendix B). In other words, a quite brief overview of the nature of the case, but also containing a rather large variation in length. Too short descriptions may not contribute much to our model. Showcasing all facts shorter than 100 characters reveals some hidden "null-values" with values such as "currently unavailable", "Currently unknown" and "Not available" (Appendix C).

Potential duplicates have also been assessed. All ID-numbers are unique, but the docket number shows "duplicate" cases. Upon further examination, it's evident that these numbers recur because some linked cases are listed under the same docket. Cases that are revisited or reopened are also catalogued under the original case number. Additionally, it appears that the

"fact" column includes not only string values but also some HTML tags, which will need to be addressed during the preprocessing stage. (Appendix D).

Finally, to get an overview of the most prominent words in each class, word clouds are created. These show signs of being highly similar with words like “court”, “appeal”, “claim” and “argued” being distinct in both.

3.2 Data preprocessing:

With the insight from the data exploration, some necessary steps will be executed making the data ready for training. Null values will be handled by simply removing these rows. Facts without an associated target variable are not very useful, and with only 15 such instances in a dataset of 3,303 rows, removing them is not a significant loss. Rows with strings in the “fact” column representing null values are also few and therefore also removed.

We defined a custom function to perform some cleaning of the text in the "fact" column. As mentioned in the exploration section, the data did not only contain strings, but also some html tags, creating noise in the data. Therefore the first step of the cleaning function was a regular expression removing these tags. The text was further processed by converting all characters to lowercase, which ensures that all words are interpreted equally, regardless of the casing of the words. Punctuation and numbers are also considered to not provide meaningful information and are therefore removed.

Tokenization follows, splitting the text into individual words using the “word_tokenize” function from NLTK. This is a fundamental step in NLP and can be explained as a processing task of breaking up text into smaller components, usually called tokens (Friedman, 2023). The tokens are then refined removing stopwords, which are common words that typically do not add significant meaning to a text corpus. Examples are “the”, “and", “or” and so on. The idea is that noise is removed, making the algorithms able to focus on the meaningful words that provide actual context.

3.3 Text Vectorization

In this section, we explore key text vectorization techniques in Natural Language Processing (NLP): Bag-of-Words (BoW), Term Frequency-Inverse Document Frequency (TF-IDF), and Word2Vec. We will assess how each method processes text data and their implications for improving our project's NLP tasks.

3.3.1 Bag-of-Words

Bag-of-Words (BoW) is a straightforward vectorization technique used in NLP to represent text. In the BoW model, each document is converted into a vector, where each element indicates the presence or frequency of words from the corpus (Jurafsky & Martin, 2024: 62). This method disregards the order and context of words, focusing solely on their frequencies within the document. Although this method fails to capture contextual representations of words, it is based on the reasonable assumption that words that occur more frequently are the ones that present the text's main ideas (Murel & Kavlakoglu, 2024). Given the shortcomings as well as the potential effectiveness of this approach, we will use it for our baseline model. However, for our other models, we will explore more advanced methods..

3.3.2 Term Frequency - Inverse Document Frequency

Term Frequency - Inverse Document Frequency (TF-IDF) is similar to the BoW approach, but takes measures to reduce the noise of words that occur often, but do not contribute significant meaning to the text. The term frequency is the same calculation as BoW, but the inverse document frequency reduces the weight of words which occur in many documents (Jurafsky & Martin, 2024: 115-121). We suspect that short descriptions of cases brought to the Supreme Court could contain legal phrases that are necessary to describe any court case, which is indeed supported by a visualisation of the most frequent words for each of the classes (Appendix I). By reducing the weights of these phrases, we aim to enhance the accuracy of our other models' predictions.

3.3.3 Word2Vec

Word2Vec creates dense vector embeddings of words, improving upon earlier methods such as Bag-of-Words and TF-IDF (Jurafsky & Martin, 2024: 121). It is trained using the skip-gram model, which works by predicting the surrounding words for a given word to understand the context. In our approach, each word in the text is turned into a vector by passing it through the Word2Vec model. This process positions similar words closely together in a multi-dimensional space, capturing their context and relationship to one another. Unlike the sparse vectors produced by Bag-of-Words and TF-IDF, Word2Vec embeddings are dense and context-aware. This makes Word2Vec an excellent match for our LSTM model, as the rich, contextual embeddings provide improved inputs that can assist in interpreting complex textual patterns.

3.4 Class imbalance

For binary classification tasks, an imbalance between positive and negative classes can affect the model's ability to learn patterns about the minority class leading to a bias towards the majority class. Techniques for dealing with such class imbalances include undersampling and oversampling. Undersampling is the process of removing instances of the majority class to restore balance, while oversampling includes creating new synthetic instances of the minority class by copying them or through more sophisticated approaches.

As put forward in the data exploration section of this paper, our dataset is imbalanced to such a degree that it could have a negative effect on our models' predictions. To handle the imbalance in our dataset, we have opted for the Synthetic Minority Oversampling Technique (SMOTE). SMOTE generates new observations of the minority class by randomly selecting samples from this class, using K-Nearest Neighbours to identify similar samples, randomly selecting one of these neighbours, and interpolating between the two to create a new synthetic sample (Blanca Li & Peter Lu, 2021). This method negates the need to remove potentially important information in the majority class while balancing the dataset with plausible and informative synthetic samples.

