MIS 749 BUSINESS ANALYTICS

PROJECT REPORT

Team: Stats whispers
-Deepika Siriah
-Ketul Patel
-Rashmi Sawant
-Shraddha Masuti

I. Executive Summary

The H-1B visa is an employment-based, non-immigrant visa category for temporary foreign workers in the United States (US). For a foreign worker to apply for H-1B visa, a US employer must offer a job and petition for H-1B visa with the US immigration department. H-1B visa class is very industry specific and many workers and companies rely on this yearly allotment. Currently, U.S Citizenship and Immigration Service (USCIS) has H-1B Lottery visa allotment system and it is completely independent of any matric criteria.

As per U.S Citizenship and Immigration Service (USCIS), H-1B Visa Case status is lottery based. Every year there is H1B CAP of 65,000 in general quota and 20,000 for the people who have completed their Master's studies in US. There is tough competition in applications of H1B, and thus employers face difficulties to determine their chances of employees' visa-status outcome. In Spite of H1B Visa Case status being a lottery based we propose that there is no stratified randomness while allocating H1B visa to the applicants. If there is no stratified randomness present then our objective is to find the attributes that contribute towards the final visa-status result.

Thus, our main objective for the project is:

1. To predict the H-1B Visa case status outcome
2. Identify attribute(s) which has any inference on visa-case status outcome
3. Determine the best predictive model for visa-case status
4. Provide recommendation(s)

The H-1B disclosure data released by the Office of Foreign Labor Certification (OFLC) comprises of large dataset along with the case status. The H-1B visa case status has been

classified into two categories as “Certified” and “Denied”. Additionally, the dataset contains attributes like case submitted date, decision date, employment start and end date, employer information, SOC code, NAICS code, prevailing wage, prevailing wage level, H-1B Dependent, and other relevant information filled in the petition.

The raw dataset from OFLC had 610304 rows with 52 columns and consisted of 4 types of case status i.e ‘CERTIFIED’, ‘DENIED’, ‘WITHDRAWN’ and ‘CERTIFIED WITHDRAWN’. For our objective we proceeded further with the subset of data of 136674 rows and 52 columns. Data preparation included data cleaning initially, through excel based on the decision and removed irrelevant attributes such as Employer telephone number and later in R. After feeding data, we have used R to explore and learn more about the relationship of attributes. After completing pre-processing of data by scaling, reducing number of levels of predictors, converted date columns into days, removing irrelevant/non-predictive values, missing values, duplicates, unique factor levels and highly correlated predictors to the response variable. Subsequently, we have split the data into training set and test set with the proportion of 70-30. As the data is highly imbalanced, we have used balancing techniques – upsample, ROSE, SMOTE, Ovun out of which SMOTE found to be best suitable for us. As model planning and selection, we have selected various classification models and trained models with 10 folds cross-validation. Since there are 71 dimensions, we have used PCA to reduce the dimensions. Unfortunately, this did not help much as we had to take almost all components as significant PCs. After testing all models on test data, Random Forest without PCA was found to be the best model with AUC of 82.05% ,Sensitivity of 88%, Specificity of 59% and Accuracy of 88%.

After successful classification of articles on the basis of perceived popularity, we have tested below hypothesis:

Hypothesis. We would like to determine the H1B visa case status id lottery based or not. If not, is there any relationship between case status and other attributes such as prevailing wages, prevailing wage level, SOC, employment type, etc.

Null hypothesis:

H_0: There is a stratified randomness and hence there is no relationship between the predictors and case status.

Alternative hypothesis:

H_A: There is no stratified randomness and hence there is a relationship between the predictors and case status.

After running various models and analyzing the results it was found that we are rejecting Null Hypothesis as it has been observed that there is a relationship between case status and other predictors. The attributes that are important and impact the decision of H1B visa case status are New Employment, Amendment Petition, SOC (Standard Occupational Classification system), NAICS (North American Industrial Classification System).

Our model will help the employers to predict and increase their chance of visa case status to be certified. Also, employers can understand the attributes that are impacting the decision of H1B visa case status.

However, further study is recommended to use other advanced models such as support vector machine, a neural network to improve accuracy and sensitivity. Additionally columns with very high levels of data like Employer name and job title can be added to get more accurate results.

II. Discovery

The H-1B dataset is released by the US Department of Labor – Office of Foreign Labor Certification (OFLC). The OFLC generates disclosure data that provides useful information that is essential both for internal assessment of program effectiveness and for the other stakeholders like employers, employees.

