Labor Condition Application Disclosure

Predicting the Case Status of LCA Application

Abstract

Labor condition application (LCA), is a document that a prospective H-1B employer files when it seeks to employ nonimmigrant workers at a specific job occupation. In this Analysis, we are trying to predict the case status of an application submitted by the employer to hire non-immigrant workers under H1-B visa program.

Analysis By

Anuj Verma

Depaul University

anuj.vermacom@gmail.com

Table of Contents

[Project Description 2](#_Toc515915153)

[Data Background 2](#_Toc515915154)

[Introduction to Dataset 2](#_Toc515915155)

[Project Goal 2](#_Toc515915156)

[Dataset Information 2](#_Toc515915157)

[Attribute List 2](#_Toc515915158)

[Target Attribute 4](#_Toc515915159)

[Data Cleaning 4](#_Toc515915160)

[Data Filtering 4](#_Toc515915161)

[Column Rename 4](#_Toc515915162)

[Checking for Data Correctness 4](#_Toc515915163)

[Data Transformations: 5](#_Toc515915164)

[Train Test Split 6](#_Toc515915165)

[Data Scaling 6](#_Toc515915166)

[Exploratory Analysis 6](#_Toc515915167)

[Data Analysis 7](#_Toc515915168)

[Model Comparison 8](#_Toc515915169)

[Model Performance 8](#_Toc515915170)

[Hyper Parameterization of Gradient Boosting Classifier 9](#_Toc515915171)

[Summary 10](#_Toc515915172)

[Future Works 10](#_Toc515915173)

[Appendix 10](#_Toc515915174)

[References 10](#_Toc515915175)

# Project Description

## Data Background

This dataset is from Office of Foreign Labor Certification (OFLC) department. The OFLC is a division of the U.S. Department of Labor. The main duty of OFLC is to assist the Secretary of Labor to enforce part of the Immigration and Nationality Act (INA), which requires certain labor conditions exist before employers can hire foreign workers. H-1B is a visa category in the United States of America under the INA, section 101(a)(15)(H) which allows U.S. employers to employ foreign workers. The first step employer must take to hire a foreign worker is to file the Labor Condition Application. In this project, we will analyze the data from the Labor Condition Application.

### Introduction to Dataset

The H-1B Dataset selected for this project contains data from employer’s Labor Condition Application and the case certification determinations processed by the Office of Foreign Labor Certification (OFLC) where the date of the determination was issues on or after October 1, 2016 and on or before June 30, 2017.

The Labor Condition Application (LCA) is a document that a perspective H-1B employer files with U.S. Department of Labor Employment and Training Administration (DOLETA) when it seeks to employ non-immigrant workers at a specific job occupation in an area of intended employment for not more than three years.

### Project Goal

Our goal for this project is to predict the case status of an LCA application submitted by the employer to hire non-immigrant workers under the H-1B visa program. Employer can hire non-immigrant workers only after their LCA petition is approved. The approved LCA petition is then submitted as part of the Petition for a Non-immigrant Worker application for work authorizations for H-1B visa status.

Through this analysis, we want to uncover insights that can help employers understand the process of getting their LCA approved.

### Dataset Information

Source: This dataset is taken from Kaggle data repository. Kaggle have a wide range of datasets that we can use for machine learning applications.

Link: <https://www.kaggle.com/trivedicharmi/h1b-disclosure-dataset>

Number of Attributes: 27

Number of Instances: 528134

### Attribute List

The dataset attributes are shown in below table:

|  |  |
| --- | --- |
| **ATTRIBUTES** | **DESCRIPTION** |
| CASE\_SUBMITTED\_DAY | Date and time the application was submitted. |
| CASE\_SUBMITTED\_MONTH |
| CASE\_SUBMITTED\_YEAR |
| DECISION\_DAY | Date on which the last significant event or decision was recorded by the Chicago National Processing Center |
| DECISION\_MONTH |
| DECISION\_YEAR |
| VISA\_CLASS | Indicates the type of temporary application submitted for processing. R = H-1B; A = E-3 Australian; C = H- 1B1 Chile; S = H-1B1 Singapore. Also, referred to as “Program” in prior years. |
| EMPLOYER\_NAME | Name of employer submitting labor condition application |
| EMPLOYER\_STATE | Contact information of the Employer requesting temporary labor certification. |
| EMPLOYER\_COUNTRY |
| SOC\_NAME | Occupational name associated with the SOC\_CODE |
| NAICS\_CODE | Industry code associated with the employer requesting permanent labor condition, as classified by the North American Industrial Classification System (NAICS) . |
| TOTAL\_WORKERS | Total number of foreign workers requested by the Employer(s). |
| FULL\_TIME\_POSITION | Y = Full Time Position; N = Part Time Position |
| PREVAILING\_WAGE | Prevailing Wage for the job being requested for temporary labor condition. |
| PW\_UNIT\_OF\_PAY | Unit of Pay. Valid values include “Daily (DAI),” “Hourly (HR),” “Bi-weekly (BI),” “Weekly (WK),” “Monthly MTH),” and “Yearly (YR)”. |
| PW\_SOURCE | Variables include "OES", "CBA", "DBA", "SCA" or "Other". |
| PW\_SOURCE\_YEAR | Year the Prevailing Wage Source was Issued |
| PW\_SOURCE\_OTHER | If "Other Wage Source", provide the source of wage. |
| WAGE\_RATE\_OF\_PAY\_FROM | Employer’s proposed wage rate. |
| WAGE\_RATE\_OF\_PAY\_TO | Maximum proposed wage rate. |
| WAGE\_UNIT\_OF\_PAY | Unit of pay. Valid values include “Hour", "Week", "Bi-Weekly", "Month", or "Year". |
| H-1B\_DEPENDENT | Y = Employer is H-1B Dependent; N = Employer is not H-1B Dependent. |
| WILLFUL\_VIOLATOR | Y = Employer has been previously found to be a Willful Violator; N = Employer has not been considered a Willful Violator |
| WORKSITE\_STATE | State information of the foreign worker's intended area of employment. |
| WORKSITE\_POSTAL\_CODE | Zip Code information of the foreign worker's intended area of employment. |
| CASE\_STATUS\* | Status associated with the last significant event or decision. Valid values include “Certified,” “Certified- Withdrawn,” Denied,” and “Withdrawn”. |

### Target Attribute

Our target attribute for this dataset is ‘CASE\_STATUS’. There are 4 categories for case status as given below:

1. Certified: Case approved by OFLC
2. CertifiedWithdrawn: Case withdrawn by employer post approval from OFLC
3. Withdrawn: Case withdrawn by employer
4. Denied: Case denied by OFLC.

# Data Cleaning

Cleaning the dataset is critical to avoid errors when running machine learning algorithms. Our dataset was processed to detect, correct or remove any corrupt or inaccurate records. Various measures were considered while cleaning and correcting the data to make it suitable for our machine learning algorithms and to achieve high prediction results.

## Data Filtering

Our dataset originally consists of data for Case submitted years 2015, 2016 and 2017. Although, it is fairly sized data for analysis, but due to my current hardware restrictions, I decided to filter the dataset for Case submitted Year 2016 only. The filtered dataset has following characteristics:

Filter applied: CASE\_SUBMITTED\_YEAR =2016

New Data Instances: 100161

New Attributes: 27 (unaffected)

## Column Rename

Only one column was renamed. ‘H-1B\_DEPENDENT’ was renamed to ‘h1b\_dependent’ and all other column names were changed from upper case to lower case for easy reference in the project.

The datatype for ‘worksite\_postal\_code’ was corrected to float.

## Checking for Data Correctness

Several columns were checked for data correctness and possible outliers after observing their statistical description.

