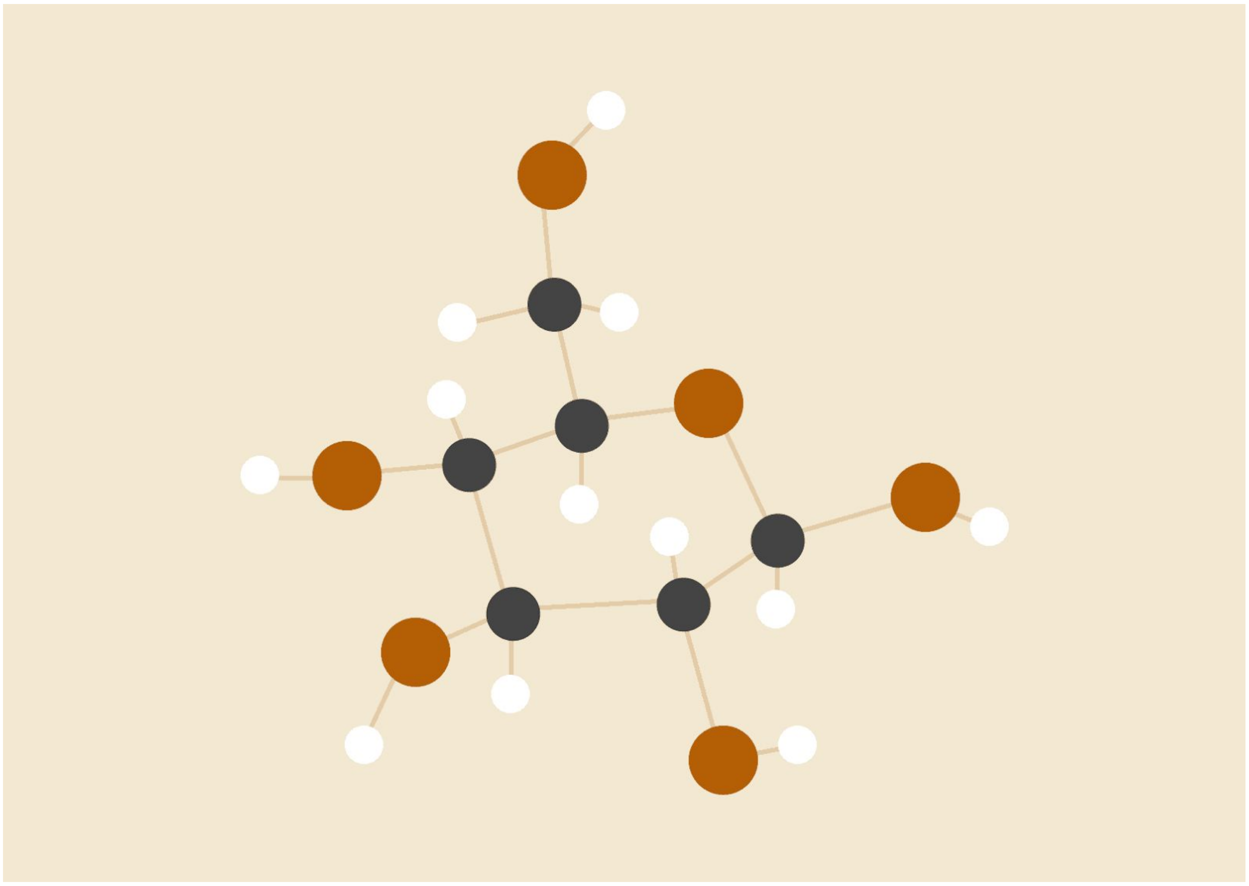
LEGAL ANALYTICS

Predict the Winning Party



Group 1

Mentor : Mr. Animesh Tiwari

Shraddha Ganesh Vispute
Arunava Nandy
Harish Chauhan
Prithvi Parihar
Pradeep Vasan D
Monish Martin

Table of Contents

1. Project Statement

The project aims to employ Legal Analytics in order to predict the winning party given a case. The primary objective of a Law Firm is to fight cases where chances of success is high. Appropriate analysis and Machine Learning models aid a firm or a lawyer to make an informed choice based on historical data and sound statistical analysis.

2. Dataset Details

The database used dates back to the year 1946 during the beginning of Vinston Court and Warren Court. There have been many versions since then. The data available can be used with various statistical packages to perform various analyses with respect to what the business/ Law firm considers important. The advanced modelling and EDA techniques used in the project will enable the business to calculate and view relationships among the variables which shall aid the management in making informed decisions. The database consists of information from 1946. Thus with increase in information, the needed responsibility to alter/ transform the data is carried out

3. Complexity Involved

1. Find the key area, gaps identified in the topic survey where the project can add value to the customers and business.

2. Since datasets involving court cases have multiple variables, the variables that aid in the prediction of "partyWinning" have to be selected. Certain predictor variables involve information that a law firm may not have access to, before the case begins in the supreme court. Hence these variables are removed

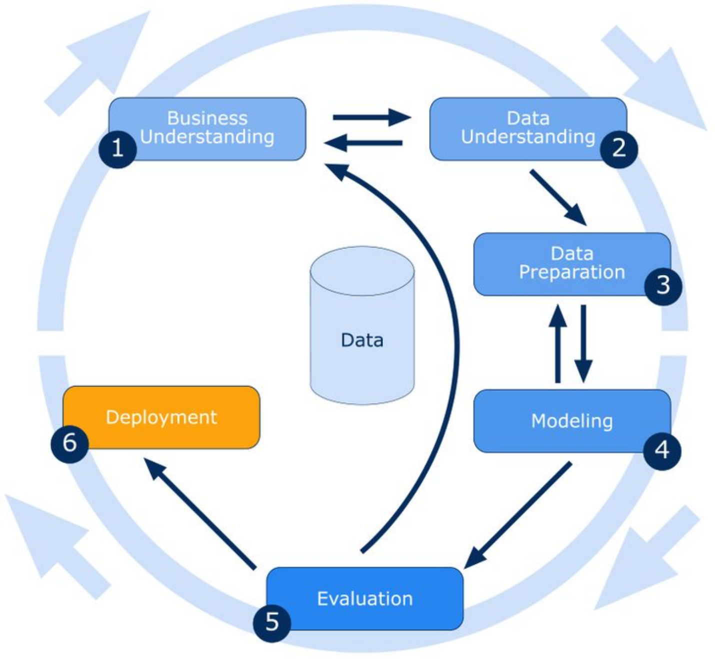
3. Merging of different columns so that it makes sense which further reduces processing and execution time and then building an accurate model with decent respective model scores using required libraries and algorithms.To obtain duration of the case , various columns were merged through appropriate datetime techniques.
4. Since each column is unique, the null value imputation was done diligently using hot deck imputation.
5. Finding out the best target variable as the dataset contained multiple potential target variables.

4. Project Outcome

- ***Commercial*** : Given the details of a case, the law firm will be able to predict the outcome which shall enable the agency to maintain a better track record and select cases where the chances of winning are high.

- ***Social*** : Should the petitioner reconsider filing a case based on model results. This will help the petitioner in saving time and money. Hence the law firm will be able to provide alternate solutions to the petitioner and gain reputation for the same.

5. METHODOLOGY TO BE FOLLOWED



[1] The Primary step of this project is to understand the relationship between data and the field of Law. The US supreme court database is used for this project. A dataset which focuses on the case details and final judgement is chosen for analysis . **[2]** As Legal Analytics is a broad field, the plausible target variables and their respective predictor variables are studied. The Target variable has been identified as the winning party and the case details as the predictor variables.**[3]** The necessary data is prepared only when the data has been studied for its important characteristics and the potential anomalies it may possess. Exploratory Data Analysis is performed on the data using Univariate-Bivariate-Multivariate analysis through visual inspection or statistical tests based on the type of data. These steps build the necessary foundation to perform feature engineering, data reduction and transformation. **[4]** The next phase is to create a model which is capable of capturing the characteristics of the target variable. This involves selecting an optimal Hypothesis function (ML model) which captures the true relationship between the explanatory and response variables (Low Bias). At the same time the model must also be able to make good predictions for data that it has not seen before (Low Variance). Basic models such as Logistic Regression, Linear Regression to advanced models such as AdaBoost and XGBoost could be tested for the same. **[5]** Various metrics can be used to evaluate the efficiency of a model. Based on what the business/ research considers important, specific metrics and thresholds can be used to make our model more appropriate for the Business problem at hand. On achieving the necessary sufficiency , the model and the methodology is further evaluated in the presence of an industry mentor.

6. Data Dictionary

The data dictionary consists of the feature name, feature definition, corresponding data type and the % of null values in the respective independent feature. The functions datasetname.info() and dasetname.isnull().sum() are used to provide the necessary information.

S no.	Variables/ Features	Definition	Data type	% of null values
0	caseId	The first of four unique internal identification numbers.	object	0
1	docketId	The second of four unique internal identification numbers.	Object	0
2	caseIssuesId	The third of four unique internal identification numbers.	Object	0
3	voteId	The fourth of four unique internal identification numbers.	Object	0
4	dateDecision	The year, month, and day that the Court announced its decision in the case.	object	0
5	decisionType	Cases the Court decides by a signed opinion.	Int64	0
6	usCite	The citation to each case from the official United States Reports (US) and the three major unofficial Reports, the Supreme Court Reporter (S.CT), the Lawyers' Edition of the United States Reports(LEd), and the LEXIS cite.	Object	3.898819
7	sctCite	As mention in point 6.	Object	0
8	ledCite	As mention in point 6.	Object	0.009895
9	lexisCite	As mention in point 6.	Object	0
10	term	The term in which the Court handed down its decision.	Int64	0
11	naturalCourt	A natural court is a period during which no personnel change occurs.	Int64	0
12	chief	The chief justice during whose tenure the case was decided.	Object	0
13	docket	The docket number that the Supreme Court has assigned to the case.	Object	0.089059
14	caseName	The name of the case	Object	0
15	dateArgument	The day, month, and year that the case was orally argued before the Court.	Object	11.575237

16	dateRearg	On those infrequent occasions when the Court orders that a case be reargued, this variable specifies the date of such argument following the same day, month, and year sequence used in the preceding variable (dateArgue).	object	98.022141
17	petitioner	The party who petitioned the Supreme Court to review the case.	Int64	0
18	petitionerState	The place or jurisdiction where petitioner belongs to	float64	79.657369
19	respondent	The party being sued or tried and is also known as the appellee.	float64	0.011132
20	respondentState	The place or jurisdiction where respondent belongs to	float64	72.588878
21	jurisdiction	The Court uses a variety of means whereby it undertakes to consider cases that it has been petitioned to review.	float64	0.011132
22	adminAction	Administrative agency activity occurring prior to the onset of litigation.	float64	72.666213
23	adminActionState	Administrative action may be either state or federal. If administrative action was taken by a state or a subdivision thereof, this variable identifies the state.	float64	0
24	ThreeJudgeFdc	If the case was heard by a three-judge federal district court	float64	0.178119
25	caseOrigin	The court in which the case originated, not the administrative agency	float64	3.810996
26	caseOriginState	If the case originated in a state court, this variable identifies the state.	float64	72.472014
27	caseSource	The court whose decision the Supreme Court reviewed.	float64	2.228957
28	caseSourceState	If the source of the case (i.e., the court whose decision the Supreme Court reviewed) is a state court, this variable identifies the state.	float64	76.122209
29	lcDisagreement	The Supreme Court's majority opinion mentioned that one or more of the members of the court whose decision the Supreme Court reviewed dissented.	float64	0.111324
30	certReason	The Court gives for granting the petition for certiorari.	float64	0.956151
31	lcDisposition	The treatment the court whose decision the Supreme Court reviewed accorded the decision of the court it reviewed; e.g., whether the court below the Supreme Court---typically a federal court of appeals or a state supreme court---affirmed, reversed, remanded, etc. the	float64	13.887068

		decision of the court it reviewed---typically a trial court.		
32	lcDisposition Direction	Whether the decision of the court whose decision the Supreme Court reviewed was itself liberal or conservative as these terms are defined in the direction of decision variable	float64	2.308120
33	declarationUncon	The Court either declared unconstitutional an act of Congress; a state or territorial statute, regulation, or constitutional provision; or a municipal or other local ordinance.	float64	0.011132
34	caseDisposition	The treatment the Supreme Court accorded the court whose decision it reviewed is contained in this variable; e.g., affirmed, vacated, reversed and remanded, etc.	float64	1.475663
35	caseDisposition Unusual	To signify that the Court made an unusual disposition of the cited case which does not match the coding scheme of the preceding variable.	float64	0.011132
36	partyWinning	The petitioning party (i.e., the plaintiff or the appellant) emerged victorious.	float64	0.178119
37	precedentAlteration	If the majority opinion effectively says that the decision in this case "overruled" one or more of the Court's own precedents.	float64	0.011132
38	voteUnclear	The majority opinion in a number of Marshall Court decisions reports that unnamed justices were in disagreement about the resolution of the case.	float64	0.022265
39	issue	The issue for each decision.	float64	0.666708
40	issueArea	Issue areas are both over- and under-specified; especially those of largest size: criminal procedure, civil rights, and economic activity.	float64	0.666708
41	decisionDirection	Whether the Court supports or opposes the issue to which the case pertains.	float64	0.432927
42	decisionDirectionDissent	The majority as well as the dissenting opinion in a case will both support or, conversely, oppose the issue to which the case pertains.	float64	1.958068
43	authorityDecision1	The bases on which the Supreme Court rested its decision with regard to each legal provision that the Court considered in the case	float64	0.644443
44	authorityDecision2	As mentioned in point 43	float64	83.881502

