CSE343: Machine Learning (Monsoon 2020)
FINAL PROJECT REPORT

H1-B Work Visa Approval

Himanshu Raj
2018038
himanshu18038@iiitd.ac.in

Naman Tyagi
2018055
naman18055@iiitd.ac.in

Vishwesh Kumar
2018119
vishwesh18119@iiitd.ac.in

Abstract

With another fresh change in the Immigration policies adopted by USA during the pandemic, a focus has been bought on the state on the H1B visa. In 2019, over 150,000 people applied fresh, and near a million applicants applied for renewal. The immigration process is very closed process, as no details are provided to applicants on which basis the selection / rejection has been made. However, with this dataset release, which has been anonymized, we can now take a deep dive into what their policies revolve around. We want to analyse and see which factors has the US been caring about, and which traits downplay a profile. We also want to learn how well machine learning methods can predict the selection procedure. Using the analysis, one can potentially reduce a lot of risk involved in the application process, both to independent applicants and employers who have funded the visa application. [Github]

1. Introduction

With the volatile nature of US Immigration rules, we have had a great deal of uncertainty as to which applications have a good chance of getting through, and which fail. From an organizations point of view, a lot of effort and financial resources are exhausted in simply the application process for bringing an immigrant labor in. Post the project, with analysis of the data, and model training, we aim to answer the following questions:

1. Which traits can be deemed desirable, and which traits are really a red flag in the US immigrations eyes.

2. Help people understand which profiles have a good chance of getting through the process (risk estimation), show them which traits/features of theirs contribute significantly to their chances.

3. Help organizations predict approval, to help the hiring process, and conserve resources.

2. Literature Review

This paper [1] shows the study of the Search for Skills: Demand for H-1B Immigrant Workers in U.S. Metropolitan Areas: This paper analyzed the h1-b visa requests in 2011.The following were its findings Employer demands have exceeded the number of visas issued. Out of these, 2/3rd of visas was for STEM jobs Almost 250 people on an average request for visas from 160 metropolitan cities. The funds for training are not allocated in proportion to the volume of requests coming in.

This paper [2] shows the study of the Effects of High-Skilled Immigration Policy on Firms: Evidence from Visa Lotteries: The paper analyzed the effect of raising the H1-B visa cap and the visa lottery. It concluded the following: Increasing the cap resulted in moderately more employment. There is an insignificant effect of the cap on R&E credit Median firm profits increased slightly by the cap increment; however median employee earnings decreased. In general, H-1B visa holders crowd out their alternatives and work at lower wages, incrementing the firm's profits, despite no increment in innovation.

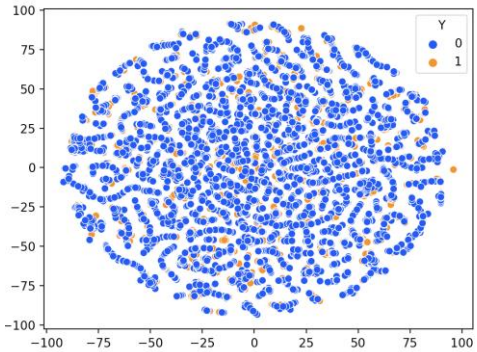
3. Dataset Description

1. Using official statistics, we were able to find the datasets for H1-B approval, and we selected the dataset for the year 2019.
2. There were over 260 features recorded in that, and the target variable was CASE_STATUS. Some of the major attributes included Job title, employer name, wage, employment duration, renewal and so on.
3. The dataset contained a large number of features, a lot of which had missing values.
4. There were relations between attributes (SOC_CODE and JOB_TITLE, NAICS code and JOB TYPE). We needed to identify these relations, and add or drop columns appropriately.
5. Additionally, a lot of the data was categorical (and in text format). We needed to convert this to proper input features for the classification problem.
6. Removing null values: We removed columns which had 50% or more attributes empty. This reduced our columns to 55 from 260. We observed many co Feature Selection:

7. Then, by iterating over the features we had, we researched each feature (from the US website), derived relations between them, and chose the relevant features accordingly. We were down to 16 columns after this.
8. Reducing skewness: Seeing that the data was skewed towards acceptance, we felt sampling the dataset would help reduce the skewness, and give a better fit for negative predictions. We sampled 1,16,753 data points, which reduced the skewness.
9. Feature Processing: Then, we handled binary categorical values, by mapping them to 0 or 1. Using appropriate weights, we changed all wages unit to per annum. We also extracted the net time of employment.

Methods applied for feature extraction/reduction and handle imbalance:

1. Tried One-Hot and Success Rate method to handle categorical values.
2. MRMR and ADASYN: Too computationally expensive.
3. Used SVD for dimensionality reduction (reduced to 30).
4. Applied Normalization and standardization on dataset.