The below attrubutes were corrected for outliers

|  |  |
| --- | --- |
| Attribute | Corrections/Imputations |
| Prevailing\_wage | Right skewed distribution  0 – incorrect value. Imputed based on median wagevalues  >250K – outliers. Impute median value based on worksite\_state and soc\_name. |
| Pw\_source\_year | 1 – incorrect value. Impute most occurring year based on prevailing\_wage, pw\_unit\_of\_pay and pw\_source |
| wage\_rate\_of\_pay\_from | Highly right skewed distribution  0 – incorrect value. Impute value of prevailing\_wage.  >400000 – outliers. Impute larger value from prevailing\_wage or median based on worksite\_state and soc\_name. |
| employer\_country | CANADA – incorrect value. Impute value for employer\_country and employer\_state based on other application records from same employer but from United states. |
| Employer\_name | NaN – delete the entries with null values. |
| Pw\_unit\_of\_pay | NaN – Impute values based on wage\_unit\_of\_pay. |
| Pw\_source | NaN – Impute values based on most common values of this attribute |
| Pw\_source\_year | NaN – Impute values based on most common values of this attribute |
| H1b\_dependent | NaN - Impute values if the same employer have any valid entry for this field. |
| Willful\_violator | NaN - Impute values if the same employer have any valid entry for this field. |

Drop all remaining null values from the dataset. The final clean dataset have a below characteristics:

Number of attributes: 25

Number of instances: 98289

Case Status value counts

CERTIFIED: 79974

CERTIFIEDWITHDRAWN:14309

WITHDRAWN: 2787

DENIED: 1219

## Data Transformations:

The categorical features were either label encoded using sklearn.preprocessing.LabelEncoder() or dummy encoded using pandas.get\_dummies(). The list of features is as below:

LabelEncode features: ['employer\_name','employer\_state','soc\_name','pw\_source\_other','worksite\_state']

DummyEncoded features: ['visa\_class','full\_time\_position''pw\_unit\_of\_pay','pw\_source','wage\_unit\_of\_pay','h1b\_dependent','willful\_violator']

The Target variable was label encoded with below labels:

'CERTIFIED' = 0

'CERTIFIEDWITHDRAWN' = 1

'DENIED' = 2

'WITHDRAWN' = 3

Since we are dealing with a multiclass classification problem, we will have to binarize our target variable with the help of label\_binarize() method from sklearn.preprocessing library. This will be explained later.

### Train Test Split

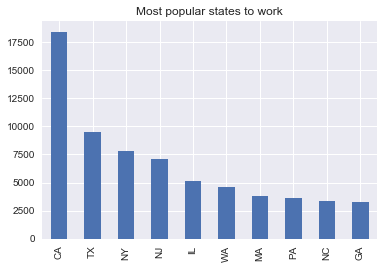
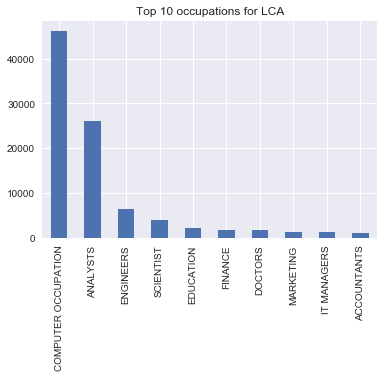
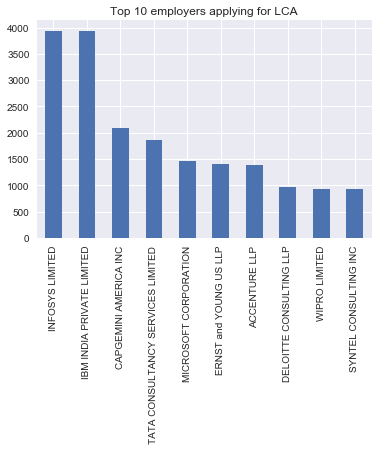
The data was split into Train and Test sets using sklearn.model\_selection.train\_test\_split method with 80/20 ratio. The test data will be used as validation data for our final model performance evaluation.

### Data Scaling

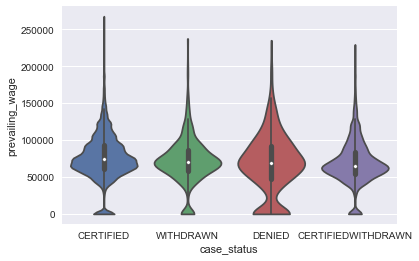
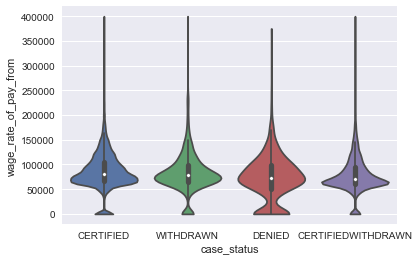
Data scaling is done on Train and Test sets separately using MinMaxScaler method from sklearn.preprocessing.