45	lawType	The constitutional provision(s), statute(s), or court rule(s) that the Court considered in the case.	float64	13.916754
46	lawSupp	As mentioned in point 45	float64	13.916754
47	lawMinor	As mentioned in point 45	Object	82.810316
48	majOpinWriter	The author of the Court's opinion or judgment.	float64	19.092090
49	majOpinAssigner	The assigner of the opinion or judgment of the Court	float64	2.576535
50	splitVote	Whether the vote variables (e.g., majVotes, minVotes) pertain to the vote on the first or second issue (or legal provision).	Int64	0
51	majVotes	As mentioned in point 50	Int64	0
52	minVotes	As mentioned in point 50	Int64	0
53	justice	A unique identification number for each of the justices.	Int64	0
54	justiceName	The first initial for the five justices with a common surname (Harlan, Johnson, Marshall, Roberts, and White) and last name of each justice.	Object	0
55	vote	Information about each justice's vote in the case	float64	2.483765
56	opinion	The opinion, if any, that the justice wrote.	float64	2.494898
57	direction	It indicates whether the justice cast a liberal or conservative vote.	float64	5.681242
58	majority	The frequency with which given justices vote with the majority and/or in dissent overall or in certain sets of circumstances.	float64	3.641536
59	firstAgreement	Whether the justice agreed with a dissent or concurrence written by another justice (indicated by the justice's id number).	float64	87.489641
60	secondAgreement	Whether the justice agreed with a dissent or concurrence written by another justice (indicated by the justice's id number). Two agreements are coded---one in this variable and the second in secondAgreement.	float64	98.312821

7. Overview of the final process

1. Variable Categorization
2. Feature removal (Null Values based)
3. Split Data into Train and Test
4. Null Value Imputation
5. Multicollinearity [Cramer's V test]
6. Feature Removal (Information based)
7. Exploratory Data Analysis (EDA) - Visualization
8. Statistical Tests to establish Significance of independent features
9. Representation of Train and Test of original data
10. Model Building
11. HyperParameter Tuning
12. Model Evaluation

The process of building a bankable model to predict the outcome for the petitioning party includes all the above mentioned steps. The methodology involves diligent understanding of each independent feature in the dataset. Techniques such as feature engineering were also employed to obtain the duration of each case. This is followed with various statistical tests to establish the significance of our feature set with respect to the target variable. Before model building we perform appropriate visualizations to understand our data better. Various algorithms such as Logistic regression to Ensemble techniques such as XGBoost are used to obtain the best fit model.

8. Step by Step walkthrough of solution

8.1 Variable Categorization

All datasets contain both categorical (string based) and numerical features. The identification of categorical and numerical features using appropriate techniques allows an analyst to understand the data better. This is done using the select d types function available in the pandas tool. Since there are 61 features in the dataset, it becomes imperative to study each feature in order to aid the process of data cleaning.

8.2 Feature Removal (Based on % Null Values)

Handling Missing values / Null values play an important role in data cleaning. When an independent feature has more 85% of Null values , it is best to remove the feature. This is because all imputation techniques are designed to handle only 5% -10% of null values without distorting the data distribution.

The raw dataset consists of 61 independant features. Some features do not add to the model building process. An example of such a feature would be “docketId”. DocketId represents a unique number that identifies a case (Primary Key). The features cannot be used to train or model and hence they are removed. The list of those feature are mentioned below :

["dateRearg","adminActionState","firstAgreement","secondAgreement","docketId","caseId","caseIssuesId","usCite","sctCite","ldCite","lexisCite","term","docket","caseName","petitionerState","respondentState","adminAction","caseOrigin","caseOriginState","caseSourceState","authorityDecision2","lawSupp","lawMinor","majOpinWriter","majOpinAssigner","justice","justiceName","splitVote"]

8.3 Split data into Train and Test

The dataset is split into train and test before any important process such as data cleaning or EDA. This is done to avoid **data leakage**. Data leakage occurs when the train dataset influences the test dataset. Hence our model may perform better (as the test data is influenced by the train data) but it will lead to erroneous results when new data is provided to the model. The target feature consists of 3 classes.

- **0 :** Unfavourable judgement for the petitioner
- **1 :** Favourable judgement for the petitioner
- **2 :** Unclear Judgement

Unclear Judgements are very rare. Our dataset also establishes the same fact as it contains only 45 records where the target class is 2. Introducing rare labels in our dataset can cause overfitting. Hence we remove those records where the target class is 2.

8.4 Null Value Imputation

Industry standard machine learning libraries such as scikit-learn do not encourage users to incorporate null values in their dataset. Null values must be imputed using appropriate statistical estimates such as mean/median based on the distribution of data. Since all features are nominal in nature (categories with no intrinsic order), we use mode as our statistical estimate. When a single mode value is used to impute all null values in a feature, the data distribution may become distorted. Hence a technique called Hot-Deck imputation is used to perform the necessary replacement of null values in our data.

1. The dataset is split into two. One where the target class is 1 (favourable judgement) and the other where target class 0 (unfavourable judgement)
2. Two mode values are calculated for the same.
3. Now when a null value has to be imputed, the target variable is checked first.
4. Based on the class [1 or 0], the appropriate mode value is imputed

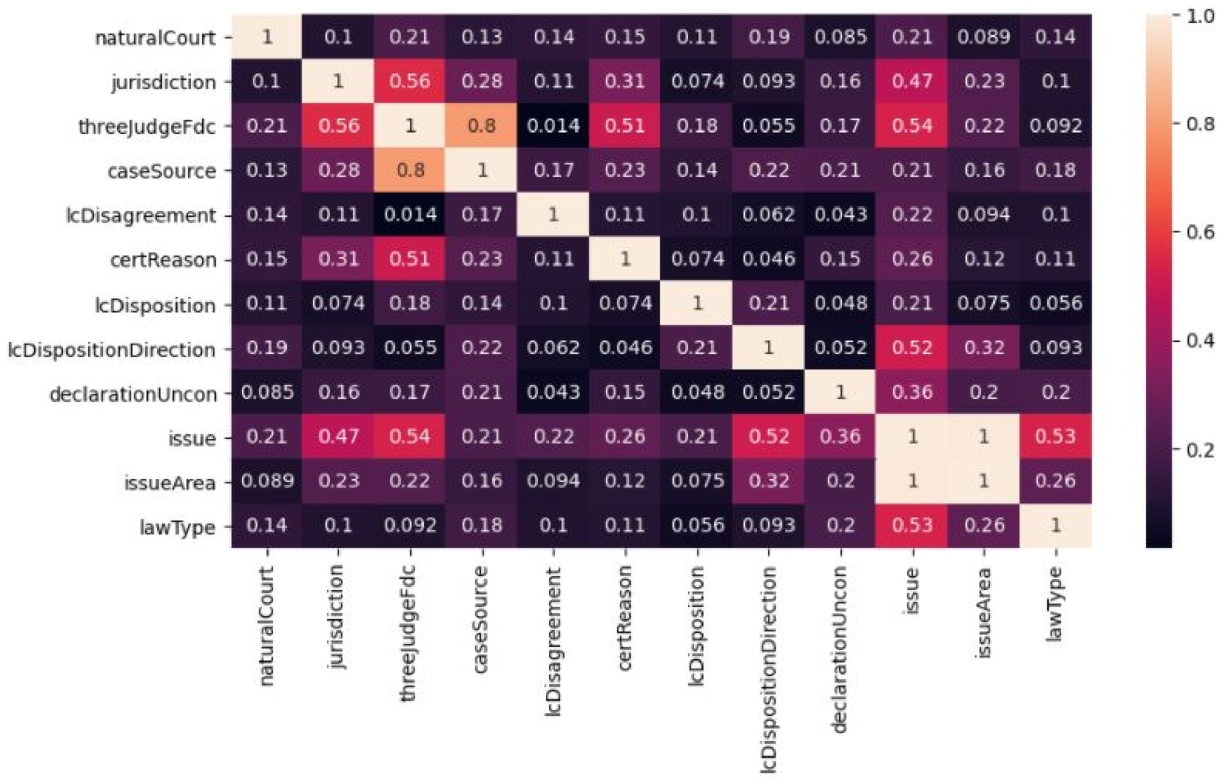
To avoid Data Leakage, Null value imputation is done separately for both train and test data.

8.5 Multicollinearity

Multicollinearity occurs when two or more explanatory variables / independent features are related or have a high association between each other. The presence of multicollinearity makes it difficult to understand which feature actually contributes to predict the target variable. Hence it must be detected and the appropriate features must be removed.

Since most of our features are nominal in nature, a correlation plot or VIF technique will not suffice. Hence we use the **Cramer's V test**. In statistics, Cramér's V (sometimes referred to as Cramér's phi and denoted as φc) is a measure of association between two nominal variables, giving a value between 0 and +1 (inclusive). It is based on Pearson's chi-squared statistic and was published by Harald Cramér in 1946

- Similarly to correlation, the output is in the range of [0,1], where 0 means no association and 1 is full association. (Unlike correlation, there are no negative values, as there's no such thing as a negative association.)



The above plot demonstrates the association of each independent feature with the remaining independent features. Issue and IssueArea seem to have perfect association. This is because Issue is a subcategory of IssueArea. Hence the feature "issue" is removed.

The features lcDispositionDirection and ThreeJudgeFdc also show moderate association. These features will be removed later as a result of a chi square test.

8.6 Feature Removal (Information Based)

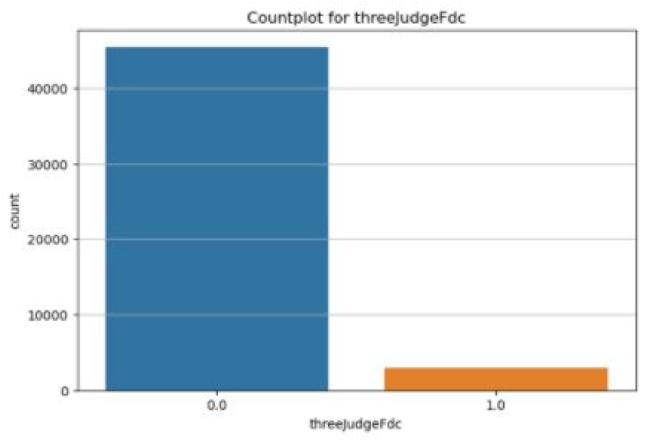
The US Supreme Court dataset includes information about the judgement passed by the apex court, duration of case etc. These features may not be available to the business / Law firm while looking at a new case.Information such as duration of a case is available only after the case has been closed. Hence a law firm may not have access to such information. The data set also includes multiple target variables such as "decisionType". They are removed as well. Such features are removed. The list of features removed are as follows :

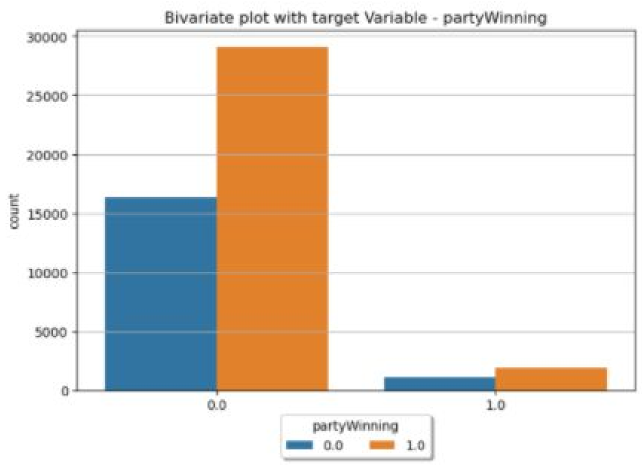
["issue","authorityDecision1","precedentAlteration","caseDispositionUnusual","caseDisposition","majority","voteUnclear","vote_id_num_of_Justices","duration_of_case","decisionType","decisionDirection","decisionDirectionDissent","majVotes", "minVotes", "vote", "opinion", "direction"]

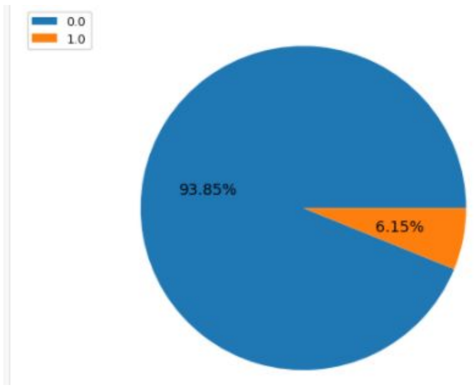
8.7 Exploratory Data Analysis (EDA) - Visualization

The remaining set of features are visualized using univariate and bivariate plots. The univariate plots include pie chart and bar chart. The bivariate plot includes a bar chart with hue as target variables.