TSNE Plot of our Preprocessed Dataset

4. Model Description

4.1 Naïve Bayes

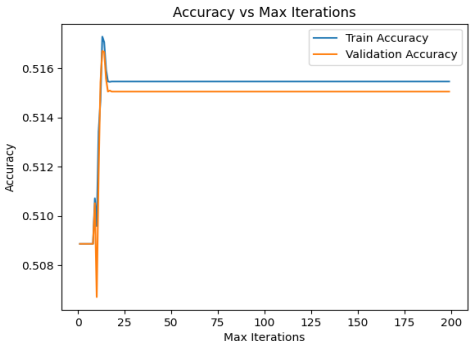
Applied Naive Bayes Classifier here. Made 80-20 train-test splits of dataset. This is with one-hot encoding on categorical columns.

	Predicted: 1	**Predicted: 0**
Actual: 1	244	2652
Actual: 0	180	2820

4.2 Logistic Regression

Applied Logistic Regression on dataset with 60-20-20 train-validation-test split. Used grid search for tuning hyperparameter such as max_iter. Used cross-validation of 4 folds in grid search. Used a graph to analyse the results on different hyperparameter.

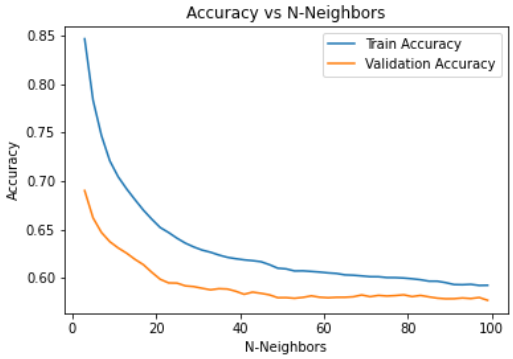
	Predicted: 1	**Predicted: 0**
Actual: 1	266	2690
Actual: 0	153	2847



4.3 KNN

Applied KNN on given dataset. Used 60-20-20 train-validation-test split. Used grid search to tune hyperparameters like n-neighbors. Used graphs to analyse the results obtained on different hyperparameter settings. Optimal n-neighbor came out to be 3.

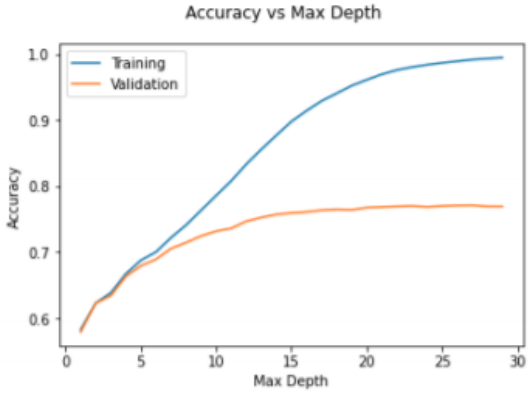
	Predicted: 1	**Predicted: 0**
Actual: 1	2234	662
Actual: 0	1062	1938



4.4 Decision Tree

Applied Decision Tree Classifier on given dataset. Used 60-20-20 train-validation-test split. Used grid search to tune hyperparameters like max-depth of decision tree. Used graphs to analyse the results obtained on different hyperparameter settings. Optimal Max depth value come out to be 27.

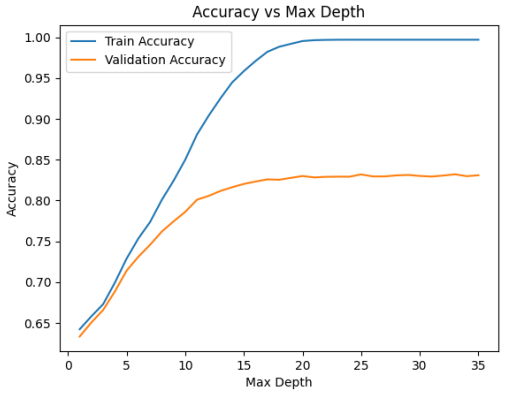
	Predicted: 1	Predicted: 0
Actual: 1	2276	620
Actual: 0	687	2313



4.5 Random Forest

Applied Random Forest Classifier on the given dataset with 60-20-20 train-validation-test split. Used grid search for tuning hyperparameter such as max_depth, n_estimators, max_features, bootstrap, etc. Used cross-validation of 4 folds in grid search. Used graphs to analyse the results on different hyperparameter. Optimal hyperparameters - Max Depth, Number of Trees, Max Features come out to be 26, 350 and 9 respectively.

	Predicted: 1	Predicted: 0
Actual: 1	2441	455
Actual: 0	454	2546



4.6 Support Vector Machine

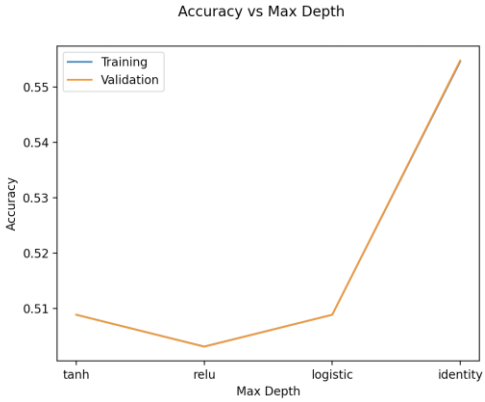
Applied SVM on dataset with 60-20-20 train-validation-test split. Used grid search for tuning hyperparameter such as C, gamma and kernel. Used cross-validation of 4 folds in grid search. Best Hyperparameter are kernel = 'rbf', c = 100 and gamma =0.01.

