H-1B Acceptance Prediction Over Imbalanced Data

Junting Wang
jwang852@wisc.edu

Xi He
xhe93@wisc.edu

Xinyi Yu
xyu286@wisc.edu

Abstract

This project focuses on the H-1B acceptance prediction based on both applicants' and job-related information in 2018. The goal is to understand the important factors that can influence the outcomes of H-1B applications using Machine Learning algorithms. The target variable in this project is the application results: Certified or Denied. In the original data set, we first remove the variables whose majority values are missing and those who are relatively irrelevant by intuition. Due to the imbalanced distribution of these two classes, we used Undersampling and Combination of Oversampling and Undersampling methods to get a relatively balanced data set. Based on these balanced data sets, we apply k-Nearest Neighbors (kNN), Logistic Regression, Random Forest, and XGBoost to compare the AUC results of predictions and later conduct hyperparameter tuning for model improvements. From the final results, we find that these four algorithms roughly provide fair good results, while the XGBoost has the best performance in terms of AUC value. We further notice that the naics_code and prevailing_wage are the two most important factors that can influence the outcome of H-1B applications.

Keywords: H-1B, Feature selection, Imbalanced Data, Random Forest, XGboost

1. Introduction

As one of the countries which absorb the most working immigrants in the world, the United States issues working visas(H-1B visas) to nearly one million foreigners every year and that number will continue to grow slowly because the companies in the United States have a high demand on high-tech workers.[3] After foreigners apply for the H-1B visas, the United States Department of Labor (USDL) will know various information of these applicants, such as wages, job titles, the cities of the workplace, and many other information of applicants. Therefore, the USDL need to determine which factors should be used to qualify the candidates and which factors are relatively more important when evaluating various candidates. While it is difficult to quantify the economic benefits of these work visa holders, there is no doubt that they are crucial to the United States' efforts to globalize its economy and to strengthen its position in high-tech industries. For international students, employment opportunities are limited. In other words, it is difficult for international students to find a job, and it is even more challenging to find a job that is willing to help them apply for a work visa. Therefore, the goal of this project is to build a machine learning model to predict the outcome (certified or denied) of H-1B visas based on the candidates' information, and thus providing applicants some information about certified criteria.

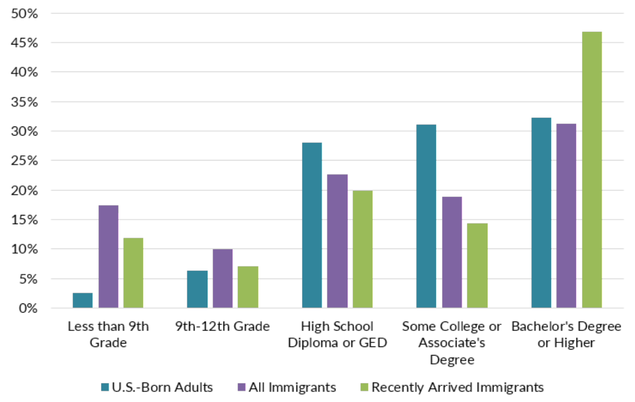

Figure 1. Educational Attainment of U.S.-Born, Immigrant, and Recently Arrived Immigrant Residents, 2017

A data set of the 2018 H-1B petition status has been picked up from kaggle.[1] Here CASE STATUS is the dependent variable which we are aiming to predict and the other 51 columns are the information collecting from each applicants, which include employer name, employer sites, prevailing wage, and job title, etc. There are over 650,000 examples in the original dataset. To find out the best model for prediction, we compared several machine learning algorithms, including kNN, Logistic Regression, Random Forest and XGBoost. Our project mainly consists of three parts. We first conduct data manipulation to deal with missing values, dimensionality reduction, and imbalanced distribution. Based on the balanced datasets, we further fit models with each algorithm to get the results. We finally implement model evaluation to compare each model and get the best one. Meanwhile, we also find out the most important features that can influence the result of H-1B application, thus helping candidates better understand the criteria for H-1B visa. Detailed processes and explanations are presented in the following sections.