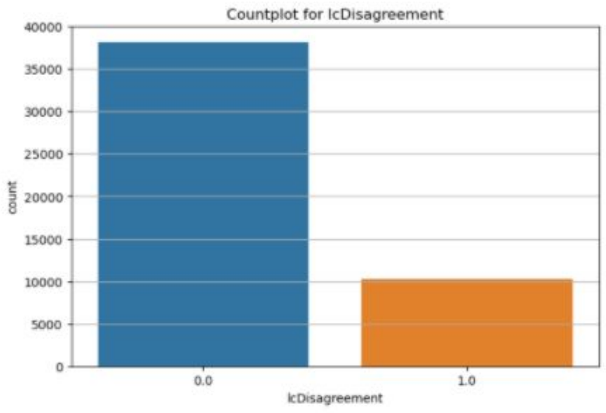
ThreeJudgeFdc : This is a binary variable which indicates if the case was heard by a three-judge federal district court. The countplot shows that very few cases were heard by a three judge district court. This also proves to be beneficial as the petitioner gets a favourable judgement if the threejudgeFdc value is 0.

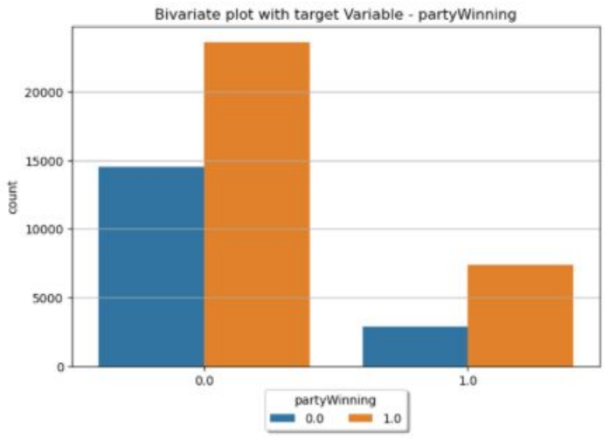


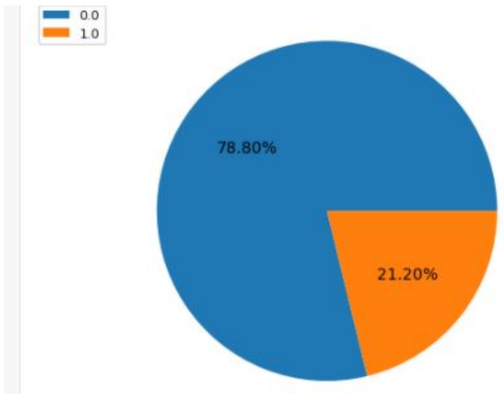




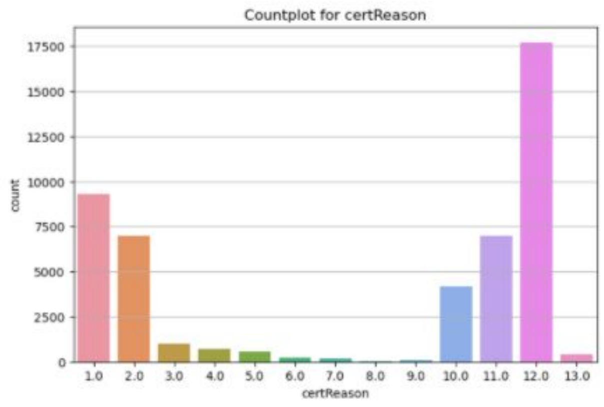
lcDisagreement : This is also a binary variable which indicates that the Supreme Court's majority opinion mentioned that one or more of the members of the court whose decision the Supreme Court reviewed dissented. The univariate analysis and bivariate analysis show that lcDisagreement value does not influence the petitioner getting a favourable judgement

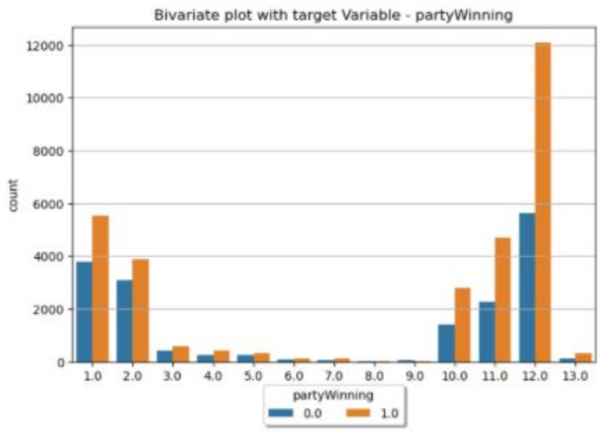


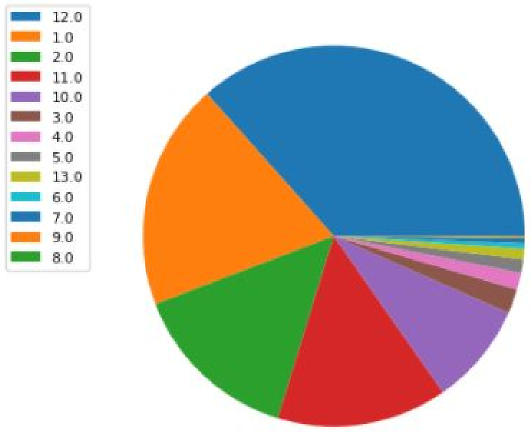




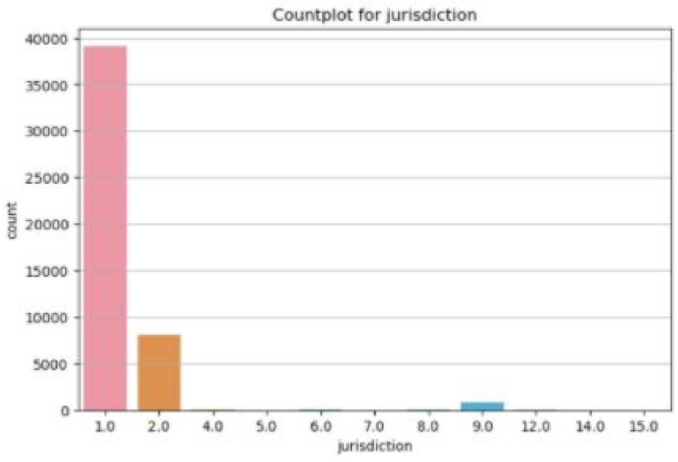
certReason : This variable provides the reason, if any, that the Court gives for granting the petition for certiorari. In law, certiorari is a court process to seek judicial review of a decision of a lower court or government agency**.** From the count plot we can see that a certiorari was asked the maximum number of times when no reason was given. Reason based appeal for certiorari is highest when the court needs to resolve a question presented.

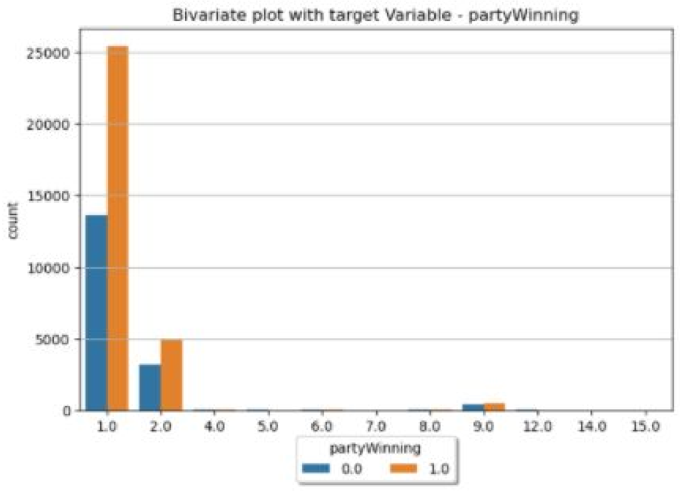


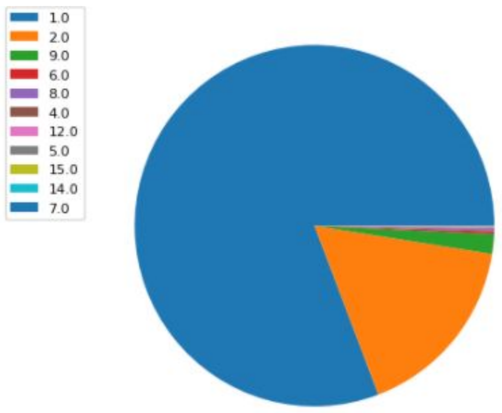




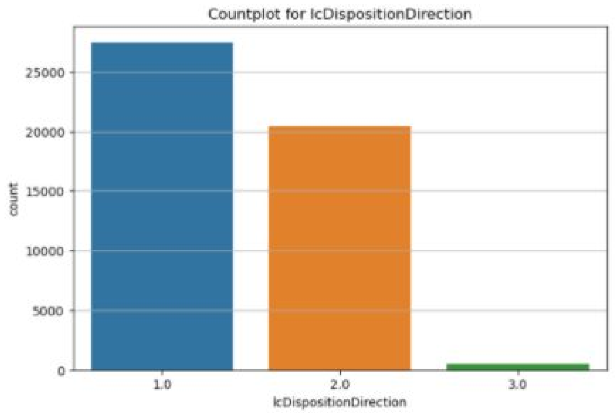
Jurisdiction : The Supreme Court uses a variety of means whereby it undertakes to consider cases that it has been petitioned to review**.** Count plot of jurisdiction shows that the highest number of cases for review pertains to Cert, while 2nd highest number of cases relates to appeal and the remaining sets of cases are negligible. Bivariate plot of cert and appeal depicts that favourable disposition received by partitioning party is more compared to non-favourable disposition.

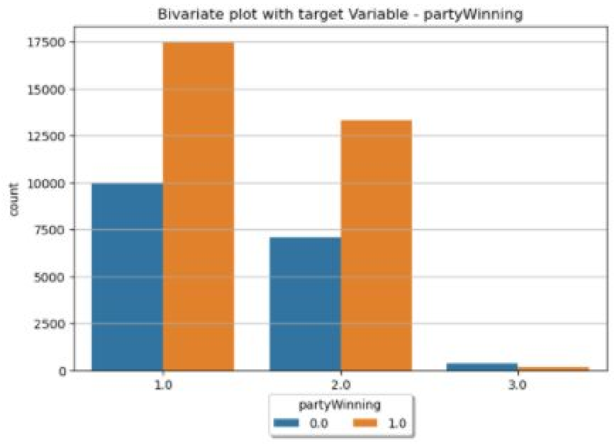


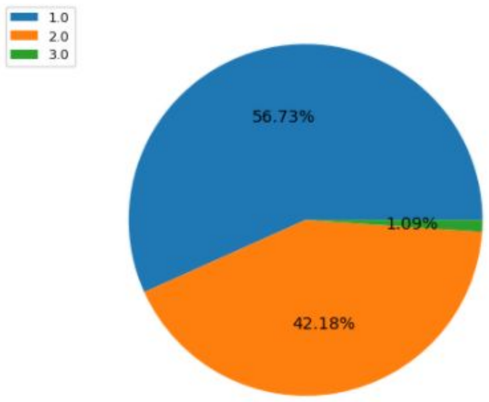




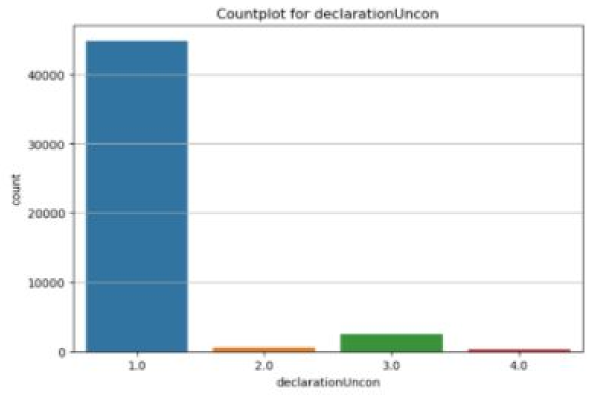
lcDispositionDirection : Count plot for lower court disagreement shows that in the majority of cases, dissent did not happen, and these also are reviewed by the supreme court. Bivariate plot shows that the majority of the cases where dissent did not occur are in favour of the petitioning party. In addition, majority of the cases which are dissented by lower court and further reviewed by supreme court are in favour of the petitioning party.