3.5 Models

3.5.1 Baseline: Naive Bayes

The baseline for our binary classification problem using supervised machine learning will be a Naive Bayes model. The Naive Bayes model assumes that the position of each word is irrelevant and that words are independent, allowing us to multiply conditional probabilities of each word given a class, significantly reducing the computational intensity (Jurafsky & Martin. H., 2024: 62). The algorithmic complexity of this model makes it ideal to handle the high dimensional data which is generally associated with text processing, and it makes the model interpretable. We can easily extract the features most strongly correlated with each of the classes to get a deeper understanding of the complexity of the problem we are trying to solve. These attributes make the Naive Bayes classifier an ideal baseline for our project where we aim to progress through increasingly complex models to evaluate the tradeoff between complexity and performance.

We employ a grid search to tune the model, applying ranges of values for regularisation, various length n-grams and various methods of reducing words to their base form.

Regularisation helps to prevent overfitting by adding a penalty to the model's complexity, and in our grid search, we adjust the strength of this penalty by varying the value of alpha. N-grams have the potential of capturing context and patterns over short spans of text, and our grid search explores different lengths of n-grams to find the most informative sequences for our model. We also apply lemmatization, stemming and no word base transformations to our grid search as there are potential benefits to all these approaches.

3.5.2 Support Vector Machine

A Support Vector Machine (SVM) is a machine learning algorithm which aims to fit the hyperplane that best separates classes in the multidimensional space. (IBM, 2023a). SVM works by identifying the hyperplane that best separates the different classes in the feature space. Similar to naive bayes, SVM is commonly used in text classification tasks, although SVM tends to perform better for non-linear data. Another benefit of SVM is that it is particularly well suited for small and medium complex datasets which is promising given our dataset (Géron, 2019, p. 153). It provides a beneficial balance between interpretability and performance, and its ability to use different kernels makes it flexible in capturing various data patterns and enhancing the prediction accuracy.

Similar to the approach with Naive Bayes, we use a grid search across ranges of n-grams to capture context, regularisation to limit overfitting and various word base transformations. Another important hyperparameter to handle the complexity of our data is the choice of kernel, so this was also included in our grid search. Different kernels can capture different types of relationships in the data, making it essential to grid search various kernels to identify the one that best models the underlying structure of our dataset.

3.5.3 Recurrent Neural Networks: LSTM

A more advanced approach to this text classification problem is employed to potentially capture the sequential nature of the textual facts about the case. One of the major challenges of NLP has been to accurately address unbounded dependencies. Long Short-Term Memory (LSTM) handles these unbounded dependencies through a gated cell structure regulating the flow of information. The model allows a node to take sequential input values which are retained or forgotten over arbitrary time intervals. Utilising a gradient descent algorithm, the model improves parameters affecting what is remembered and forgotten, making the model capable of capturing long-range dependencies within the text.

Due to complex interactions between components within each of the LSTM gates as well as the process of Backpropagation Through Time (BPTT), the model is more computationally intensive than the previous two models. BPTT lets the model adjust its weights based on errors made in each layer of the model for every input. This complex process also makes the model's decision-making process less transparent.

Our LSTM model architecture included an embedding layer initialised with Word2Vec embeddings from the 'word2vec-google-news-300' model to leverage pre-trained semantic knowledge. This was followed by two LSTM layers with 100 and 64 units aiming at capturing long-term dependencies while keeping training time down. Dropout layers with a rate of 0.2 were added after each LSTM layer to prevent overfitting. The final dense layer performs a linear transformation of its inputs and applies a sigmoid function, enabling binary classification by mapping the output to a probability between 0 and 1. The model was trained with the Adam optimizer, chosen for its efficient handling of sparse gradients, using binary cross-entropy as the loss function to measure performance. We set the batch size to 32 and trained the model for 10 epochs, balancing between sufficient learning and computational efficiency.

3.5.4 Pre-trained Language Models: BERT

Bidirectional Encoder Representations from Transformers (BERT) are based on the transformer architecture, which represents a major advance in NLP by introducing a self-attention mechanism. For each layer of the neural network, the model aims to produce richer contextualised representations of its input words over large spans of text to include important contextual dependencies which can be sentences apart (Jurafsky & Martin, 2024: 215). BERT uses a bidirectional self-attention mechanism, allowing it to draw from both previous and future words to produce its contextual representations (Jurafsky & Martin, 2024: 242 - 255). BERT models are pretrained on large datasets, performing unsupervised learning tasks to learn general language representations. After pretraining, the model can be fine tuned to a specific dataset, using supervised learning to slightly adjust its weights to optimise performance for a specific task. This transfer learning approach allows the model to perform well even for smaller datasets.

We utilised the 'bert-base-uncased' variant, which is pre-trained on a large corpus and hence well-equipped to capture the nuances in legal texts. The dataset was initially tokenized using

the BERT tokenizer, ensuring that each fact was represented appropriately for the BERT model. The model architecture included a BERT layer followed by a dropout layer and a fully connected layer, with dropout set to 0.2 to mitigate overfitting. A maximum sequence length of 128 was chosen based on the distribution of sequence lengths. This ensured coverage of most text instances without excessive padding. Parameters for batch size, learning rate and warm-up phase have been chosen with a goal of developing a well balanced model that performs well without being too computationally intensive.