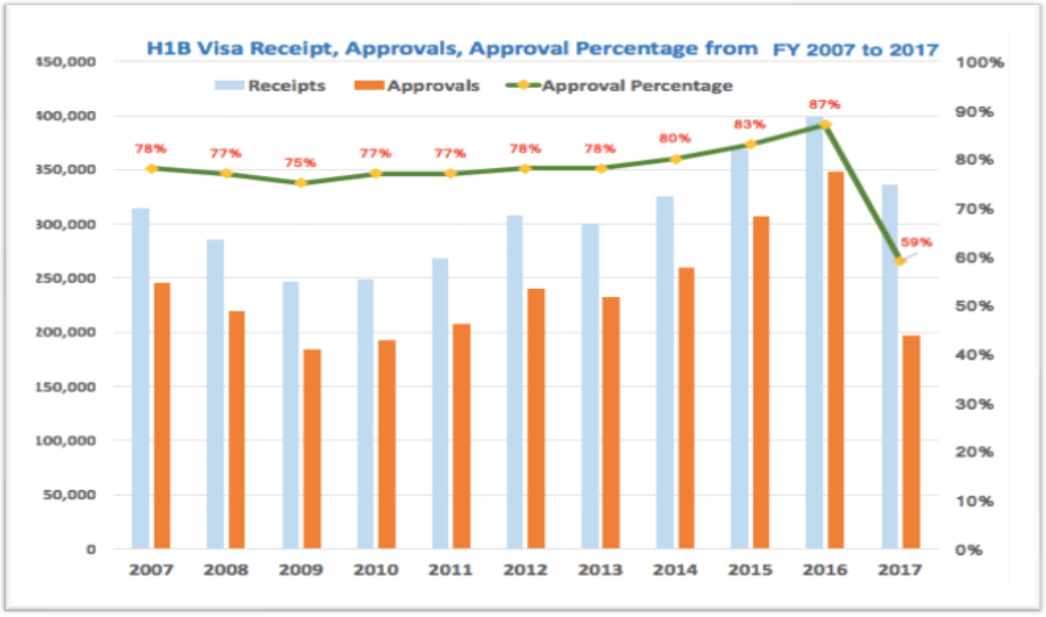


Figure 1: H1B Visa Receipt, Approvals, Approval Percentage from FY 2007 to 2017

In Figure 1, we can see that the allocation of H1B visa has been increased from 2007 to 2016. The highest approval percentage was observed in 2016 with 87% of approval percentage. This percentage was drastically hampered in 2017 due to proposal of changes in rules and

policies of H1B visa. In such scenarios, as the approval percentage is low in 2017 i.e 59%, employers are more concerned about their selection of potential H1B candidates. As their time and investment are on stake, they need a system which would help them to understand the factors influencing the H1B status.

The dataset of H-1B disclosure was collected from their official website for the year 2017.

https://www.foreignlaborcert.doleta.gov/performancedata.cfm

We have used this set of attributes and visa case status as initial data exploration and gain the understanding of its features. The raw dataset from OFLC had 610304 rows with 52 columns and consisted of 4 types of case status i.e 'CERTIFIED', 'DENIED', 'WITHDRAWN' and 'CERTIFIED WITHDRAWN'. For our objective we proceeded further with the subset of data of 136674 rows and 52 columns. The attributes (predictors) extracted consists of data types - categorical, numerical, date, textual, geospatial, and binary (Refer Table 1).

Datatype	**Predictors**
Categorical	Wage Level, PW Sources, Units of Pay
Text	Employer Name, Address, Job Title
Numeric	Wage, Total Workers
Binary	Full Time Position, H1B Dependent
Date	Application date, Employment Date
Geo-spatial	City, Country, ZIP Code

Table 1: Predictors with Data type category

The complete summary of attributes with its short descriptions can be referred in Appendix A. We are interested to predict visa case status which has two categories as “Certified” and “Denied”. Certified means the Labor Condition Application (LCA) of an employer is approved and Denied means the case is denied by OFLC.

Problem Statement.

Employers face difficulties to determine their chances of employees’ visa case status certified or denied as the H-1B visa allotment is lottery based. Also, there is need to identify whether there is any relationship between attributes and visa case status decision. If there is a relationship then which attributes are important and most impactful to the outcome of visa case status.

Business Case.

Employers want to predict the visa case status of the H-1B petition filled. The outcome of the visa case status will help employers to know what are the chances of their employees’ getting visa “CERTIFIED” before filing the H-1B petition. Besides, employers are interested to know which attributes are essential in the decision of visa case status.

Initial Hypothesis.

H_0: There is stratified randomness and hence there is no relationship between the predictors and case status.

H_A: There is a no stratified randomness and hence there is a relationship between the predictors and case status.

III. Data Preparation

This phase of the data analytics lifecycle involves exploring, preprocessing and conditioning the data prior to modeling and analyzing the results. In this case we had to ensure that the data is clean, robust and ready for feeding in R for modeling. The tools used for cleaning the data was CSV and R.