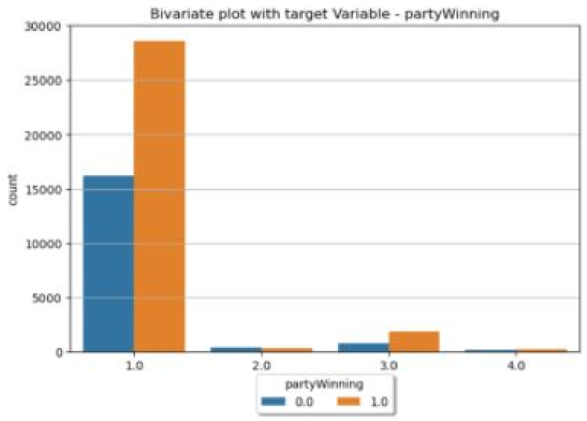


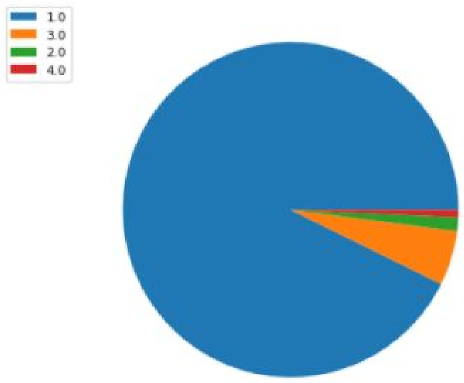




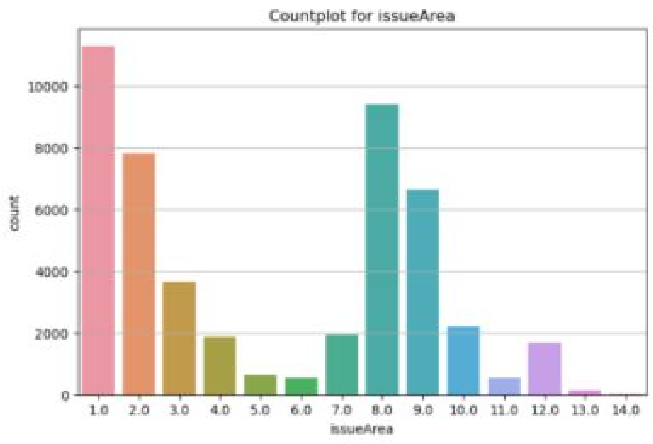
declarationUncon : An entry in this variable indicates that the Court either declared unconstitutional an act of Congress; a state or territorial statute, regulation, or constitutional provision; or a municipal or other local ordinance. Count plot for declarationUncon shows that, majority of cases declared by court are considered as constitutional acts, while few acts based on state or territorial law, regulation, or constitutional provision were considered unconstitutional.

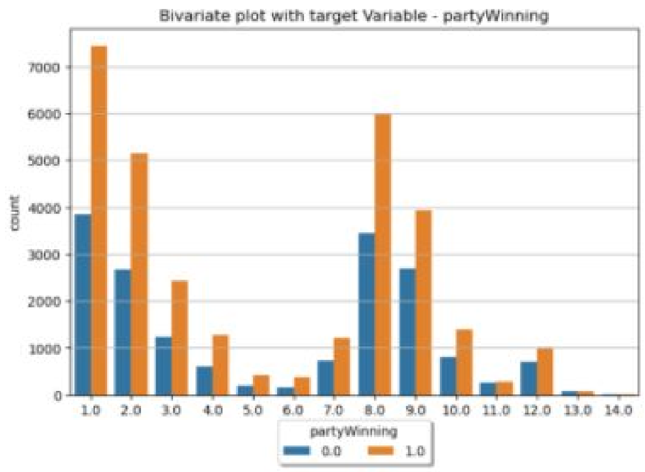


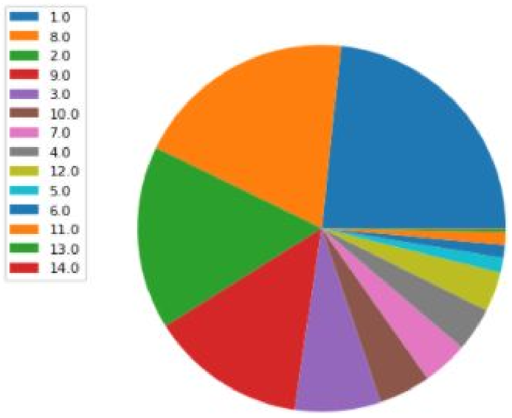




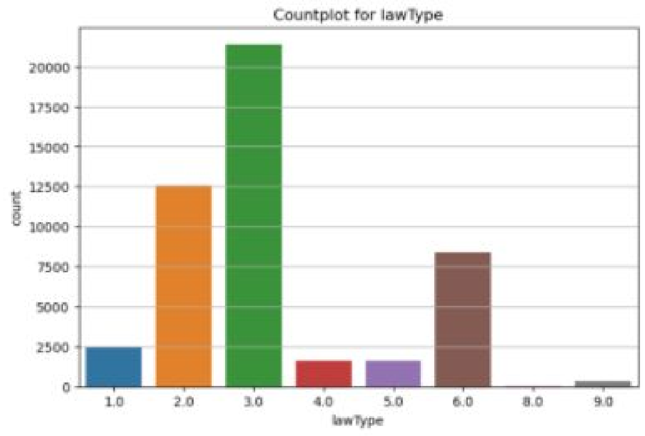
IssueArea : The countplot shows that most cases in our dataset pertain to the issue of 'Criminal Procedure' or 'Economic Activity'. Bivariate plot shows that, majority of the decisions are in favour of the petitioning party where the cases fall under the categories such as criminal procedure, civil rights, first amendment, due process, economic activity, judicial power, federalism.

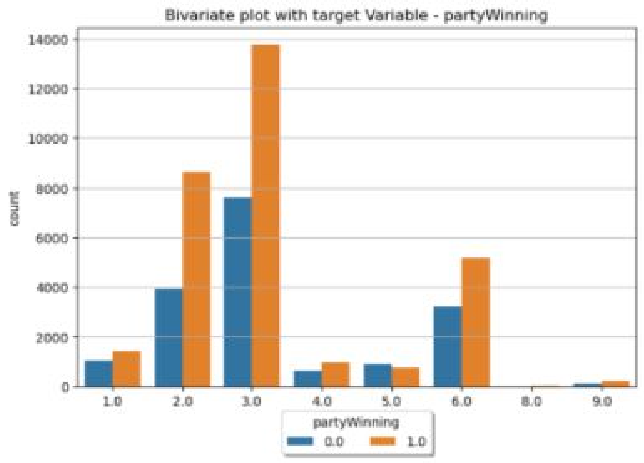


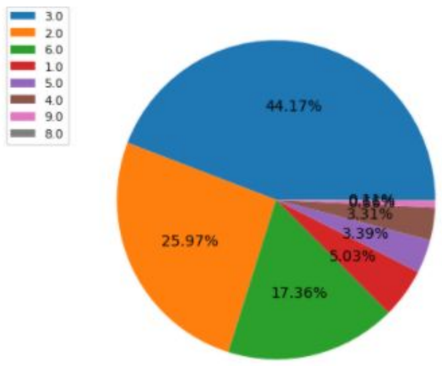




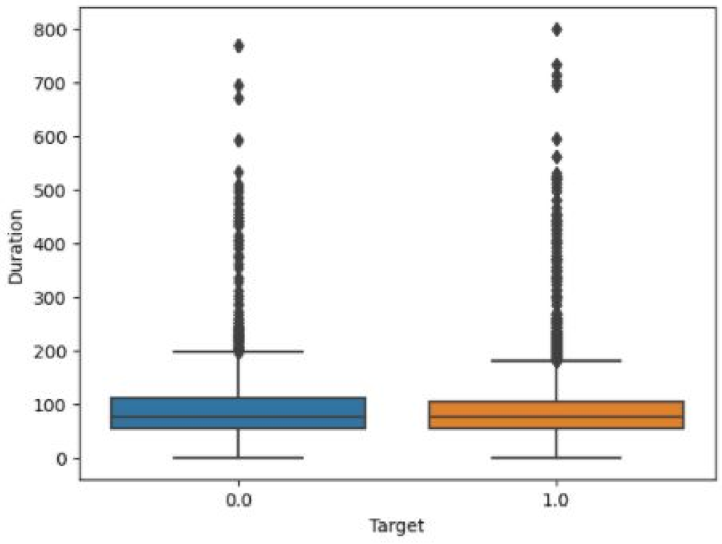
lawType : Count plot for lawtype shows that, the law types considered by court are selected from Constitutional amendment, Federal Statute, Infrequently litigated statutes.







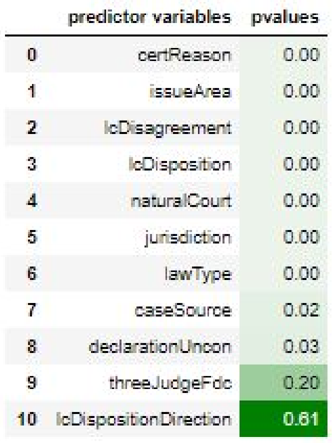
Duration of a case is not used in model building, but can be used to understand if the time taken for a case plays an important role in determining a favourable outcome for the winning party.



From the bivariate Boxplot we can see that the duration of a case plays no major role in influencing the target variable. This can also be verified using a Hypothesis test. The test used for this problem is a Mannwhitney U test. This Hypothesis test is used as both samples are nonparametric and independent in nature. A expected the p value returned is greater than 0.05. Hence the Null Hypothesis is accepted.

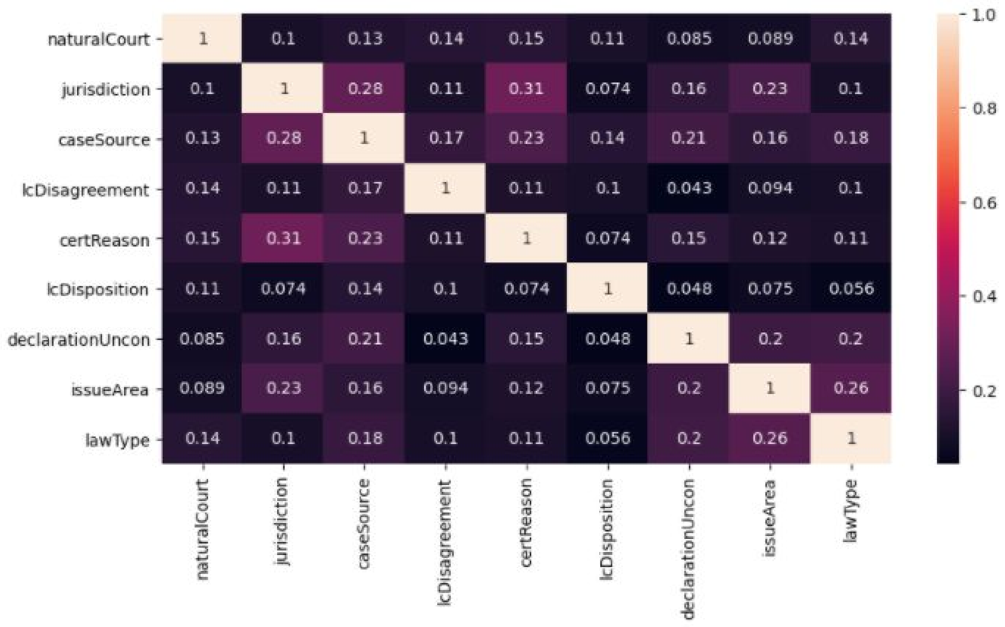
8.8 Statistical Tests for data

Pearson correlation cannot be used for nominal data. Hence we use the chi square test. Select K Best is used to perform a chi_square test for all predictor variables. A **chi_square test** is performed between each predictor variable and the target variable. The P values and the corresponding columns are converted into a dataframe for reference.



	predictor variables	pvalues
0	certReason	0.00
1	issueArea	0.00
2	lcDisagreement	0.00
3	lcDisposition	0.00
4	naturalCourt	0.00
5	jurisdiction	0.00
6	lawType	0.00
7	caseSource	0.02
8	declarationUncon	0.03
9	threeJudgeFdc	0.20
10	lcDispositionDirection	0.61

threejudgeFdc and lcDispositionDirection show high p values. They must be removed. Incidentally, once they are removed, the multicollinearity amongst the variables is also removed. This can be verified by running the cramer's V test. The association between independent features has reduced.



	naturalCourt	jurisdiction	caseSource	lcDisagreement	certReason	lcDisposition	declarationUncon	issueArea	lawType
naturalCourt	1	0.1	0.13	0.14	0.15	0.11	0.085	0.089	0.14
jurisdiction	0.1	1	0.28	0.11	0.31	0.074	0.16	0.23	0.1
caseSource	0.13	0.28	1	0.17	0.23	0.14	0.21	0.16	0.18
lcDisagreement	0.14	0.11	0.17	1	0.11	0.1	0.043	0.094	0.1
certReason	0.15	0.31	0.23	0.11	1	0.074	0.15	0.12	0.11
lcDisposition	0.11	0.074	0.14	0.1	0.074	1	0.048	0.075	0.056
declarationUncon	0.085	0.16	0.21	0.043	0.15	0.048	1	0.2	0.2
issueArea	0.089	0.23	0.16	0.094	0.12	0.075	0.2	1	0.26
lawType	0.14	0.1	0.18	0.1	0.11	0.056	0.2	0.26	1

8.9 Representation of Train and Test of original data

The Null Hypothesis states that Y train and Y test are both representative of the original data. The alternative Hypothesis states that the distribution of data in train and test is not representative of the original data Y. The T test for independent data is performed and p values >0.05 is obtained. Hence the null hypothesis is accepted. This indicates that the ytrain and ytest are representative of the original data.