4 Results

In the results section, we begin with a presentation of the model scores for the models we tested in our project. The evaluation metrics shown in Table 1 are the weighted average of the precision, recall, and F1-score, in addition to the accuracy of the model.

4.1 Model scores

Naive Bayes:				
Text vectorisation technique	**Precision**	**Recall**	**F1-Score**	**Accuracy**
Bag of Words	0.61	0.65	0.59	0.65
SVM:				
	Precision	**Recall**	**F1-Score**	**Accuracy**
TF-IDF	0.77	0.65	0.51	0.65
LSTM:				
	Precision	**Recall**	**F1-Score**	**Accuracy**
Word2Vec	0.63	0.63	0.63	0.63
BERT:				
		Recall	**F1-Score**	**Accuracy**
Bert Tokenizer	0.58	0.62	0.57	0.62

Table 1: Evaluation metrics of the models used in this paper

Our baseline model, Naive Bayes, using BoW and SMOTE, showed an accuracy of 0.65, which is relatively promising considering the accuracy shown by the other, more advanced models (Appendix G.1).

The model with the joint highest accuracy score of 0.65 is the SVM. It did, however, score significantly lower on the f1-score, indicating worse performance than the baseline model. Upon further examination, we can see from the classification report that the SVM model is highly inefficient in predicting class 0, which displays a recall value of 0.00 (Appendix G.2). This, combined with a recall for class 1 of 1.00, indicates that the model predicts all instances as class 1.

Our third-best performing model was the LSTM with Word2Vec and SMOTE , achieving an accuracy of 0.63. It also showed values of 0.63 for precision, recall, and f1-score. Even though it showed better values than the Naive Bayes for both precision and f1-score, the model achieved a lower accuracy (Appendix G.3).

Our fourth and final model, BERT using SMOTE, achieved an accuracy of 0.62, which is also lower than our baseline model (Appendix G.4). Despite being the most advanced model used in our project, BERT showed itself inferior to our baseline model when looking at the weighted average for both precision and recall.

4.2 Overfitting

When analysing the standard deviation of cross-validation scores, which is 0.002 for SVM and 0.019 for the Naive Bayes model, we observe a high level of consistency in model performance across different folds (Appendix G). This consistency is generally favourable as it indicates model stability (Altman & Bland, 2005). However, the learning curves expose signs of overfitting for both models (Géron, 2019: 130-134; Appendix H). They perform exceptionally well on the training data, but fail to achieve similar effectiveness on the validation data, which is more indicative of their performance on unseen data.

The LSTM model also shows clear signs of overfitting. This is evident from its training data performance, with accuracy on the training data reaching approximately 0.98 and a loss decreasing to about 0.05, compared to the validation results where accuracy remains around 0.57 and loss increases to approximately 1.91 (Appendix G.3). Additionally, the gap visualised in the learning curve indicates that the model is overfitting to the training data and not generalising well to unseen data (Géron, 2019: 130-134; Appendix H.3). Lastly, the

BERT model demonstrates good generalisation from looking at the consistent validation and test accuracies around 0.62. There is no significant evidence of overfitting since the test performance closely mirrors the validation results (Appendix G.4).

5 Discussion

5.1 Project findings

The chosen algorithms and hyperparameters with the "justice" dataset performs quite poorly in classifying the outcome of the case. With the best model having an accuracy of 0.65, in a binary classification, it must be expressed that this is not satisfactory. The issue of overfitting which we have elaborated on also suggests that the models struggle to find general trends in the data and seem to converge against finding unique patterns specific for each case. This could for instance be the model "remembering" a specific name or place mentioned in a case description with the actual outcome. Seeing the classification reports for the different models also reveals that many of them are simply labelling all instances as true, as this results in an accuracy of 0,65 because of the class imbalance. In other words, the models are not able to learn any important predictors which can be generalised to unseen data. It is therefore important to look into what may have caused such bad performances.

5.2 Limitations

There is a limited amount of data points, despite the number of years with recorded observations (1955-2021). With only 50 observations on average per year, the foundation that the model is built upon might be too weak. Since we are dealing with supreme court decisions, most cases come as a consequence of an appeal. The party that appealed the decision of a lower court will also become the first party of the case, meaning that both the original prosecutor and defendant can stand as first party. This also implies that a "first_party_winner" being true could mean both a conviction and an acquittal.

The content and nature of the fact column is a source of confusion for the model. In many cases, this column will describe the case from the starting point, which in many cases describes other related issues. An example is the case between Douglas Brownback, et al. and James King. The original case covered an incident where King was wrongfully approached by undercover cops. As they were undercover and talking to the wrong guy, King became suspicious and gave resistance thinking they were trying to rob him. King won the first case

where he was pursued with charges by the prosecutor, and ended up suing the United States and the FBI-agents. It was this lawsuit that was dealt with in the supreme court, even though a large part of the fact column dealt with the original charges against King. In other words, even though the fact column provides the right context of the case at hand, important details might come off as nuances drowning in case descriptions.

Looking back at the word cloud and the most frequent words from both target classes, it becomes even clearer why the models seem to struggle separating the two instances. Many of the most frequent words are overlapping the two instances, possibly making it hard to find proper explanatory distinctions between them (Appendix I). In other words, it appears that the "facts" columns indeed are as unbiased and objective as they should ideally be for a trial.