# Exploratory Analysis

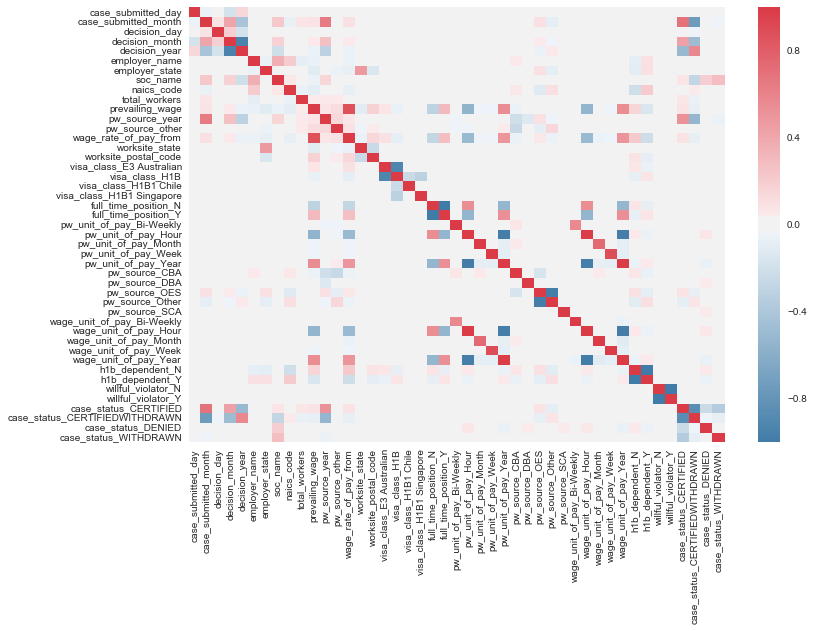
This is the stage where we plot all the data in as many ways as possible. These charts are for internal use, so we don’t bother much to make them pretty. Our goal is to get insights from the data using data visualizations.



Violin plots of wage rate of pay from. Distribution of prevailing wage



Heatmap to show correlation of variables with others.



# Data Analysis

We will use below models to classify the target variable.

- Naive Bayes

- Linear Regression

- Decision Tree

- Random Forest Classifier

- Gradient Boosting Classifier

- Support Vector Machine

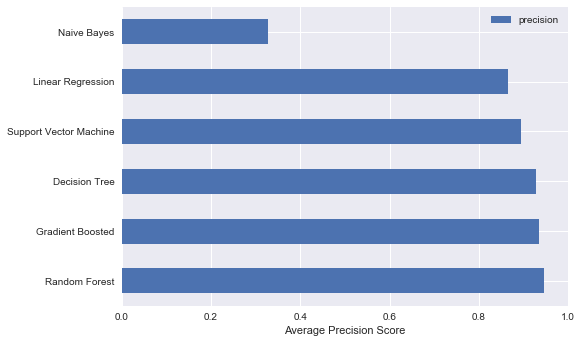
We should note that here we are addressing a multiclass classification problem. Out of above listed models only SVM have built-in capability to handle multiclass classifications. Other than that, all other models works best for binary classification problem and will not produce good results for multiclass problem. We used advanced multiclass classification technique known as One-Vs-The-Rest Classification (also known as OneVsAll Classification) where only one classifier per class is fitted. For each classifier, the class is fitted against all the other classes. In addition to its computational efficiency (only n\_classes classifiers are needed), one advantage of this approach is its interpretability. Since each class is represented by one and only one classifier, it is possible to gain knowledge about the class by inspecting its corresponding classifier.

The Naïve Bayes model was run to determine the baseline performance and then other models were run to select the best model for hyper-parameter tuning.