8.10 Model Building

The required data has been primed for model building. The final dataset consists of 9 independent features. The dataset has been condensed from 61 to 9 features. Now that we have the required data, we can move ahead with training various machine learning algorithms. The ML models used in this project are

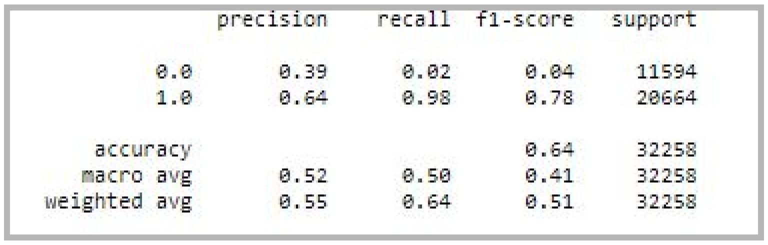
1. Logistic Regression
2. Decision Trees
3. AdaBoost
4. Gradient Boosting
5. XGBoost
6. Random Forests

8.10.1 Logistic Regression

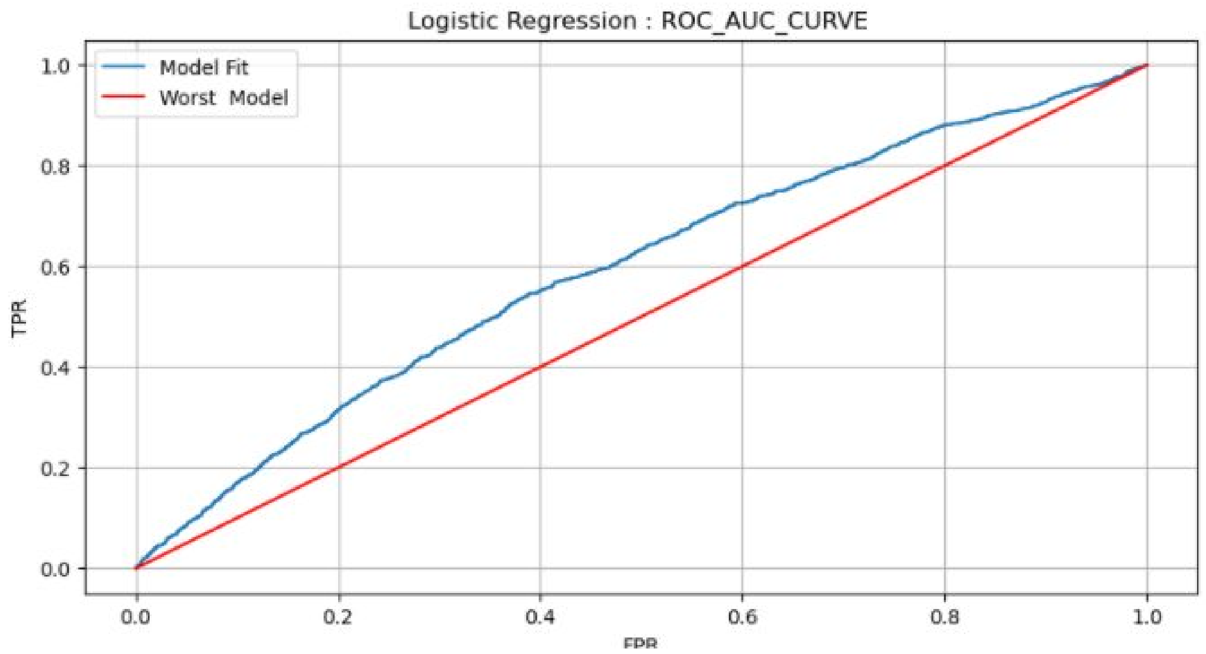
Logistic Regression is a model that comes under Generalized Linear Models. The General Linear Model adopts from the concepts of Linear Regression. The classification model fits a line [log(odds)] and hence is similar to a Linear Model. The model begins with a Y axis that represents probability [0-1]. This is converted into Log(odds). Hence The Y axis now has a range between -infinity and +infinity. Hence a line can be fit. A line is fit based on the input data and the log(odds) is converted back into probability. This is repeated for various candidate best fit lines. The candidate line that returns maximum Likelihood Value is selected as the best fit line. This process is similar to that of Linear Regression and hence the model gets its name Logistic *Regression.*

Most of our data is nominal in nature. When few features are ordinal / nominal, the performance of a General Linear model such as Logistic regression is not hampered. But when all features are nominal in nature, the performance is bound to reduce. This phenomenon is seen in our dataset as well.

Logistic Regression provides for an ROC_AUC Score of 0.59 and accuracy of 0.63. The classification report is as follows :



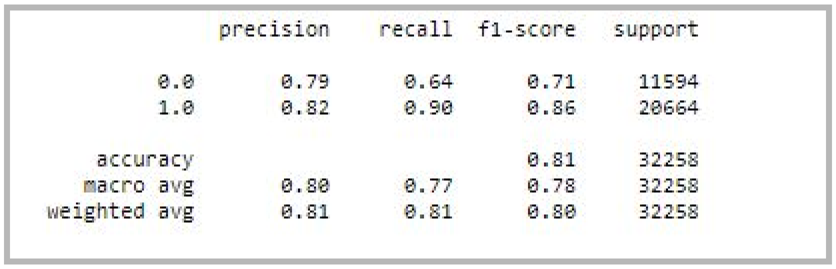
```
              precision    recall  f1-score   support

         0.0       0.39      0.02      0.04     11594
         1.0       0.64      0.98      0.78     20664

    accuracy                           0.64     32258
   macro avg       0.52      0.50      0.41     32258
weighted avg       0.55      0.64      0.51     32258
```



From the ROC_AUC_CURVE we can see that our model results are just above the worst model that can be built. Though the dataset has eighty thousand plus records, since they are all categorical in nature, Logistic regression will not perform well.

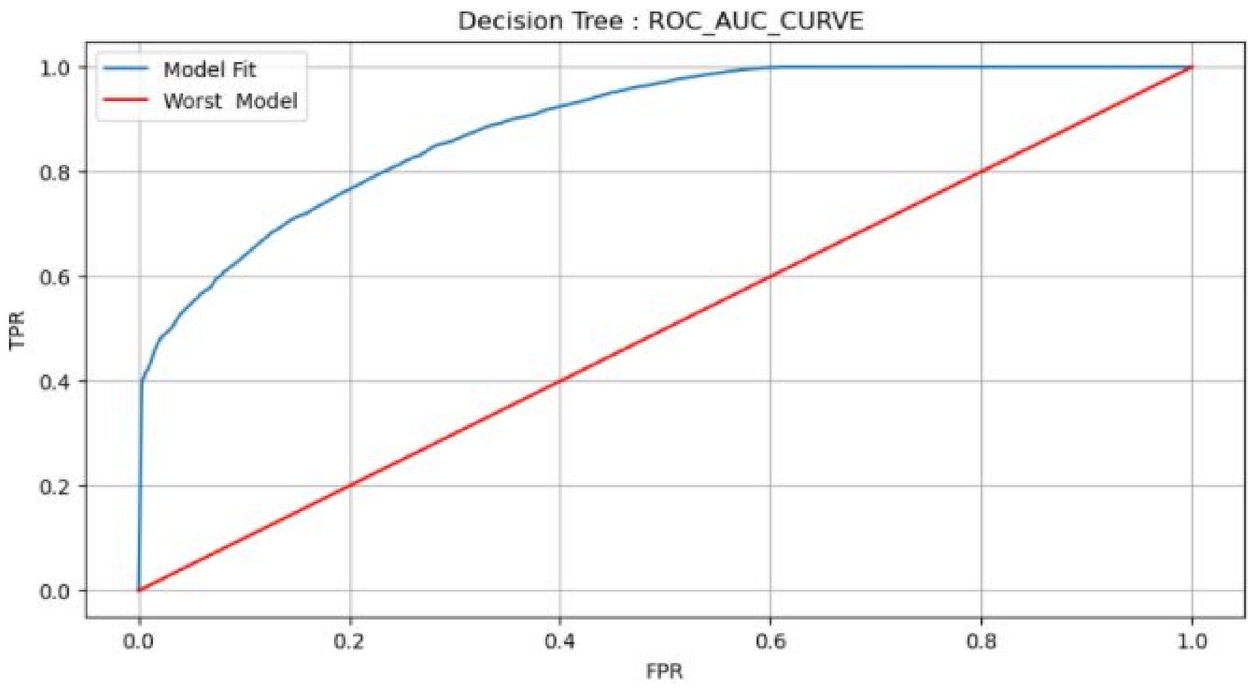
8.10.2 Decision Tree

Decision Tree / Tree based algorithms are one of the best solutions for the dataset involving case details. A tree based algorithm provides any business / Law firm a clear decision path to determine if the petitioner will get a favourable judgement based on the case details. Decision Trees work based on a greedy algorithm. They are known to overfit, but if the dataset is large enough, decision trees and Random forests can prove to provide the best quality of predictions. A decision tree begins with a root node. The root node is split into two or more nodes. The feature to split based on metrics such as entropy, information gain, gain ratio, gini index. Since no line is fit in the backdrop of the model, the performance increases manyfold. This is shown in the classification report of the decision tree model.



```
              precision    recall  f1-score   support

         0.0       0.79      0.64      0.71     11594
         1.0       0.82      0.90      0.86     20664

    accuracy                           0.81     32258
   macro avg       0.80      0.77      0.78     32258
weighted avg       0.81      0.81      0.80     32258
```

There is a considerable increase in metrics such f1 score, precision and recall. The classification report also shows that the models are capable of predicting both Target class 1 and Target class 0 with similar accuracy. This can also be verified using the confusion matrix. The increase in accuracy was expected as a tree based model tends to perform better with categorical features and large training datasets.



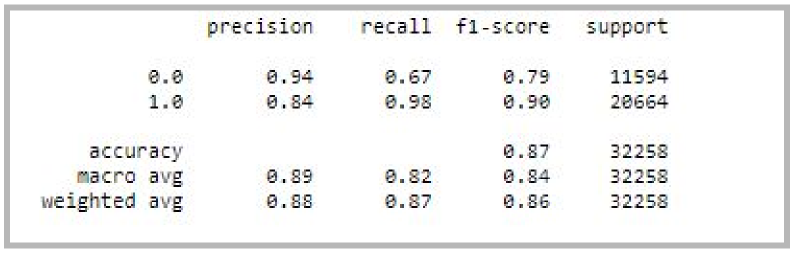
The ROC_AUC_CURVE shows tremendous improvement in the quality of the model fit using the train dataset. An Ideal curve would represent a situation where the value of FPR reduces but the value of TPR remains constant at 1. Given the size of the dataset, an ensemble technique such as Random Forests may be able to perform better.

8.10.3 Random Forests

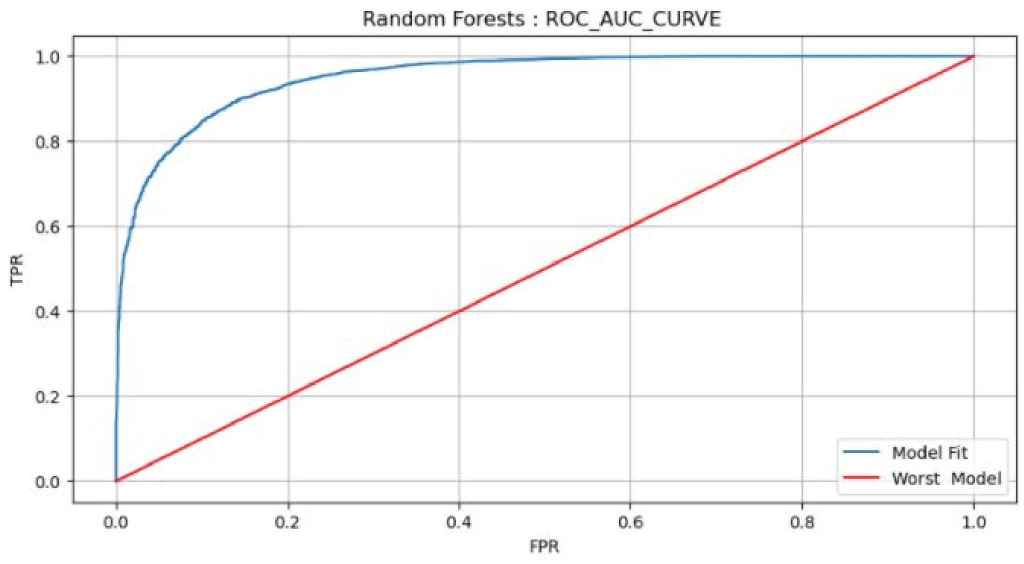
Ensemble learning techniques combine more than one base learner to provide greater performance. Random Forests is an example of ensemble learning. Ensemble learning techniques can be broadly classified into sequential and parallel methods. Parallel ensemble methods can be further divided into Bagging and Stacking techniques. Random Forests comes under the field of Bagging ensemble techniques. This means that each base learner is created independently of other base learners. The independence between base learners may reduce the bias.