The nature of cases brought up in the Supreme Court can also negatively affect the models ability to classify them. As explained in the introduction, these cases usually involve conflicts addressing the constitution as well as cases affecting a lot of people, where a Supreme Court practice has yet to be established. In other words, cases that are intricate and subject to varying interpretations and often prove challenging to resolve and reach consensus on.

5.3 Ethical considerations

Even though our model shows little signs of being prone to be implemented in a practical setting, further work and possible implementation calls for a cautious and respectful approach. Responsible AI (RAI) provides a framework to design, develop and use AI systems responsibly (Stryker, 2024). The framework offers a total of five pillars, where two of them will be discussed specifically due to their natural connection to our given context.

Fairness. Even though algorithms are not biased or discriminatory in themselves, their training data might be. It is reasonable to assume that the judicial powers in the United States may have had some biases in their court decisions throughout the time period of the data sample. These biases may be inherited in the model, disfavoring certain cases, ethnic groups, social classes and similar, compared to an objective decision should offer. If these biases in the next instance, makes a law firm using a similar model advice differently according to a clients ethnicity and societal background, we have a problem.

Transparency. For these solutions to work as a reliable tool, users must have visibility into how the model makes predictions, and what it is based on. This means being able to independently evaluate its functionality as well as having an understanding of its strengths

and weaknesses. An example could be the understanding of this tool being a product of cases where most of them are rather “normal”. Edge cases as well as newer legal problems might not be very applicable as a consequence.

In addition to ethical concerns in the model predictions, there could be other implications related to the use. The Constitution grants everyone the right to an attorney representing them in a trial. Tools like this could lead to attorneys on a macro level discharging weaker cases at a greater rate than without, making a weak case even weaker for the defendant, eventually hurting the legal system.

6 Conclusion

In conclusion, our project has evaluated Naive Bayes, SVM, LSTM and BERT for classifying the outcome of US Supreme Court cases. All in all, the models performed quite poorly, with the highest accuracy (0.65) in Naive Bayes and SVM. However looking into the results, the scores of the SVM stem from simply labelling all instances as true. We can therefore conclude that modern NLP techniques can not assist in predicting US Supreme Court Decisions based on our data. In the discussion, we point out several factors, possibly explaining the poor results. First of all, the originally “accused” side of the case will appear as both first and second party. This makes it difficult to label the cases as “convicted” and “unconvicted”. Secondly, the fact column is rather noisy, explaining initial phases of the case as well as former decisions, making it hard to sort out essential details for the specific decision that will be made in the Supreme Court.

For future work, we would like to look into the possibility of building a more structured dataset where we could have consistency in first and second party. This could for instance be initial criminal cases with prosecutors as first party and defendant as second. It would also be interesting to include the closing arguments of both sides, looking into if they could reveal any sentiment on expectations regarding the outcome.

References

Aletras, N., Tsarapatsanis, D., Preoţiuc-Pietro, D., & Lampos, V. (2016). Predicting judicial decisions of the European Court of Human Rights: A Natural Language Processing perspective. *PeerJ Computer Science*, *2*, e93. https://doi.org/10.7717/peerj-cs.93

Altman, D. G., & Bland, J. M. (2005). Standard deviations and standard errors. *BMJ*, *331*(7521), 903. https://doi.org/10.1136/bmj.331.7521.903

Friedman, R. (2023). Tokenization in the Theory of Knowledge. *Encyclopedia*, *3*(1), Article 1. https://doi.org/10.3390/encyclopedia3010024

Garoupa, N., & Gomez-Pomar, F. (2008). Cashing by the Hour: Why Large Law Firms Prefer Hourly Fees over Contingent Fees. *The Journal of Law, Economics, and Organization*, *24*(2), 458–475. https://doi.org/10.1093/jleo/ewm063

Géron, A. (2019). *Hands-on Machine Learning with Scikit_Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems (Second edition)*. O'Reilly Media, Inc.

Harris, A. (1967). Judicial Decision Making and Computers. *Villanova Law Review*, *12*(2), 272.

Judicial Learning Center - How Does the U.S. Supreme Court Work? | The Judicial Learning Center. (2012, July 16). https://judiciallearningcenter.org/the-us-supreme-court/

Jurafsky, D., & Martin, J. H. (2024). *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition* (3rd ed.). [Draft]. Retrieved from https://web.stanford.edu/~jurafsky/slp3/ed3book.pdf

Li, B., Lu, P. (2021, November 4). *SMOTE - Azure Machine Learning*. https://learn.microsoft.com/en-us/azure/machine-learning/component-reference/smote?view=azureml-api-2

Lockard, K., Slater, R., & Sucrese, B. (2023). Using NLP to Model U.S. Supreme Court Cases. *SMU Data Science Review*, *7*(1). https://scholar.smu.edu/datasciencereview/vol7/iss1/4

Maheshri, V., & Winston, C. (2014). An exploratory study of the pricing of legal services. *International Review of Law and Economics*, *38*, 169–173. https://doi.org/10.1016/j.irle.2013.05.003

Murel, J., & Kavlakoglu, E. *What is bag of words? | IBM*. (2024, January 19). https://www.ibm.com/topics/bag-of-words

Oyez. (n.d) *About.* Oyez. https://www.oyez.org/about

Stryker, C. (2024) What is responsible AI? IBM. https://www.ibm.com/topics/responsible-ai