2. Related Work

Many of the international students and professionals aspiring to work in the U.S. are waiting for the start of the upcoming H-1B Visa season to file their petitions with USCIS and check out their luck. With the increasing number of international students and professionals in the U.S., it can be of great significance to understand the criteria for H-1B visa selection. Based on what we have found, although there are abundant resources for H-1B analyses from the economics aspect in the U.S. [6], the historical evolution [8], and the relationship with native workers [5], it seems not many of these analyses measure the effect of candidates' work background on the H-1B applications. Therefore, it can be meaningful to provide this almost new perspective of H-1B visa criteria and hopefully can help candidates better understand and prepare for their applications.

Although there are not many resources available for the H-1B acceptance prediction, our project can be viewed as a typical machine learning project about a classification problem. And there have been a lot of algorithms we have learned in class, such as kNN, decision tree, and logistic regression. They are powerful to deal with classification problem. However, one thing that makes our project different from a typical coursework is the imbalanced distribution in our dataset, which will be explained in detail later. Because of this, we cannot simply apply those algorithms for model fitting and test, which can be greatly biased. In this way, following the suggestions from Prof. Sebastian Raschka, we find out several ways to solve such imbalanced problem from the `imbalanced-learn`, from which we can generate a relatively balanced dataset from the original one and further split it as our training and test sets.

3. Proposed Method

In this project, we have tried four different algorithms to predict the outcome of H-1B applications. To begin with, we use a simple kNN model as a benchmark. Since the dependent variable is a two-class feature, we try the Logistic Regression method as a comparison with kNN model. We further apply a more complex Random Forest algorithm and later train the XGBoost to check the improvements of model performance. During this process, we use the ROC Area Under the Curve (AUC) as our performance metric, which can be formulated as

$$AUC = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} 1_{p_i > p_j}}{mn} \tag{1}$$

[4], where i goes over all the m data points of class label 1, and j goes over all the n data points of class label 0. The value of AUC curve can be interpreted as the probability that a positive example is ranked higher than a negative example [9].

Table 1 indicates the standards to evaluate AUC values. As table 1 shows, having 0.5-0.6 AUC value is very poor, having 0.6-0.7 AUC value is fair,and having 0.9-1.0 AUC value is excellent. We use this standard to interpret the model performance later.

AUC Value	Model Performance
0.5-0.6	Poor
0.6-0.7	Fair
0.7-0.8	Good
0.8-0.9	Very Good
0.9-1.0	Excellent

Table 1. AUC Value Interpretation
Sources: [?]

3.1. k-Nearest Neighbors

We use kNN model as a predictive performance benchmark because kNN model is a lazy, well-studied, and widely used in practice model in classification.

The kNN Algorithm classifies a data point based on the outcome of a 'plurality voting' among its k nearest neighbors. In this project, we use the Euclidean distance function to measure the distance between two data points. The formula of the Euclidean distance function is defined as

$$d(p^{[i]} - p^{[j]}) = \sqrt{\sum_{i=1}^{N} (p_i^{[i]} - p_i^{[j]})^2} \tag{2}$$

where p[i]and p[j] are two different data points in an N-dimensional space. The kNN algorithm will select the k nearest neighbors of the target data point and predict the label of the target data point based on the mode label of the k nearest neighbors.

We apply feature selection to the data before fitting the model because kNN is susceptible to the curse of dimensionality. As the number of features increases, the distance of two data points increases. Therefore, it is hard to determine whether two data points are close to each other as the number of features increases. In our data set, there are 52 columns in the original data set. We perform feature selection to reduce the dimension of kNN model in order to make better prediction.

We perform 5-fold and 10-fold cross validation to do hyperparameter tuning on our balanced data set in the kNN algorithm. If the balance data is small, cross validation will help kNN model to avoid pessimistic bias. Then, we select k which gives us the best performance. Then, the highest accuracy of kNN model will be selected and used as a benchmark for other models.