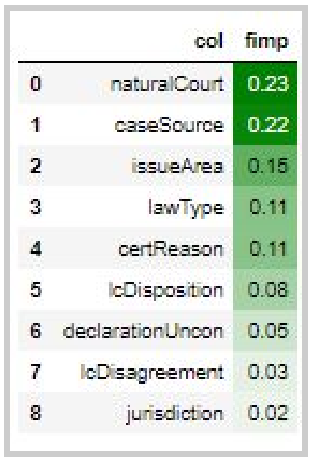
Random Forests work on the basis of Bootstrap sampling Algorithm and random selection of features at each split. The Bootstrap sampling algorithm is a process used to feed different data for each decision tree in the random forest. In simple words, the bootstrap algorithm is just sampling with replacement. For each split in each Decision Tree present in the random forest, the best split is selected from a random selection of features. The number of features to be selected at random can be controlled. Those entries that do not enter the bootstrap dataset are called the Out-of-Bag dataset and can be used to calculate the out of bag error. Random Forests are one of the most accurate algorithms which provide excellent accuracy. This is especially so when the dataset used for training is large. For our dataset that deals with predicting the winning party, the random forests algorithm provides the second highest ROC_AUC Score.

Though the dimensionality of the dataset has been reduced, random forests are capable of handling a large number of independent features. The classification report and ROC_AUC curve is given below. The classification report indicates good efficiency of the model in predicting both target class 1 and 0. The ROC_AUC curve shows that the TPR value remains constant even though the value of FPR decreases, This is ideal for any machine learning model. It indicates that the model has separated the target class 1 and 0 with good accuracy. The feature importance values for the predictor variables are given below using appropriate charts.

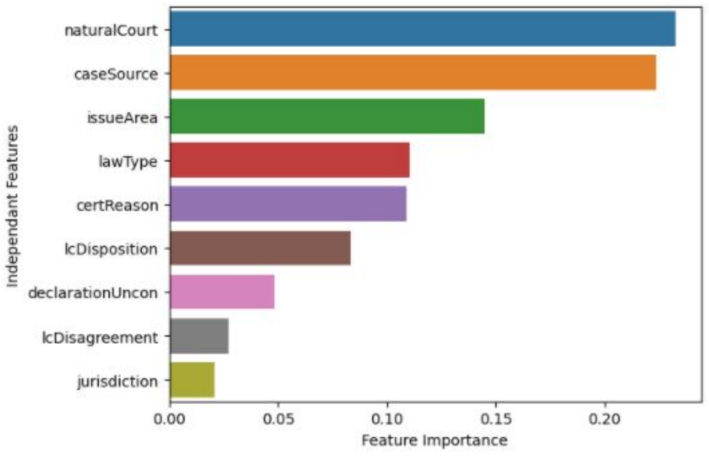


```
              precision    recall  f1-score   support

         0.0       0.94      0.67      0.79     11594
         1.0       0.84      0.98      0.90     20664

    accuracy                           0.87     32258
   macro avg       0.89      0.82      0.84     32258
weighted avg       0.88      0.87      0.86     32258
```





	col	fimp
0	naturalCourt	0.23
1	caseSource	0.22
2	issueArea	0.15
3	lawType	0.11
4	certReason	0.11
5	lcDisposition	0.08
6	declarationUncon	0.05
7	lcDisagreement	0.03
8	jurisdiction	0.02

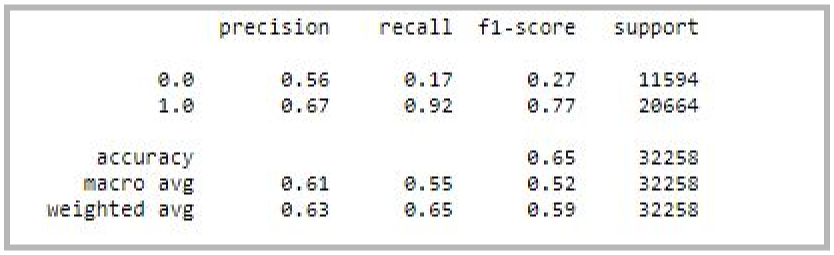


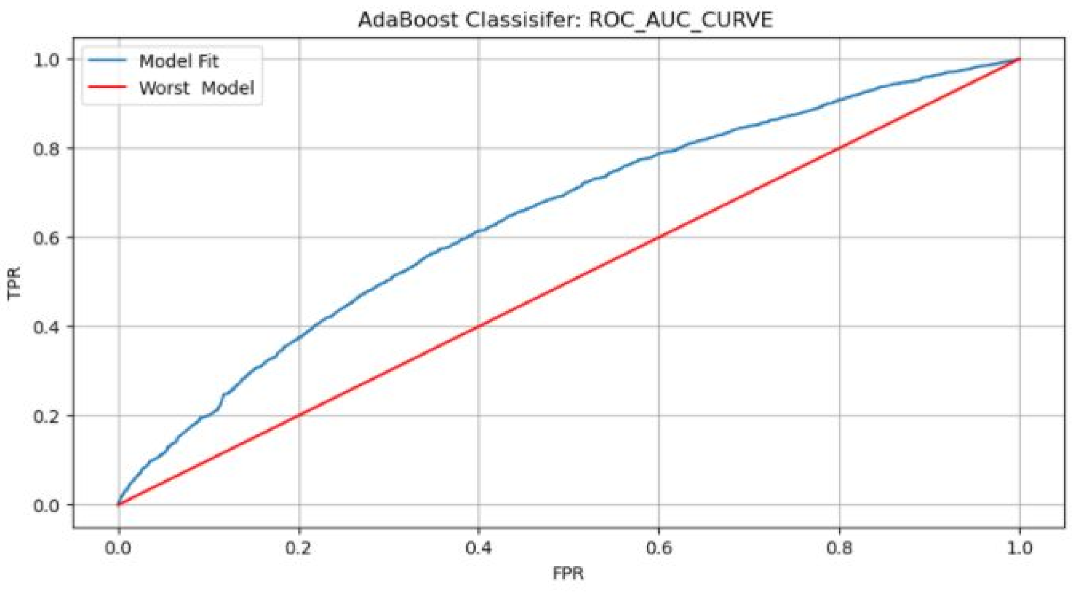
8.10.4 AdaBoost

AdaBoost is one of the first boosting algorithms to be adapted in solving practices.Works by putting more weight on difficult to classify instances and less on those already handled well. The weak learners in AdaBoost are decision trees with a single split, called decision stumps.Can be used for both regression and classification

At each iteration, adaptive boosting changes the sample distribution by modifying the weights attached to each of the instances. It increases the weights of the wrongly predicted instances and decreases the ones of the correctly predicted instances. The weak learner thus focuses more on the difficult instances. After being trained, the weak learner is added to the strong one according to its performance (so-called alpha weight). The higher it performs, the more it contributes to the strong learner.

In Random Forests, trees of a certain depth are built. The AdaBoost model uses trees with single splits called stumps. On their own stumps are weak models as they do not capture the relationship between the independent features and the target variable. But many such stumps together form a decent predicting model called AdaBoost. The first step involves initialising sample weights for each record in the dataset. The first stump is created where the root node is split based on the gini error metric. Each stump created in AdaBoost has its say in the final classification. The amount of say for a stump in the model depends on the error it generates. If the stump has high error, the amount of say decreases. Based on the errors made by the first stump, the sample weights are updated and normalized. The new sample weights are used to create a new dataset based on which the second stump is created. Based on the sample weights, those records that were wrongly classified by the first stump will be repeated more often in the second dataset. Hence the errors from the first stump determine how the second stump is made. AdaBoost represents a sequential ensemble learning technique as the errors of the first stump determine how the second stump is built. Though AdaBoost employs a boosting algorithm, the base learners present in the model are weak as the max depth is 1 for all base learners. This hinders the model from learning the complete relationship between independent and dependent features.


```
              precision    recall  f1-score   support

         0.0       0.56      0.17      0.27     11594
         1.0       0.67      0.92      0.77     20664

    accuracy                           0.65     32258
   macro avg       0.61      0.55      0.52     32258
weighted avg       0.63      0.65      0.59     32258
```

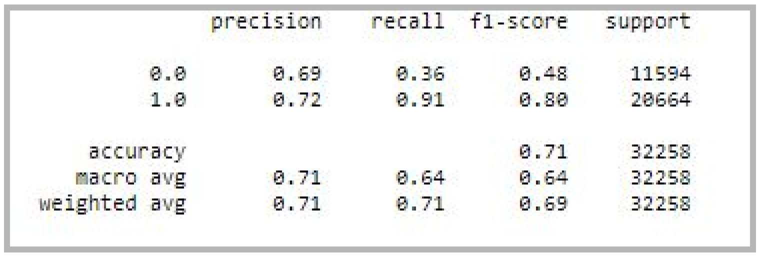


The classification report indicates a model performance greater the Logistic Regression. The classifier is still not able to predict both classes of the target variable with equal accuracy. The ROC_AUC curve also shows a minimal increase (with respect to Logistic Regression) in the ROC_AUC_SCORE (0.65). Though the model does not provide best results, it is useful to understand the working of a Boosting algorithm.

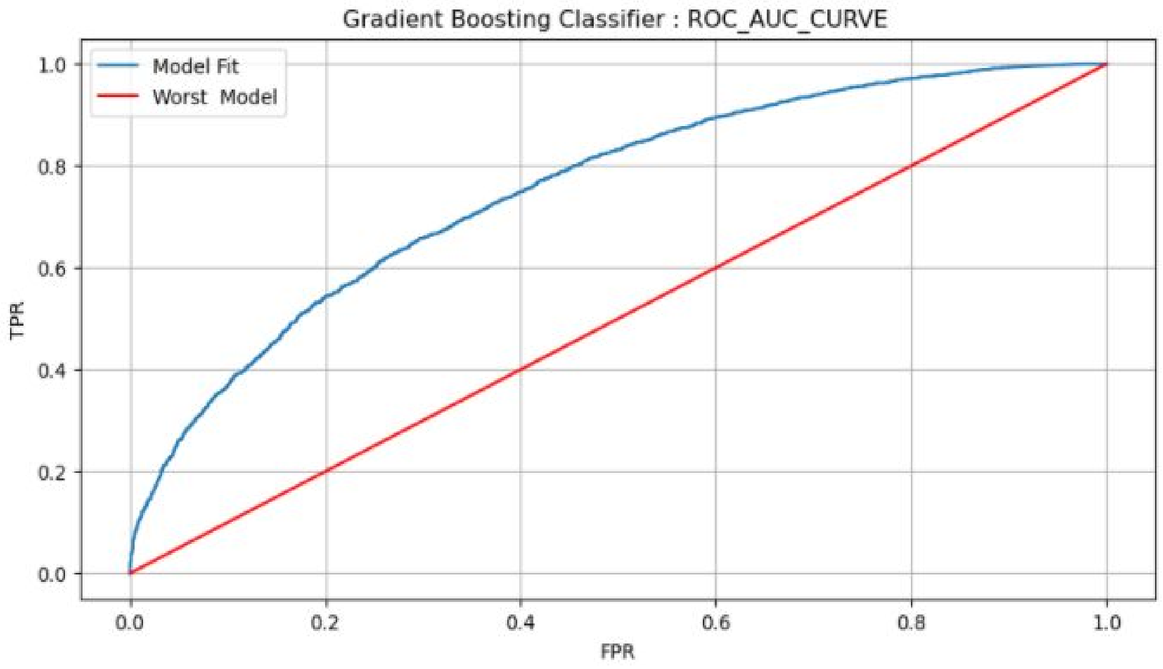
8.10.5 Gradient Boosting Classifier

The intuition behind gradient boosting algorithms is to repetitively leverage the patterns in residuals and strengthen a model with weak predictions and make it better. Once we reach a stage where residuals do not have any pattern that could be modelled, we can stop modelling residuals (otherwise it might lead to over-fitting). Algorithmically, we are minimizing our loss function, such that test loss reaches its minimum. **Gradient boosting doesn't modify the sample distribution**. Instead of training on a new sample distribution, the weak learner trains on the remaining errors (called pseudo-residuals) of the strong learner. It is another way to give more importance to the difficult instances. At each iteration, the pseudo residuals are computed and a new tree is fitted to those residuals. Thus Gradient Boosting Classifier uses the Gradient descent optimization process where the entire model is updated.