Surden, H. (2019). Artificial Intelligence and Law: An Overview. *Georgia State University Law Review*, *35*(4). https://readingroom.law.gsu.edu/gsulr/vol35/iss4/8

Virtucio, M. B. L., Aborot, J. A., Abonita, J. K. C., Aviñante, R. S., Copino, R. J. B., Neverida, M. P., Osiana, V. O., Peramo, E. C., Syjuco, J. G., & Tan, G. B. A. (2018). Predicting Decisions of the Philippine Supreme Court Using Natural Language Processing and Machine Learning. *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, *02*, 130–135. https://doi.org/10.1109/COMPSAC.2018.10348

What Is Support Vector Machine? | IBM. (2023, December 12). https://www.ibm.com/topics/support-vector-machine

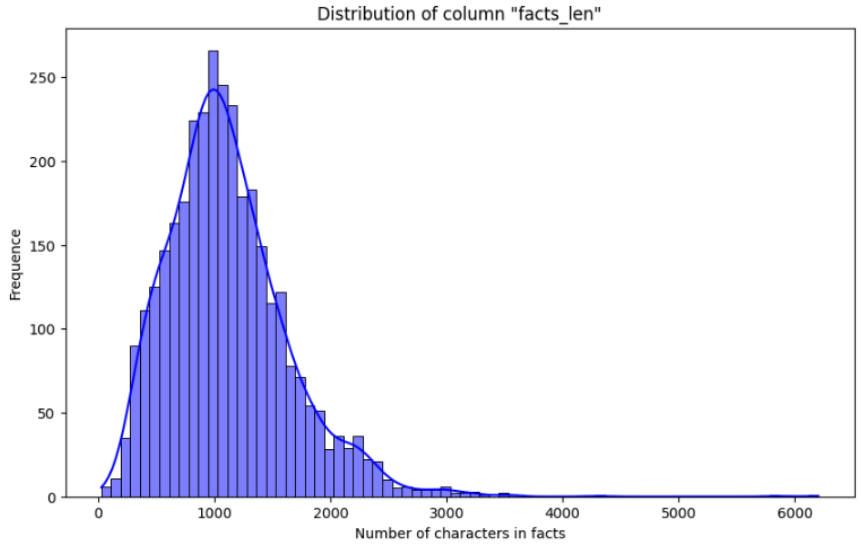
Appendix:

Appendix A: Columns and null values

```
Unnamed: 0              0
ID                      0
name                    0
href                    0
docket                 11
term                    0
first_party             1
second_party            1
facts                   0
facts_len               0
majority_vote           0
minority_vote           0
first_party_winner     15
decision_type           7
disposition            72
issue_area            142
dtype: int64
```

Appendix B: Fact_len. Mean, STDV and distribution

```
Mean: 1113.02
Standard Deviation: 530.34
```



Appendix C: Displaying short fact descriptions

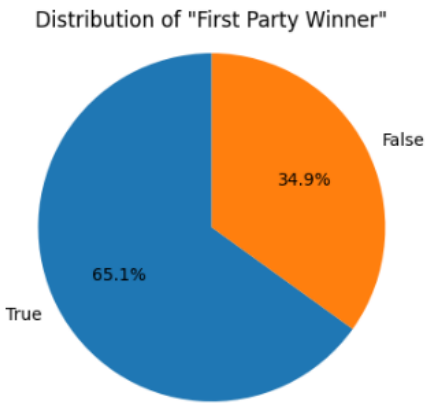
Datapoints with "fact_len" < 100:

	facts	facts_len
543	Currently unavailable. \n	30
1015	Currently unknown. \n	26
2470	Not Available. \n	31

Appendix D: Showcasing fact columns:

	facts
1256	In 1997, Maria Ohler was arrested and charged with importation of marijuana and possession of marijuana with the intent to distribute, after a customs inspector noticed that someone had tampered with one of Ohler's van interior panels and discovered approximately 81 pounds of marijuana. Before the trial, the government filed in limine motions to admit Ohler's prior felony conviction as character evidence under Federal Rule of Evidence 404(b) and as impeachment evidence under Rule 609(a)(1). Also before the trial, the District Court denied the motion to admit the conviction as character evidence. After the beginning of the trail, the court ruled that if Ohler testified, evidence of her prior conviction would be admissible under Rule 609(a)(1). While testifying, Ohler admitted on direct examination that she had been convicted of possession of methamphetamine in 1993. Subsequently, Ohler was found guilty. On appeal, Ohler challenged the District Court's in limine ruling, allowing the government to use her prior conviction for impeachment purposes. In affirming, the Court of Appeals held that Ohler waived her objection by introducing evidence of the conviction during her direct examination.
2162	A Michigan trial court convicted Richard Perry Bryant of second degree murder, being a felon in possession of a firearm, and possession of a firearm during commission of a felony. On appeal, Mr. Bryant challenged the admission of the victim's statements at trial for violating his Sixth Amendment right of confrontation. The victim stated that Mr. Bryant shot him, but died shortly thereafter. The Michigan Court of Appeals affirmed the trial court. The Michigan Supreme Court reversed, holding that the statements that the victim made to police before his death were testimonial and their admission violated Mr. Bryant's right to confrontation. The court reasoned that the victim's statements were made in the course of a police interrogation whose primary purpose was to establish or prove events that had already occurred, not to enable police to meet an ongoing emergency. Therefore, the lower court held that the statements were "testimonial" for the purposes of the enhanced confrontation protections set forth by the U.S. Supreme Court in Crawford v. Washington and should not have been admitted against Mr. Bryant at trial because he did not have the opportunity to cross-examine the victim prior to his death.