	Predicted: 1	Predicted: 0
Actual: 1	1812	1111
Actual: 0	660	2313

4.7 Neural Networks

Applied Neural Network on dataset with 60-20-20 train-validation-test split. Used grid search for tuning hyperparameter such as activation function and batch size. Used cross-validation of 4 folds in grid search. Best activation function came out to be identity and batch size of 256.

	Predicted: 1	Predicted: 0
Actual: 1	1131	1765
Actual: 0	670	2330



5. Result Analysis

We have tested different models and optimized it using hyperparameter tuning. We will now analyse the results obtained on different models and we will also analyse the confusion matrix corresponding to each model.

	Train Accuracy	Validation Accuracy	Test Accuracy
Gaussian Naïve Bayes	0.5172	-	0.5196
Logistic Regression	0.5154	0.5150	0.5178
KNN	0.8456	0.6976	0.7075
Decision Tree	0.9926	0.7756	0.7827

Random Forest	0.9971	0.8313	0.8458
SVM	0.95	0.6917	0.71
Neural Network	0.5545	0.5547	0.5870

	Precision	**Recall**	**F1 Score**
Gaussian Naïve Bayes	0.5754	0.0842	0.1469
Logistic Regression	0.5738	0.0711	0.1265
KNN	0.6777	0.7714	0.7215
Decision Tree	0.7681	0.7859	0.7769
Random Forest	0.8431	0.8428	0.8430
SVM	0.7330	0.6199	0.6717
Neural Network	0.6279	0.3905	0.4815

Linear Models such as Naïve Bayes and Logistic Regression are giving very low accuracy compared to Decision Tree and Random Forest. In linear models, True Positives are very low. The models are not able to classify True samples correctly. Linear models are able to classify True Negatives relatively better.

Using KNN we get around 75% test accuracy which is better than linear models used above. A major thing to notice here is that as the n-neighbor value increases the accuracy decreases. We get the best accuracy on 3 n-neighbors value.

One of the major challenges we have in this dataset is the class imbalance and we handled it using balancing data using up and down sampling. In models like Decision Tree and Random Forest, the classification comes out to be appropriate as the value of positives and negatives are relatively close.

SVM gives a training accuracy of 95% but this accuracy falls to 70% on test and validation sets. The best kernel came out to be RBF. The number of false negative samples are higher and that causes misclassification of true samples. Although it performs better than the linear models but not better than Decision Tree and Random Forest Classifier.

For Neural Network, the accuracy comes out to be 55% for all sets i.e., train, test and validation. Although we expected Neural Network to perform better in this case but it fails due to much larger number of combinations to tune the hyperparameters.

Hence, **Random Forest Classifier** performs the best out of all the models we tried and we get around 84.58% test accuracy using Random Forest Classifier. The precision and recall values show that Random Forest is a relatively balanced model. On top of that, the accuracy is also far superior to other models.

6. Conclusion

By trying various methods and models, following are the conclusion we have drawn

1. Due to simplicity of Linear Models, the models are unable to create a good decision boundary for the classification problem.

2. KNN gives train accuracy of 85% and test accuracy of 70%. This shows that the model trains well on train set but underperforms on unseen data.

3. Decision Trees are able to form the decision boundary better and hence gives a much better accuracy.

4. Ensembles Techniques such as Random Forest improves the accuracy by 6%, due to combining results from numerous single decision tree. Additionally, lower number of features were shown to produce better results after grid search in random forest. The ensemble provides lower variance and higher bias which allows the model to perform better on unseen data (testing).

5. Despite obtaining high training accuracy (95%) SVM fails to generalize on validation and testing set and drops to 70% accuracy.

6. The neural network performs similar to the linear models. It gives an accuracy of 58% on test set. This result is due to limitation of system to tune the hyperparameter appropriately.

7. Team Contribution

All the three members contributed equally to data preprocessing, model training and tuning, analysis and the report.

8. References

[1] The Search for Skills: Demand for H-1B Immigrant Workers in U.S. Metropolitan Areas. Neil G. Ruiz, Jill H. Wilson, and Shyamali Choudhury LINK

[2] The Effects of High-Skilled Immigration Policy on Firms: Evidence from Visa Lotteries. Kirk Doran, Alexander Gelber, and Adam Isen LINK

[3] Likelihood of a Work Visa Approval. Hitesh Vyas, and Siddhartha Prakash LINK

[4] Predicting the Outcome of H-1B Visa Applications. Beliz Gunel, and Onur Cezmi Mutlu LINK