By fitting new models to the residuals, the overall learner gradually improves in areas where residuals are initially high. To prevent overfitting, learning rate is introduced. The only disadvantage with this approach is that it is difficult to interpret and is prone to overfit the training data.This technique is superior to the process used in building an AdaBoost classifier model. This is verified as the model performance increases when compared to the AdaBoost Classifier Model. The classification report shows a decent F1 score. The ROC_AUC_SCORE obtained is 0.75.



```
              precision    recall  f1-score   support

         0.0       0.69      0.36      0.48     11594
         1.0       0.72      0.91      0.80     20664

    accuracy                           0.71     32258
   macro avg       0.71      0.64      0.64     32258
weighted avg       0.71      0.71      0.69     32258
```

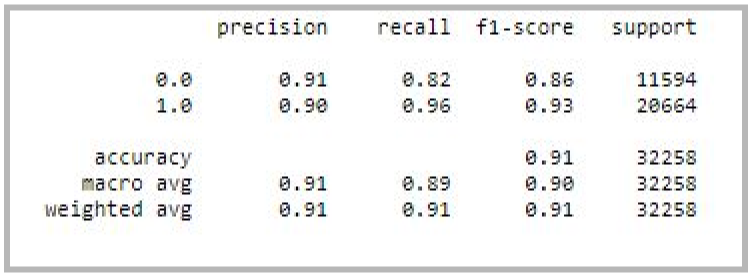


8.10.6 Extreme Gradient Boosting Classifier [XGBoost]

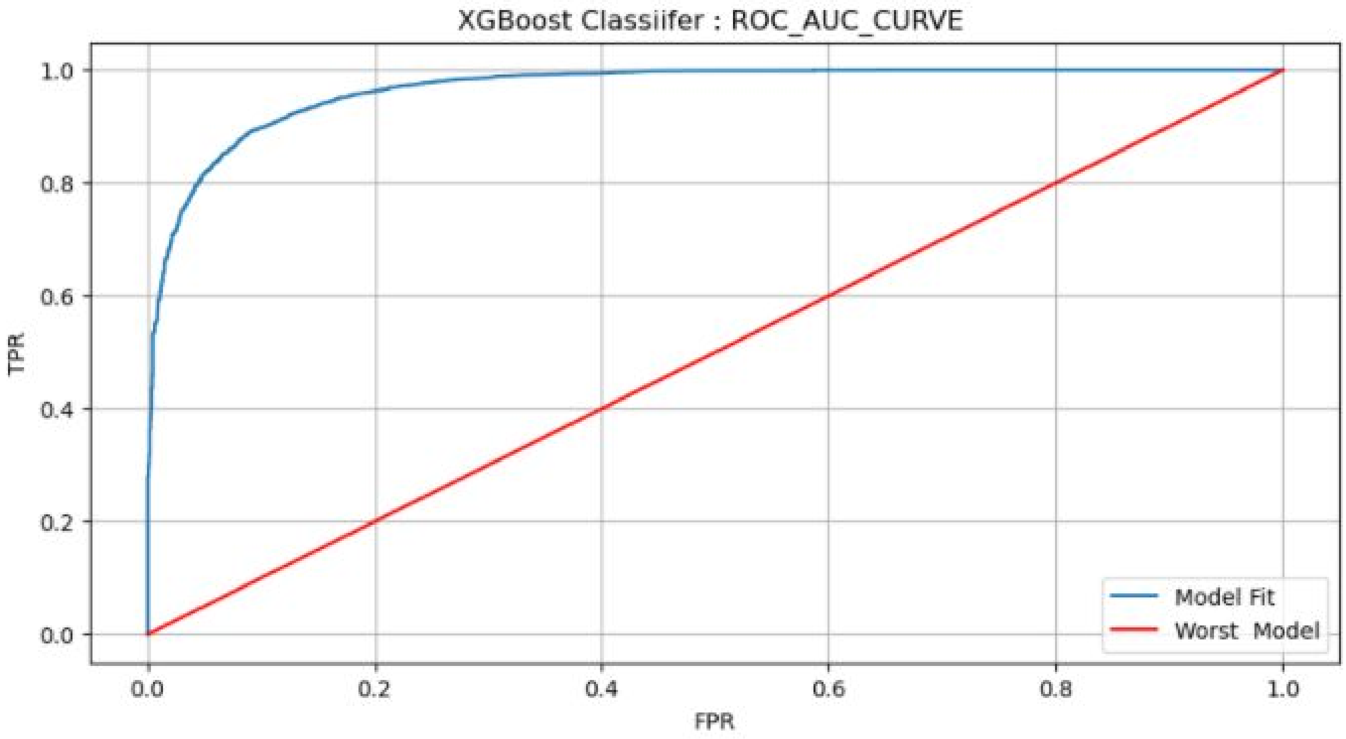
XGBoost (Extreme Gradient Boosting) is an optimized distributed gradient boosting library. It uses a gradient boosting (GBM) framework at its core. Yet, it is better than the GBM framework alone. Serves the purpose of Generalization and Reduction in error. It's faster because it is written in C++ and uses optimization algorithms for the same. XGBoost internally has parameters for cross-validation, regularization, missing values, tree parameters. Some reason what XGBoost proves to be the best classifier for large datasets is as follows

1. Approximate Greedy Algorithm : XGBoost employs an approximate greedy algorithm with the help of parallel learning and **weighted quantile sketch** to process large datasets. This is why the algorithm tends to do so well when the dataset is huge. On the contrary techniques such as XGBoost perform poorly when the dataset is small. The approximate greedy algorithm is only used when the dataset is huge, otherwise the regular greedy algorithm is used when the dataset is small.
2. Regularization: Standard GBM implementation has no **regularization** like XGBoost, therefore it helps to reduce overfitting. In fact, XGBoost is also known as 'regularized boosting' technique.
3. Faster Processing: XGBoost implements **parallel processing** and is blazingly faster as compared to Gradient Boosting Machines.
4. Missing Values : XGBoost uses **Sparsity Aware Split** finding inorder to compute for missing values in both training and testing data.
5. Tree Pruning: XGBoost o makes splits up to the max depth specified and then starts pruning the tree backwards and removing splits beyond which there is no positive gain. When the gamma value is high, it becomes easier to prune trees in the XGBoost model.
6. Built In Cross-Validation : XGBoost allows users to run a cross-validation at each iteration of the boosting process and thus it is easy to get the exact optimum number of boosting iterations in a single run.
7. Cache Aware Access : Cache memory is fastest of all available memory options in a computer. If one wants to maximize performance, the goal will be to maximize what can be done using cache memory. In some cases when there is more than one hard drive, XGBoost uses a technique called sharding to speed up disk access.

XGBoost classifier shows that Machine Learning models can be much more than applied statistics. The dataset used for the project has more than eighty thousand records. This makes it ideal for ensemble techniques such as Random Forest and XGBoost classifier.

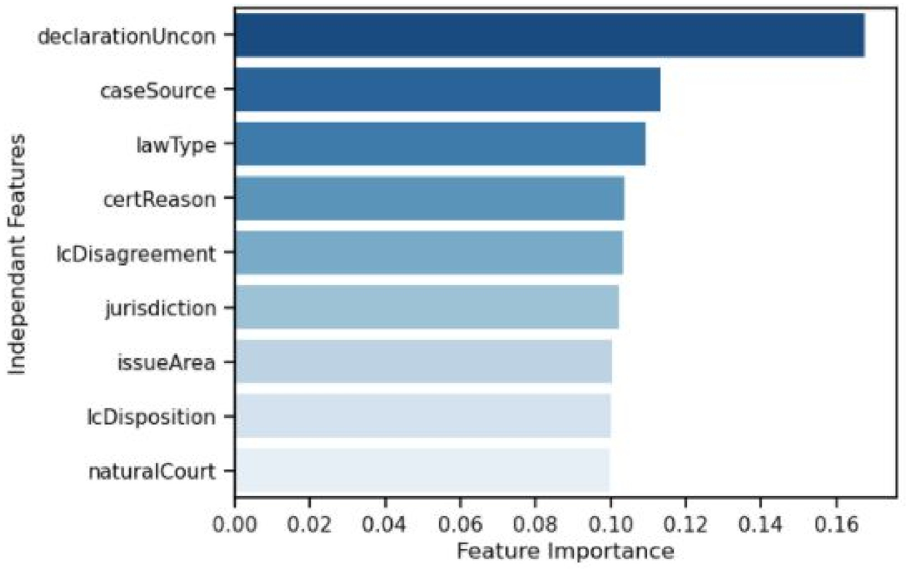

```
              precision    recall  f1-score   support

         0.0       0.91      0.82      0.86     11594
         1.0       0.90      0.96      0.93     20664

    accuracy                           0.91     32258
   macro avg       0.91      0.89      0.90     32258
weighted avg       0.91      0.91      0.91     32258
```

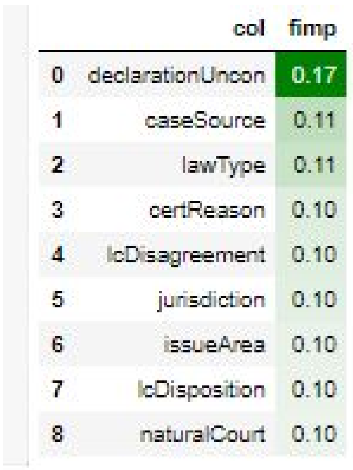
F1 score represents the harmonic mean of two classification metrics - Precision and Recall. Since accuracy is not always the best metric to use in classification problems, we use F1 Score. The classification report generated for the XGBoost classifier shows a good f1 score for both target class 1 and 0. This means that the algorithm has separated the 2 classes efficiently in the model building process. In all the previous mentioned algorithms, the f1 score for target class 0 was always lesser by a good measure. But the XGBoost classifier provides a balanced and efficient prediction of both target classes



The ROC_AUC curve determines how well the model separates between the classes present in the Target variable. If the classes 1 and 0 are separated well, the TPR value will remain constant despite constant decrease in FPR. We can see that our model shows constant TPR even though the False Positive Rate reduces consistently. This indicates a high ROC_AUC_Score. The XGBoost model produces a score of 0.96, which is the highest of all classification algorithms used in this project.

Most classification models are capable of providing values for feature importance. These values determine which features contribute the most in decreasing the error. The feature importance values are shown graphically using a bar plot. DeclarationUncon, Case Source and Law Type seem to be the features that contribute the most.



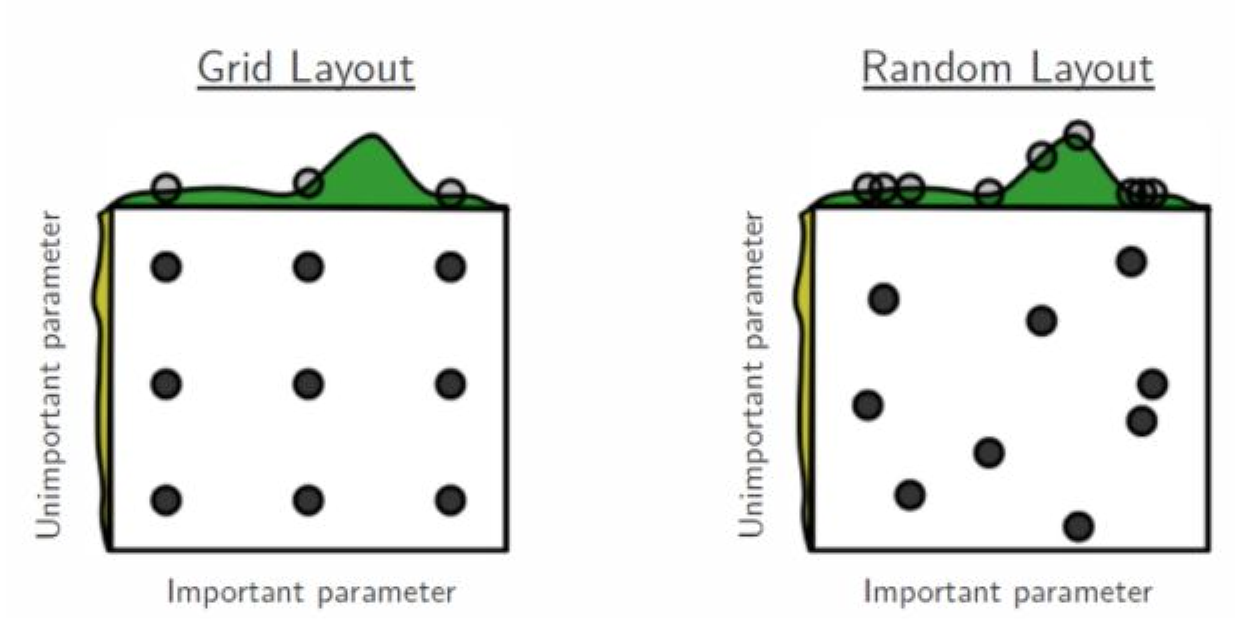


	col	fimp
0	declarationUncon	0.17
1	caseSource	0.11
2	lawType	0.11
3	certReason	0.10
4	lcDisagreement	0.10
5	jurisdiction	0.10
6	issueArea	0.10
7	lcDisposition	0.10
8	naturalCourt	0.10

8.11 HyperParameter Tuning

The values which are learnt by the hypothesis function in a ML model are called parameters. Hyper Parameters are those values which are not learnt by the model. They are specified by the practitioner. When a machine learning algorithm is tuned, we are tuning the hyperparameters to discover the parameters of the model that result in the best predictions.

Random Forest and XGBoost classifiers give exceptional model performance even without explicit HyperParameter tuning. HyperParameter Tuning is exhaustive in terms of time and computer resources. Hence the process is done for those models where the score could be improved. The two well known tuning methods are Grid Search CV and Randomized Search CV.