Here we also apply feature scaling method. Since k-Nearest Neighbors with an Euclidean distance measure is sensitive to magnitudes and hence should be scaled for all features to weigh in equally. We use Min-Max scaling which can bring value between 0 and 1 and help us narrow down the large bound of certain value. Here is the function of Min-Max scaling:

$$X_{sc} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{3}$$

3.2. Binary Logistic Regression

The second model that we use in the project is binary logistic regression because logistic regression is a common and simply algorithms and can make good prediction in some situations. We also use binary logistic regression as another micro-benchmark to weight the importance of features. Binary logistic regression make prediction based on Sigmoid Function, which is defined as

$$y = \frac{1}{1 + e^{-z}} \tag{4}$$

where y is the output between 0 and 1, z is the function's input. e is the natural log's base. If y is greater than 0.5, the binary logistic regression will classify the target data point as 1. If y is less than 0.5, the binary logistic regression will classify the target data point as 0.

Similar to kNN, we perform feature selection selection before fitting the model in order to avoid overfitting. Then, we compare the result of logistic regression with the result of kNN.

3.3. Random Forest

Random forest is currently a popular supervised learning algorithm that can be used for both classification and regression. It can be viewed as the ensemble of decision trees with random feature subsets, but random forest does not suffer the overfitting problem, which is one of its major advantages over decision tree. From the lecture, one of the reasons that random forest can provide us a fairly good performance is that its generalization error can be bounded from above, which is formulated as

$$PE \leq \frac{\bar{\rho}(1-s^2)}{s^2} \tag{5}$$

where $\bar{\rho}$ refers to the average value of correlation of trees for random forest, and s represents the strength of trees [7].

Here, we apply the random forest algorithm to our balanced data set with combination method and undersampling method, respectively. Based on the benchmark performance after randomly fitting a random forest model, we then apply cross validation and conduct hyperparameter tuning to further improve the model performance.

3.4. XGBoost

XGBoost is short term for "Extreme Gradient Boosting". Here we use the training data (with multiple features) x(i) to predict a target variable y(i).It is an implementation of gradient boosted decision trees designed for speed and performance.

Boosting is an ensemble method that seeks to create a strong classifier (model) based on "weak" classifiers. By adding models on top of each other iteratively, the errors of the previous model are corrected by the next predictor, until the training data is accurately predicted or reproduced by the model. Here, we apply XGBoost method to the balanced data set.

4. Experiments

4.1. Dataset

Our dataset is originally collected from Kaggle [1], which includes the H-1B candidates' information in 2018 and consists of 52 columns and over 650,000 rows. Each column stands for one variable, and each row corresponds to one candidate for H-1B application. In the original dataset, our target variable (CASE STATUS) has four values: certified, certified but withdrawn, withdrawn, and denied. Since our primary goal is to find the important factors that can help predict the acceptance or deny of the H-1B application, we disregard the samples with withdrawn value, from which we cannot tell whether the applications are certified or not, and interpret the value certified but withdrawn as simply certified.

We further clean our dataset by dropping columns whose majority values are missing and we cannot find additional resources to impute those missing values. Moreover, since there variables whose information can be overlapping, such as the city and state of the candidates' working sites, we choose the one providing more specific information. For example, we keep the working city and drop the working state in the dataset. We further remove irrelevant variables by intuition, such as case number, agent attorney name, and employer phone number. Since our dataset contains many categorical variables, we encode those variables and further convert dates to numbers since the earliest date in the dataset as preparation for model fitting in the next step.

4.2. Feature Selection

Although we have dropped 16 columns during data cleaning, there are still 36 variables remaining. To improve our efficiency, we perform the feature selection to reduce dimensions using logistic regression, which helps us to identify which factors are more important. Based on these results, we also apply univariate feature selection method from the function `SelectKBest` in `scikit-learn`. Here we have already converted our dependent variable which is the `CASE_STATUS` into a binary class, and since univariate feature selection is targeted at solving problems with only one dependent variable, thus this method fits our model well. We pick `chi2` as a method of our score_function and try to select 10 to 20 features among the remaining variables. After comparison, when k = 15 and the other parameters remain same all of the model will have the best performance with the highest AUC value.