Appendix E: Distribution of “First Party Winner”



Appendix F: Stemming vs Lemmatization

```
0       [jane, roe, fiction, name, use, court, documen...
1       [joan, stanley, three, children, peter, stanle...
2       [john, giglio, convict, pass, forg, money, ord...
3       [idaho, probat, code, specifi, male, must, pre...
4       [miller, conduct, mass, mail, campaign, advert...
                              ...
3298    [refugio, palomarsantiago, mexican, nation, gr...
3299    [tarahrick, terri, plead, guilti, one, count, ...
3300    [joshua, jame, cooley, park, pickup, truck, si...
3301    [ongo, case, origin, jurisdict, fact, explain,...
3302    [natur, ga, act, nga, usc, permit, privat, com...
Name: facts, Length: 3303, dtype: object
```

```
0       [jane, roe, fictional, name, used, court, docu...
1       [joan, stanley, three, child, peter, stanley, ...
2       [john, giglio, convicted, passing, forged, mon...
3       [idaho, probate, code, specified, male, must, ...
4       [miller, conducting, mass, mailing, campaign, ...
                              ...
3298    [refugio, palomarsantiago, mexican, national, ...
3299    [tarahrick, terry, pleaded, guilty, one, count...
3300    [joshua, james, cooley, parked, pickup, truck,...
3301    [ongoing, case, original, jurisdiction, fact, ...
3302    [natural, gas, act, nga, usc, permit, private,...
Name: facts, Length: 3303, dtype: object
```

Appendix G: Model results:

G1: Naive Bayes/ BoW/ SMOTE

```
Cross Val Standard Deviation:  0.01958820817093728
Best accuracy from Grid Search:  0.6410646387832699
Best parameters from Grid Search:  {'classifier__alpha': 1, 'preprocess__strategy': 'none', 'vectorizer__ngram_range': (1, 2)}
              precision    recall  f1-score   support

           0       0.53      0.12      0.19       233
           1       0.66      0.94      0.78       425

    accuracy                           0.65       658
   macro avg       0.60      0.53      0.48       658
weighted avg       0.61      0.65      0.57       658
```

G2: Support Vector Machine/ TF-IDF/ SMOTE

```
Cross Val Standard Deviation:  0.0025221481295478 24
Best accuracy from Grid Search:  0.7898520015321324
Best parameters from Grid Search:  {'classifier__C': 1, 'classifier__kernel': 'rbf', 'preprocess__strategy': 'none', 'vectorizer__ngram_range': (1, 2)}
              precision    recall  f1-score   support

           0       1.00      0.00      0.01       233
           1       0.65      1.00      0.79       425

    accuracy                           0.65       658
   macro avg       0.82      0.50      0.40       658
weighted avg       0.77      0.65      0.51       658
```

G3: LSTM / SMOTE

```
=================================================================
Total params: 4526605 (17.27 MB)
Trainable params: 202705 (791.82 KB)
Non-trainable params: 4323900 (16.49 MB)
_________________________________________________________________
Epoch 1/10
97/97 [==============================] - 14s 36ms/step - loss: 0.6645 - accuracy: 0.6170 - val_loss: 0.6838 - val_accuracy: 0.6152
Epoch 2/10
97/97 [==============================] - 2s 22ms/step - loss: 0.6044 - accuracy: 0.6956 - val_loss: 0.6732 - val_accuracy: 0.6152
Epoch 3/10
97/97 [==============================] - 2s 20ms/step - loss: 0.5457 - accuracy: 0.7296 - val_loss: 0.7204 - val_accuracy: 0.5860
Epoch 4/10
97/97 [==============================] - 2s 22ms/step - loss: 0.4659 - accuracy: 0.7897 - val_loss: 0.7750 - val_accuracy: 0.5918
Epoch 5/10
97/97 [==============================] - 2s 21ms/step - loss: 0.3697 - accuracy: 0.8419 - val_loss: 0.9502 - val_accuracy: 0.5802
Epoch 6/10
97/97 [==============================] - 2s 22ms/step - loss: 0.2682 - accuracy: 0.8906 - val_loss: 1.1366 - val_accuracy: 0.5685
Epoch 7/10
97/97 [==============================] - 2s 22ms/step - loss: 0.1841 - accuracy: 0.9348 - val_loss: 1.3083 - val_accuracy: 0.5860
Epoch 8/10
97/97 [==============================] - 2s 19ms/step - loss: 0.1179 - accuracy: 0.9555 - val_loss: 1.6651 - val_accuracy: 0.5977
Epoch 9/10
97/97 [==============================] - 2s 18ms/step - loss: 0.0958 - accuracy: 0.9649 - val_loss: 1.6145 - val_accuracy: 0.5539
Epoch 10/10
97/97 [==============================] - 2s 21ms/step - loss: 0.0654 - accuracy: 0.9747 - val_loss: 1.9076 - val_accuracy: 0.5743
27/27 [==============================] - 2s 10ms/step
Confusion Matrix:
[[297 135]
 [225 199]]

Classification Report:
              precision    recall  f1-score   support

           0       0.57      0.69      0.62       432
           1       0.60      0.47      0.53       424

    accuracy                           0.58       856
   macro avg       0.58      0.58      0.57       856
weighted avg       0.58      0.58      0.57       856
```