## Model Comparison

The models were compared using the average precision score from sklearn.metrics library. This method compares the classification problems using F1-score metric. Further, the ‘weighted-precision’ metric from F1-score was used to compare the precision of each class of target variable. This metric calculates the weighted average of F1-scores for each class of the target variable. For more information about average\_precision\_score, read [here](http://scikit-learn.org/stable/modules/generated/sklearn.metrics.average_precision_score.html).

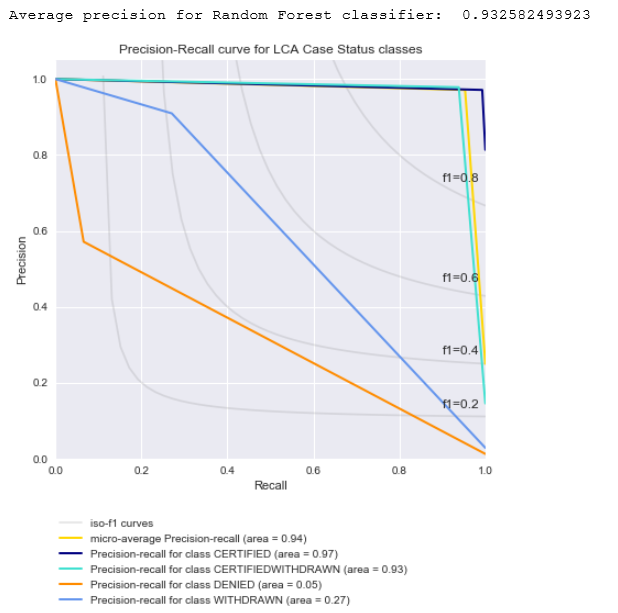
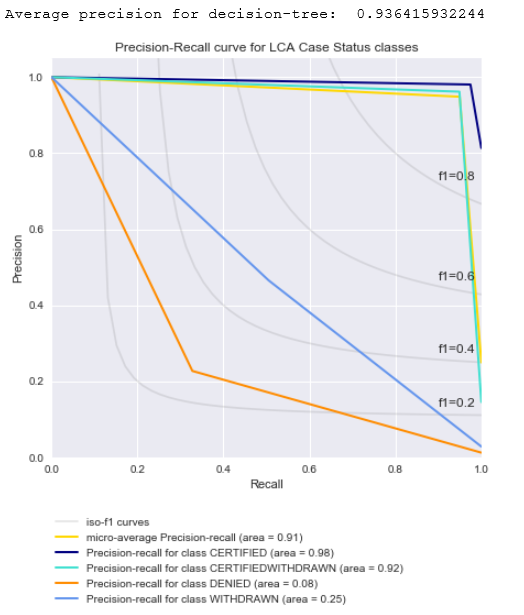
## Model Performance

The below bar-chart displays the performance of the comparison models. All these performances are calculated based on the average\_precision\_score metric. The models were run after fusing the OneVsTheRest classifier with the models.