4.3. Balanced Dataset

As we take a look at the basic distribution of our target variable, we find that it is a very imbalanced dataset. From the figure 2 that compares the number of samples classified as class 0 and class 1, we can find that there are about 93 % of candidates for H-1B are certified, while only about 7 % of them are denied. In other words, even though our models predict that all candidates should be accepted, the accuracy of classification would still be about 93 %. Therefore, we use both combination of over- and under- sampling method and undersampling method to balance out the data set. The combination method reduces the number of majority class samples (here, class 1) and increases the number of minor class samples (here, class 0), which provide us around 60,000 samples for each class. And the undersampling method reduces the number of majority samples, which reduces the number of each class to about 8,500. Here, we use the `SMOTEENN` function to combine over-sampling and undersampling, and use the `RandomUnderSampler` for undersampling. Both combination method and undersampling method are not perfect, and can bring some problems. For example, the combination method may repeat the same reject examples in the balanced dataset, which can cause overfitting problem. On the other hand, implementation of undersampling method may remove important examples from the dataset for model fitting and test, which can lead to reduced information. Detailed discussion and the specific issues in this project will be stated in the following sections.

4.4. k-Nearest Neighbors

We used the data set with feature selection, which contains 15 best variables, in kNN algorithm.

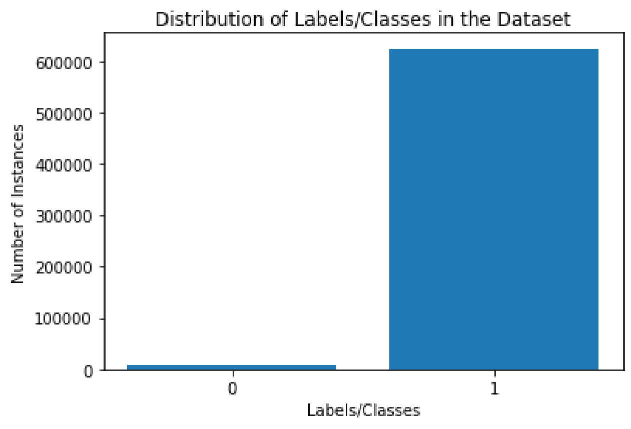

Figure 2. Number of Samples Labeled as Class 0 vs. Class 1

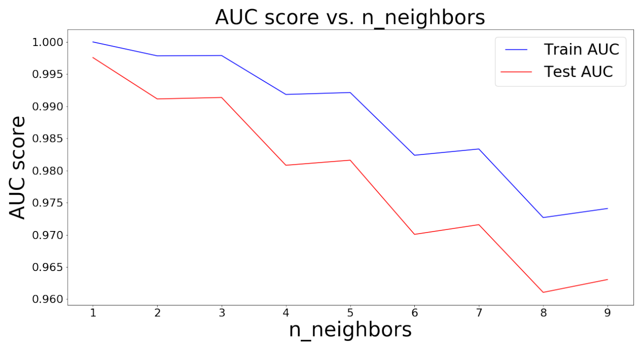

Figure 3. combination sampling method: AUC score vs. number of neighbors

We perform k-Nearest Neighbors on both combination sampling and undersampling methods. However, because of extremely high accuracy in the combination data set, as indicated in the figure 3, the combination data set can result in overfitting problems. This is because the differences between the number of rejected data points and the number of accepted data points. Combination sampling can cause many repeated rejected data points, which can cause overfitting. One solution to address this problem is to do cross validating before apply combination sampling method because the minority class in validation set would not be repeated in cross validation. However, this method is very time-consuming, it seems not necessary to further conduct hyperparameter tuning on the kNN model.