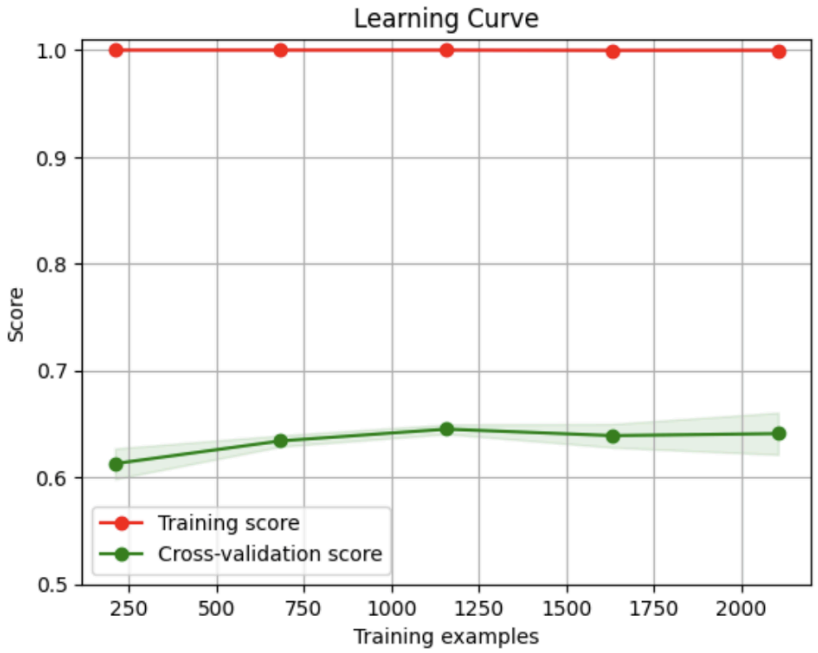
G4: BERT / SMOTE

```
Epoch 5/10
Validation Accuracy: 0.5882
              precision    recall  f1-score   support

           0       0.44      0.39      0.41       185
           1       0.66      0.71      0.68       308

    accuracy                           0.59       493
   macro avg       0.55      0.55      0.55       493
weighted avg       0.58      0.59      0.58       493

Epoch 6/10
Validation Accuracy: 0.6105
              precision    recall  f1-score   support

           0       0.46      0.21      0.28       185
           1       0.64      0.85      0.73       308

    accuracy                           0.61       493
   macro avg       0.55      0.53      0.51       493
weighted avg       0.57      0.61      0.56       493

Epoch 7/10
Validation Accuracy: 0.6227
              precision    recall  f1-score   support

           0       0.50      0.29      0.36       185
           1       0.66      0.82      0.73       308

    accuracy                           0.62       493
   macro avg       0.58      0.56      0.55       493
weighted avg       0.60      0.62      0.59       493
```

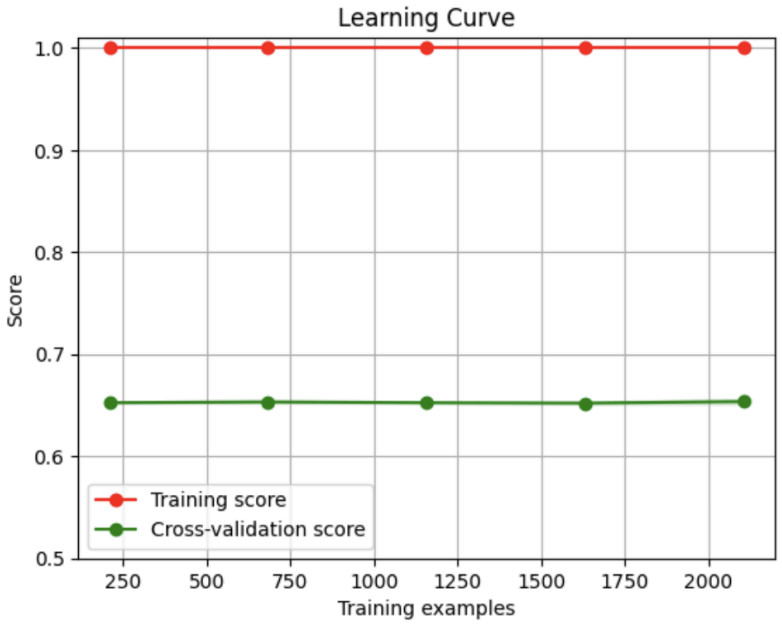
```
Epoch 8/10
Validation Accuracy: 0.6004
              precision    recall  f1-score   support

           0       0.44      0.24      0.31       185
           1       0.64      0.81      0.72       308

    accuracy                           0.60       493
   macro avg       0.54      0.53      0.52       493
weighted avg       0.57      0.60      0.57       493

Epoch 9/10
Validation Accuracy: 0.5963
              precision    recall  f1-score   support

           0       0.43      0.24      0.31       185
           1       0.64      0.81      0.71       308

    accuracy                           0.60       493
   macro avg       0.54      0.53      0.51       493
weighted avg       0.56      0.60      0.56       493

Epoch 10/10
Validation Accuracy: 0.5984
              precision    recall  f1-score   support

           0       0.43      0.23      0.30       185
           1       0.64      0.82      0.72       308

    accuracy                           0.60       493
   macro avg       0.54      0.53      0.51       493
weighted avg       0.56      0.60      0.56       493

Evaluating on the test set...
Test Accuracy: 0.6174
              precision    recall  f1-score   support

           0       0.48      0.22      0.30       185
           1       0.65      0.86      0.74       309

    accuracy                           0.62       494
   macro avg       0.56      0.54      0.52       494
weighted avg       0.58      0.62      0.57       494
```