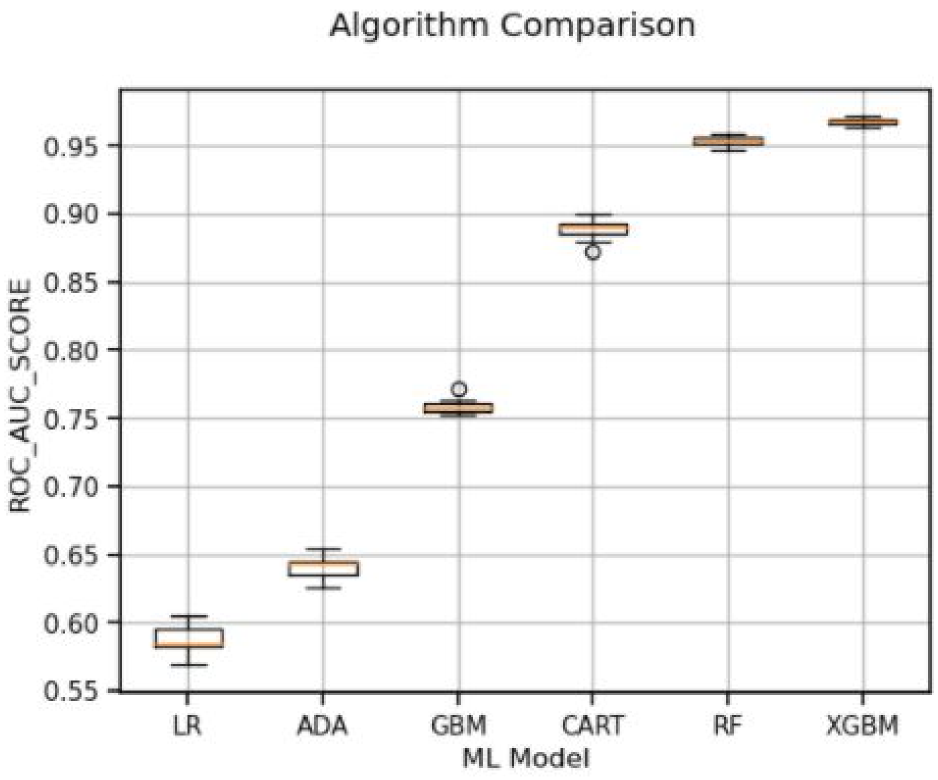


Grid Search involves a brute force technique where all values given by the analyst are used to check the performance of the model. Hence a model is built for each possible combination of all of the hyperparameter values provided, the parameters that provide best model performance are selected. This method is inefficient and time exhaustive. As shown in the above diagram GridSearchCV is inefficient as the best parameters that provide maximum efficiency may never be evaluated.

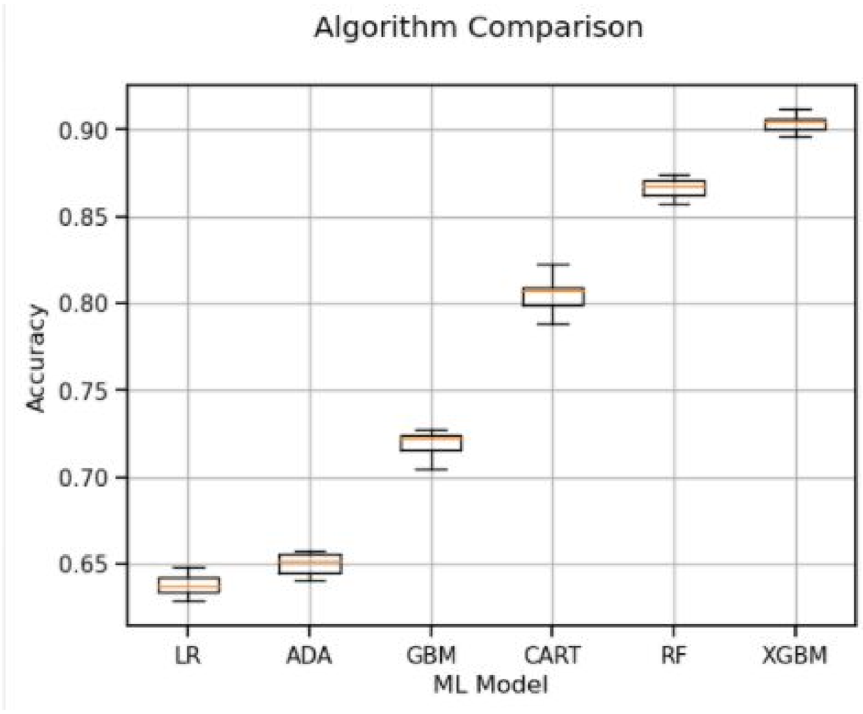
Randomized Search CV is capable of finding parameters where the model performance is high. In this technique we provide a statistical distribution for each hyperparameter from which values may be randomly sampled. We can also define how many iterations we'd like to build when searching for the optimal model. For each iteration, the hyperparameter values of the model will be set by sampling the defined distributions above. Thus random search has an improved exploratory power and can focus on finding the optimal value for the important hyperparameter.

8.12 Model Evaluation

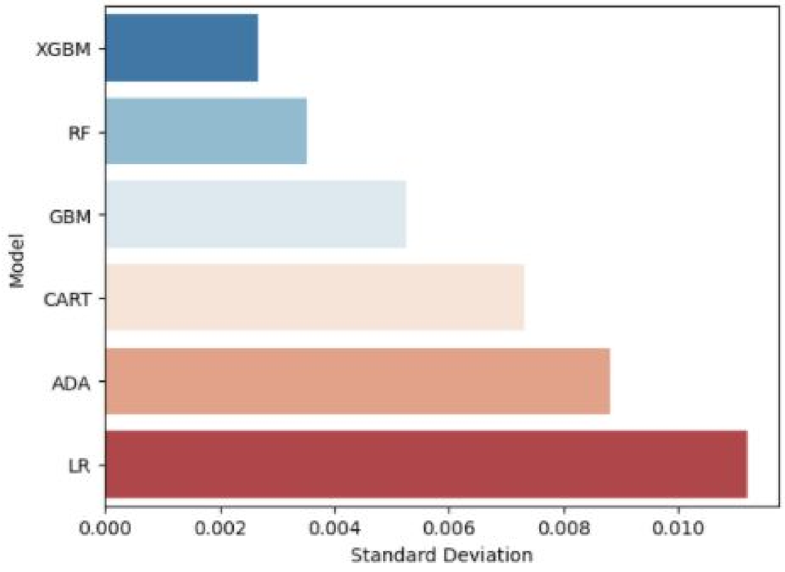
Six models have been tested for this project out of which Two are base learners and four are ensemble techniques. The evaluation of a model can be done using the train-test split procedure and the cross validation approach. Cross Validation offers more reliability as the results are generated multiple times. The results from a 10 fold cross validation can be used to create a boxplot for each model. The cv score function is used to do the same for each model. The resulting ROC_AUC_scores are used to create a boxplot for the respective model. The boxplot shows whether the model is capable of producing results with minimum variance in performance. The plot below visualizes the variance in performance for each machine learning model used in this project. Logistic Regression and AdaBoost classifier provides the worst results among six classifiers. Relatively the variance is higher in these two algorithms with lesser scores. Gradient Boosting Algorithm and Decision Trees provide decent results with occasional outliers. Random Forest and XGBoost provide the ROC_AUC scores with minimum variance. The minimum variance is an indicator that the model is consistent at delivering good results.

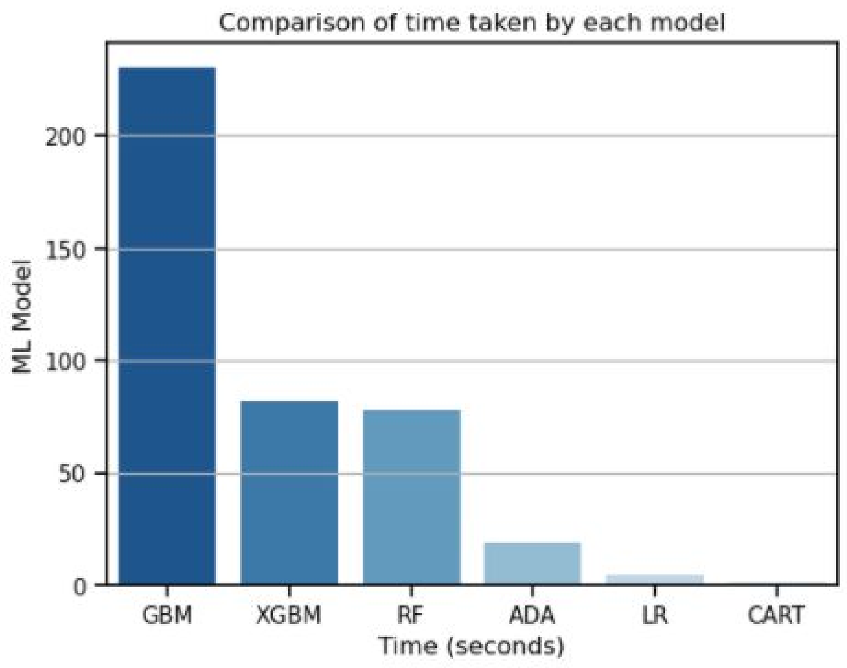


The same process can be repeated with the scoring metric as “accuracy”. The results are almost similar with a few changes. We can see that the decision tree classifier has relatively high variance as it's using only one tree to predict the outcome for the petitioning party. Random Forests and XGBoost classifiers are able to produce results with consistent accuracy.



The time taken for a model to fit over a 10 fold cross validation process is visualized in the figure below. The amount of time taken to build a model and its performance can be compared. We can see that the Gradient Boosting classifier takes the highest amount of time to finish the task. Random Forests are able to complete the same with a lesser amount of time. The Highlight of the graph is the time taken by the XGBoost model. As discussed earlier, the model handles large datasets well. Thus the XGBoost Classifier Algorithm completes the task in less than half the time taken by GBM. The Standard Deviation in model performance can be visualized using a horizontal bar chart





9. Implications

Considering and taking this as a business model, Legal Analytics can be widely used by private or even government law firms to actually predict the outcome of a particular case, whether the outcome would be 1 (In favour of the Petitioner) or 0 (Not in favour of the Petitioner) provided the information is with them. Based on the selected independent features which have influence over our target variable. This will save time, as whether to take on a certain case or not, increase the accuracy of predicting the winning party beforehand, and help a particular firm to be more competitive in their respective field.

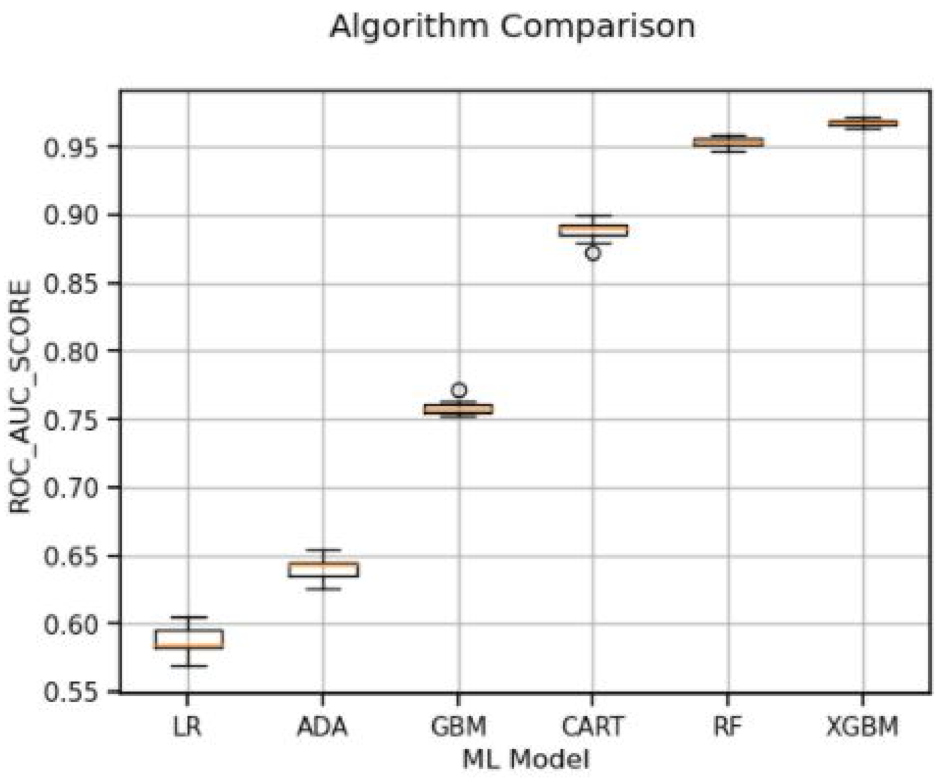
Now taking into consideration the social benefits, any person, or even family, will have an idea of whether the case will go in their favour or not. According to that prediction, one can think of alternate solutions if it’s not in favour of him/her or go ahead with full confidence if it’s in their favour.

10. Limitations

As such, we could not come across any limitations in our final model built by XGBoost Classifier. Other models had few limitations but none of them is the final model so not much attention is required there. Certain limitations would have arisen during VIF as none of our variables are continuous in nature. But we did the Cramer’s V test, which is used to test for categorical or nominal data. We have covered almost all the loopholes and potential limitations as much we could have done.

11. Conclusion

Finally, we conclude that out of all 6 classifiers used for model building, we saw that XGBoost Classifier Algorithm performed the best with very accurate and decent test results and scores. Thus, we can say that it is a reliable model to predict the favourable winning party based on certain independent features used in our model building as important features to hamper our target variable.



From the above figure, we obtained the best ROC_AUC_SCORE using the XGBoost Classifier. Thus, we conclude that this is the best classifier to predict the winning party with assured reliability.