We tune the hyperparameter on our undersampling data set by using 10-fold and 5-fold cross validation. The hyperparameter that we tune in the kNN model is `k`: the number of nearest data points. We select k from 1 to 10. And the k does not make huge difference in performance. The k with the highest accuracy is 5 in both 5-fold and 10-fold cross validation. Thereofore, we select 5 as the value for hyperparameter `k`.

Here, we also try feature scaling by using `MinMaxScaler` in `sklearn.preprocessing`, we want to have a better interpretation on some features which may lead to a different prediction before and after scaling. Since only few features we picked are numerical data, we only apply the method on these features.

4.5. Binary Logistic Regression

Similar to kNN algorithm, we apply feature selection, with 15 best features, in logistic regression algorithm to avoid overfitting. Another important function for Binary Logistic Regression is that it can provide us with an intuitive image on what specific variables will have greater weight. This is a basic try on providing some useful suggestions for H-1B applicants to help them raise the chance of accepting.

For logistic Regression, we also use combination of over-sampling and undersampling and using undersampling.

4.6. Random Forest

Similarly as before, we construct random forest models on the dataset using combination of over-sampling and undersampling as well as the dataset using undersampling. However, based on the relatively high accuracy in the combination dataset, as indicated in the table, it seems not necessary to further conduct hyperparameter tuning on the random forest model. And due to the large size of combination dataset, we do not need to implement cross validation. On the other hand, we tune the hyperparameters of the model in the dataset with undersampling. Due to the time consideration, we only focus on the three parameters: `n_estimator`, `max_depth`, and `max_features`, which are typical hyperparameters in a random forest model. Since it is time consuming to implement grid search using `GridSearchCV`, we choose to apply `RandomizedSearchCV` with 5-fold and 10-fold cross validation. The results can be found in the table 1 below.

4.7. XGBoost

We build the XGBoost model by using the classification function `XGBClassifier` in the model, we create a XGBClassifier with `max_features` as `sqrtmax_features` which is the number of features to consider when looking for the best split. Here is an quick example, let's say we have `n_features` which is 100, then the `max_features` would be 10. Next we using GridSearchCV to tune hyperparameters, it is a function import from scikit-learn. Again due to the time consideration, we only focus on two parameters: `n_estimators` and `learning_rate`.Similar to the previous kNN model and Random Forest model, we also use Cross Validation method to pick a more generalized and robust model which will have the best performance.

4.8. Software

- Python
    - ML: Scikit-Learn, Imbalanced-Learn
    - Data: NumPy, Pandas
    - Environment: JupyterLab
- Excel

Models	undersampling	Combination
kNN	0.646	0.949
kNN (feature scaling)	0.50	0.50
kNN (CV=5)	0.702	0.987
kNN (CV=10)	**0.707**	**0.989**
Logistic Regression	0.623	0.642
RF	0.751	0.996
RF (CV=5)	0.756	0.972
RF (CV=10)	0.757	0.973
RF (After tuning)	**0.80**	**0.82**
XGBoost	0.816	0.943
XGBoost (CV=5)	0.821	0.968
XGBoost (CV=10)	**0.901**	**0.973**

Table 2. AUC Values for Each Model

Figure 4. Under-sampling: AUC score for kNN

5. Results and Discussion

The results of each model have been listed in the table 2, where RF refers to random forest, CV=5(10) stands for 5(10)-fold cross validation. The models in bold fonts are the best result for each type of model after tuning. From this table, we can see that the combination of over- and under- sampling data set is particularly prominent in kNN and XGBoost models, where the AUC value is infinitely close to 1. However, we think this may be a result of overfitting since in our original data set the difference between two classes are so large and oversampling method will inevitably bring us some new data with same value, and this can leads to a high AUC value when using combination data set. In order to avoid this problem, we decide to use the result based on undersampling data set as our final result. In general, the AUC values for all these four algorithms are fairly good, as suggested by the criteria in table 1. In addition, we can see from the results of other models, cross validation with k =10 gives a better AUC value compared with k = 5, which corresponds to what we have learned in class, since the 10-fold cross validation is typically an optimal choice. The hyperparmeter tuning process can give the model a decent improvement while the XGBoost provides the best result among these models in terms of AUC score.