Appendix H: Learning curves

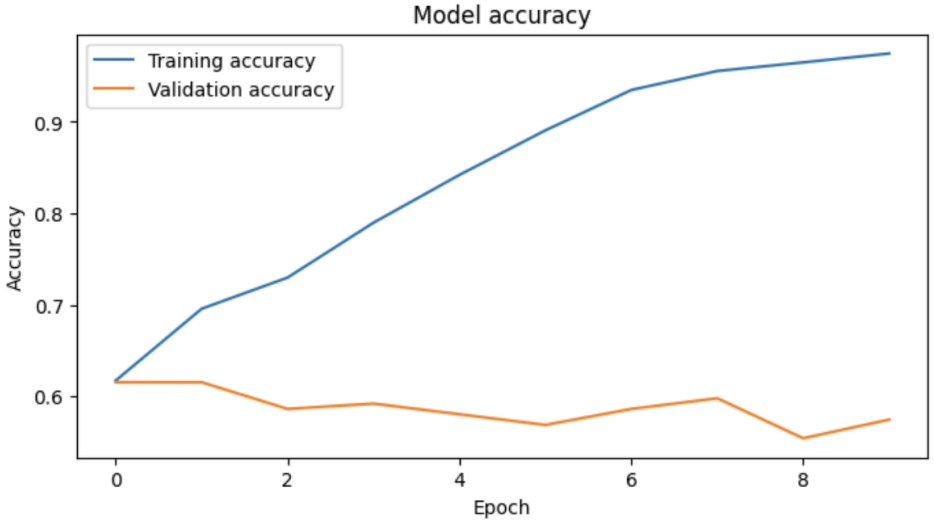
H1: Naive Bayes/ BoW/ SMOTE



H2: Support Vector Classifier/ TF-IDF/ SMOTE



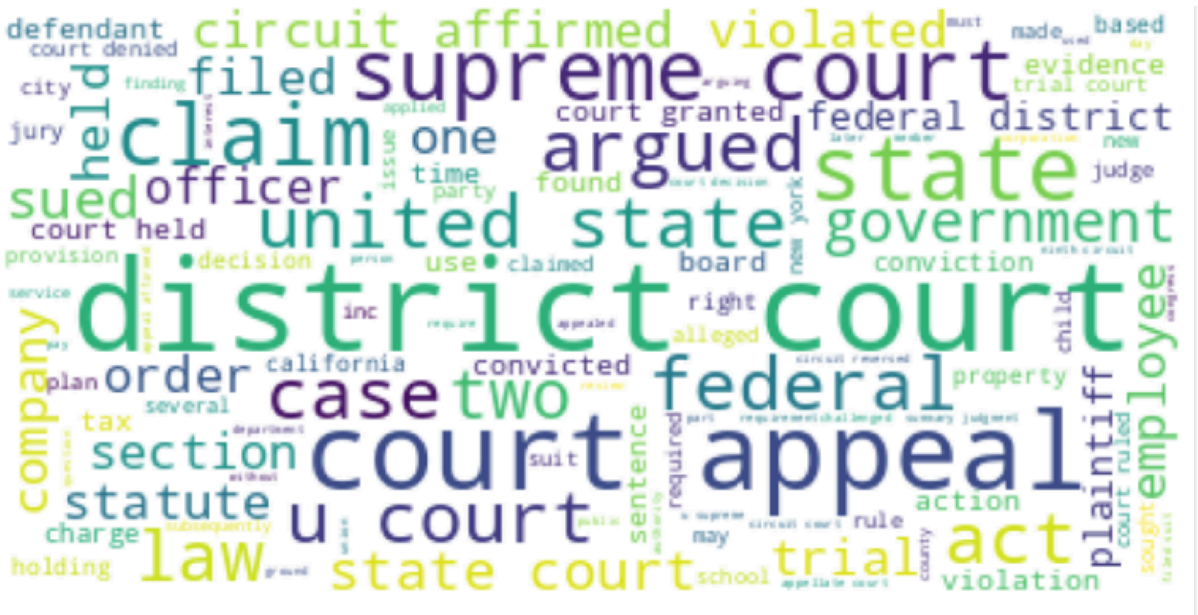
H3: LSTM / SMOTE



Appendix I: Word Clouds on First_party_winner equals true and false

I1: True

I2: False


defendant
court denied
circuit affirmed violated
made
based
supreme court
evidence
trial court
city
filed
court granted
federal district
claim
argued
held
one
state
jury
new
judge
sued
officer
time
found
party
united state
government
court held
provision
decision
use
claimed
board
conviction
district court
service
right
inc
employee
company
alleged
child
california
convicted
order
federal
case
two
plan
property
several
tax
plaintiff
section
court appeal
statute
sentence
required
suit
u court
act
action
charge
rule
subsequently
law
may
trial
holding
state court
school
violation
new york
issue
finding
applied
court ruled
sought
county