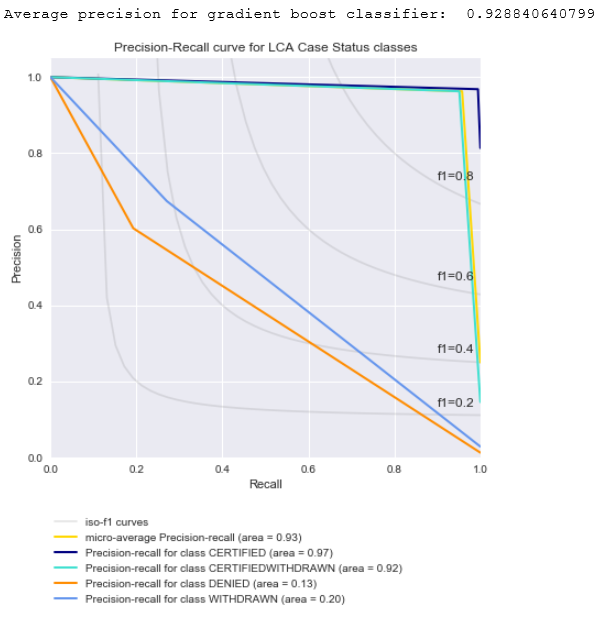
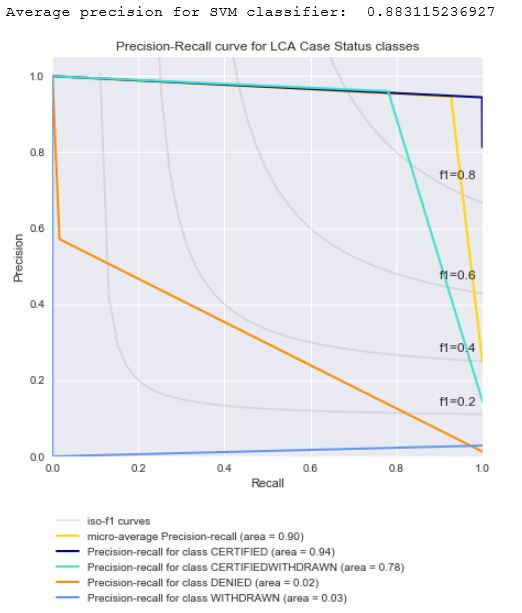
The model precision is calculated on training data after creating 5 fold StratifiedKFolds using sklearn.model\_selection library. The StratifiedKFold method is useful in creating folds that are made by preserving the percentage of samples for each class.

We can see from the figure that the Naïve Bayes performed worst of all the models and Random Forest, Decision Tree and Gradient Boosted classifiers performed similar with just 1% variance. The performance of the models can further be checked to classifying the different classes of target variable. These differences were plotted using precision-recall curve plot. Below are the results of these plots:

Random Forest Classifier Decision Tree Classifier

Gradient Boosted Classifier SVM Classifier

From the above precision-recall curves, we can see that Gradient Boosted Classifier performed well in comparing all the classes of case\_status as compared to other models.

### Hyper Parameterization of Gradient Boosting Classifier

The selected model i.e. gradient boosting classifier is hyper-parameterized using GridSearchCV from sklearn.model\_selection library to explore different combinations of parameters to see if the accuracy our model is further improved.

The best parameters were selected to create the final model and this model was tested on our validation set and the precision-recall curve is plotted to check the improved results.

# Summary

In this analysis, our main goal was to be able to predict the case status of LCA applications that the employers submit to OFLC. We were dealing with the multiclass classification problem trying to predict the case status of four classes: [Certified, Certified-Withdrawn, Withdrawn, Denied]. We witnessed that our dataset was biased based on case status classes. We had more records for Certified, Certified-withdrawn classes and very few records for withdrawn and denied classes. We tried our best to preserve and correct the corrupt data as per our knowledge. As a result of this, our model accuracy came pretty high at 93-94%. These are good accuracy scores. However, our model was able to predict two most occurring classes really well and the other less frequent classes were not classified that well. We tried to improve our results by running the hyper-parameterization but that helped only little. At the end, we can conclude that our model can still be improved by trying even more rigorous hyper-parameterization. If this does not help then we can try adding more records of the less frequent classes to make our data unbiased.

### Future Works

In this project, I was not able to implement the Feature selection to my final model. I need to research more on the feature selection process for multiclass classification problems. I have knowledge until getting the most important features for each class of target attribute but finding out which feature is which, seems another task to learn out of this project. The features are listed based on the most important ones for each class instead of the target variable.

Also, the hyper-parameterization could not be performed for all the available classes, since it took a great amount of time just to perform it for one class of variable. This can be overcome using the cloud solutions like AWS, Google Cloud, Azure etc and is definitely an area to learn more.

# Appendix

The python notebook for this project is attached:



# References

Below is the list of references used for this project:

<https://github.com/averma10/Machine-learning-Projects-walkthrough>

<https://github.com/rhiever/Data-Analysis-and-Machine-Learning-Projects>

<http://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html#sphx-glr-auto-examples-model-selection-plot-precision-recall-py>

<http://scikit-learn.org/stable/modules/generated/sklearn.multiclass.OneVsRestClassifier.html#sklearn.multiclass.OneVsRestClassifier>

<http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html>