5.1. k-nearest neighbors

The kNN model provides an extremely high AUC value 0.949, 0.987, and 0.989 for the combination dataset, this number is very huge. k-nearest neighbors is very sensitive to overfitting. This extreme vale is due to the overfitting.

For undersampling dataset, the original kNN model without cross validation has an AUC value about 0.646, which is not a fair AUC score. 5-fold CV and 10-fold CV improve the AUC value to 0.702 and 0.707 because cross validation can reduce the pessimistic bias caused by undersampling method.

The result for feature scaling is not as good as we expected. The AUC value is exactly 0.5 which means the features after scaling can not tell us anything about the `CASE_ STATUES`. It may resulted by the large range of the numerical data and since we have already convert all the numerical data into same unit (eg.`prevailing_wage`, the scaling method does not improve the performance of the model.

5.2. Binary Logistic Regression

On both undersampling and combination sampling datasets, the binary logistic regression model does not perform very well. As the table 2 shows, the binary logistic regression's AUC score is 0.623 for undersampling dataset and 0.642 for combination dataset. Different from the results for kNN models, the AUC scores for logistic regression on both combination and undersampling datasets are relatively close to each other, although the AUC score for combination dataset is slightly higher than that of undersampling set. Based on the table 1, although the results for logistic regression are not as good as kNN or the random forest and XGBoost below, we can still interpret them as fair enough.

5.3. Random Forest

Similar to the kNN, the random forest model provides an extremely high AUC value 0.996 for the combination dataset. This is mostly likely due to the same reason as kNN, which states that there can be abundant repeated rejected data points in the combination dataset, thus increasing the risk for overfitting problem. For the undersampling dataset, the random forest algorithm works well, providing about 0.751 AUC value for initial fitting. The AUC values after cross validation have been improved. From the table 2, we can see the 5-fold cross validation provides the AUC value with 0.756 for undersampling dataset and AUC value with 0.972 for combination dataset. The results after 10-fold cross validation slightly increase, and we get AUC equals 0.757 for undersampling set and 0.973 AUC score for combination set. After hyperparameter tuning, we get the highest AUC scores 0.80 and 0.82 for undersampling and combination datasets, respectively. From the standards shown in the table 1, our results for random forest are good.

5.4. XGBoost

Not surprisingly, XGBoost model perform pretty well on both undersampling data set with AUC score 0.816 and combination data set with AUC score 0.943. Again we believe that the high AUC value on combination data set is resulted by the exact same reject case repeats in our data set, thus leadning to the overfitting problem. Same with other models after applying cross validation method on undersampling data set, the model performance based on undersampling set has been improved from 0.816 to 0.821 with 5-fold cross validation and to 0.901 with 10-fold cross validation. The parameters we choose in the model are `n_estimators = 100` and `learning_rate = 0.5`.

6. Conclusions

In this project, we implement several machine learning algorithms: kNN, logistic regression, random forest, and XGBoost, to predict the outcomes of H-1B applications in the U.S. One of the main issue in our project is the imbalanced dataset. To solve this, we try the undersampling and combincation of over-sampling and undersampling methods to get a more balanced dataset. Although the results for these two datasets seem more reasonable than the original one, there is still the overfitting problem, especially for kNN and random forest. One possible reason is that the over-sampling in the combination method can generate many repeated rejected data points, thus increasing the risk for overfitting. We further conduct hyperparameter tuning and cross validation on the undersampling dataset to improve the model performance. All models perform well, but the XGBoost provide the highest AUV value. In this way, we believe XGBoost can be the suitable method to predict the H-1B applications.

Besides the suitable way for prediction, we also find that the variables `naics_code` and `prevailing_wage` seem to be the two most important factors during feature selection. The `naics_code` refers to the North American Industry Classification System codes that classify the business by industry [**?**]. And the `prevailing_wage` is simply the hourly wage, but we change it to yearly standard to make sure every record is on the same scale. This result shows us that the industry of candidates' work and their wage can be influential for their H-1B application. This can be reasonable, since the industries that hire H-1B workers are mainly technology and finance [2]. Surprisingly, the average wage of H-1B employees are not as high as that of technology industry [2], which implies that the overall work situation for international employees can be more challenging. In this way, our project provides information for H-1B candidates based on their perspectives, which is not so common compared to the majority of current analysis for H-1B applications from the economics aspect.

However, there is still room for improvements.One of the main challenges in our project is the running time. Due to the large dataset with high-dimensions, it is time consuming to conduct hyperparameter tuning, especially with cross validation. To solve this problem, we could apply the idea of clustering method, which means we could randomly select data points both with or without replacement from the balanced datasets to consist a new but smaller dataset for training and testing. Hopefullt this can reduce the running time for both model fitting and hyperparameter tuning, but still provide a fairly good performance. Meanwhile, since we select the 15 best features using logistic regression, such selection may not include additional information that can be of importance in our models. In this way, it time permits, it can be better to try kNN, logistic regression, random forest, and XGBoost on the dataset including more features.

7. Acknowledgements

This project receives much assistance and advice from Prof. Sebastian Raschka, including presentation, ways to improve model performance, and the comparison among different algorithms. We hereby would like to thank Prof. Raschka for his time and suggestions, which are important during this project and provide us many useful thoughts regarding the applications of machine learning algorithms.

8. Contributions

- Junting Wang: Junting mainly collected the dataset from Kaggle and worked with Xi and Xinyi together to clean the dataset. She wrote code for combination method and undersampling method to get balanced data. She applied random forest algorithm for both feature selection and model fitting, and futher implemented hyperparameter tuning with randomized search and grid search. And she collaborated with Xi and Xinyi on slides and report together.
- Xi He: Xi also worked with Junting and Xinyi on data cleaning, slides and report together. For data cleaning, she encoded category variables to numbers. She applied logistic regression for feature selection, and further used kNN and logistic regression for model fitting. She also implemented hyperparameter tuning and wrote code for AUC values and plots.
- Xinyi Yu: Xinyi worked with Junting and Xi together on data cleaning, slides and report. Xinyi did feature selection and further applied XGBoost for model fitting and hyperparameter tuning. He also ran cross validation method for both three models and tried feature scaling on kNN model. He collected and combined the code written by Junting and Xi together, and managed the final results for each model.

References

[1] Green card h1b (2014-2018). `https://www.kaggle.com/jonamjar/green-card-h1b-20142018/version/1#PERM_Disclosure_Data_FY2018_Q4_EOY.xlsx`.

[2] Average h-1b salary: How does it compare to yours?, 2019. Accessed: 2019-12-18.

[3] P. Martin. Immigration to the u nited s tates. *The Wiley Blackwell Encyclopedia of Race, Ethnicity, and Nationalism*, pages 1–10, 2015.

[4] M. J. Pencina, R. B. D' Agostino Sr, R. B. D' Agostino Jr, and R. S. Vasan. Evaluating the added predictive ability of a new marker: From area under the roc curve to reclassification and beyond. *Statistics in Medicine*, 27(2):157–172.

[5] G. Peri, K. Shih, and C. Sparber. Foreign and native skilled workers: What can we learn from h-1b lotteries? Working Paper 21175, National Bureau of Economic Research, May 2015.

[6] G. Peri, K. Shih, and C. Sparber. Stem workers, h-1b visas, and productivity in us cities. *Journal of Labor Economics*, 33(S1):S225–S255, 2015.

[7] S. Raschka. 07-ensembles-notes, Nov. 2019.

[8] M. L. Usdansky and T. J. Espenshade. The h-1b visa debate in historical perspective: The evolution of u.s. policy toward foreign-born workers. *UC San Diego: Center for Comparative Immigration Studies*, 2000.

[9] Wikipedia contributors. Precision and recall — Wikipedia, the free encyclopedia, 2019. [Online; accessed 22-October-2019].