A. Data cleaning in CSV

Here we explored the raw data and analyzed the values in each of the attributes in our dataset. The following steps were undertaken in cleaning the dataset in the CSV.

1. Reducing the factor level of categorical attributes - The H1B dataset primarily consisted of categorical variables. We encoded a few attributes wherein we could club the categories such 50 states can be clubbed into four regions in the united states (see Appendix B). Similarly, attributes such as SOC_Code (Occupational code associated, classified by the Standard Occupational Classification (SOC) System) was reduced to 21 factor level and NAICS_Code (Industry code associated, classified by the North American Industrial Classification System (NAICS)) was reduced to factor level 20. Details of which are listed in Appendix C and D respectively.

2. Scaling of the attributes - One of the attribute Prevailing wage which contained the wage of the H1B visa applicant didn't had uniform unit pay. Some wages were specified in terms of per hour, while some were specified bi-weekly, weekly, monthly and so forth. Such nonuniform data would provide an inconsistent picture and hence we scaled all the wages up to annually.

3. Subset of the data - Due to the complexity issue and short completion time of the project we took subset of the data. In order to ensure the randomness of the data we selected

cases which were submitted within 90 days of employment, i.e. employment start day and case submitted for H1B within 90 days in the year 2017.

B. Data cleaning in R

1. Dummy coding - Except a few models, all the models require the categorical/factors to be converted into numeric form. For this purpose all the categorical attribute were converted into dummy variable using the function DummyVars from the caret package in R. Finally there were, 71 predictors after dummy coding. The list of final predictors can be seen in Appendix E.
2. Conversion of columns consisting date - The columns containing dates are not suitable during modeling in R and the attributes which contained dates such as employment start date, employment end date, decision date, case submitted date in the H1B dataset were important predictors. Hence we decided to convert them into number of days using the lubridate package in R. We subtracted each of these other days with case submitted date. So the new transformed attribute were as below:
    a. Decision Days - This attribute represents the the number of days taken to release the decision of H-1B case status. For instance, 60 days.
    b. Working Days -This gives number of days from the employment start date to the H-1B case submitted date.
    c. Remaining Days - This attribute represents the remaining number of days of the employment.

Apart from all this we also removed the rows that had missing values. It was not necessary to impute data since we had sufficient number of complete rows. The final number of rows at the end of this stage was 136, 674.

IV. Model Planning

As a first step we explored the data and selected the attributes to be used in the modeling. Followed by exploring the relationship of important attributes with the response variable. Finally, selecting the models to be build in the next phase which would help us achieve our goals for the project. The tools used in this step includes CSV and R. Let us see each of these steps in detail as below.

A. **Feature selection**

1. Removed attributes with missing value - Attributes with more than 80% missing value were removed. As imputation is the process of replacing missing values with either mean, median or other feature based on other values in the attribute. However, if more than 80% value is missing then imputing such an attribute would result in high manipulation of data. Hence, we decided to remove the below four attributes with more than 80% of missing values which are as below:

- Original_cert_date
- Public_disclosure_location
- Wage_Rate_Of_Pay_To
- Labor_Con_Agree

2. Removed irrelevant attributes - Irrelevant attribute or attribute with non predictive value were removed since including them in the modeling will not add any value in prediction accuracy. These attributes include:

- Visa class - Since only H1B visa status was included in our project scope, so adding it as a separate attribute was not required.

- Employer_Phone
- Employer_Phone_Ext
- Employer Address

3. Removed duplicate attributes - The dataset consists of some attributes that were providing the same information. For example city, state, pin code etc. provide the same information. Such attributes were removed. We kept the attribute Employer_State and Worksite_State and removed the following attributes.

- Employer_City
- Worksite_City
- Worksite_County
- Employer_Province
- Employer_Postal_Code
- SOC Name
- Agent_Attorney_City
- Agent_Attorney_State

4. Removed attributes with unique factor levels - The dataset contained attributes with unique value more than 50k. These many factor levels makes it computationally expensive for modeling in R. Hence attributes with many unique values were removed and they are as below:

- Case_Number
- Employer Name
- Job Title

5. Removed predictor highly correlated to response variable - This step was performed in R using the cor function. The features which have high correlation with response variable are

strong predictors in a model. If we include such variables in the models it will have the maximum impact during prediction. The Appendix F lists the correlation of each predictor with dependent variable for the H1B dataset. We then decided to remove the attribute Decision Days which was highly correlated to the response variable.

B. Data Exploration

As a first step towards data exploration we plotted the graph of pairs with important variables and the response variable case status to understand the relationship of the data. The graph of the same is as below (see figure 2).

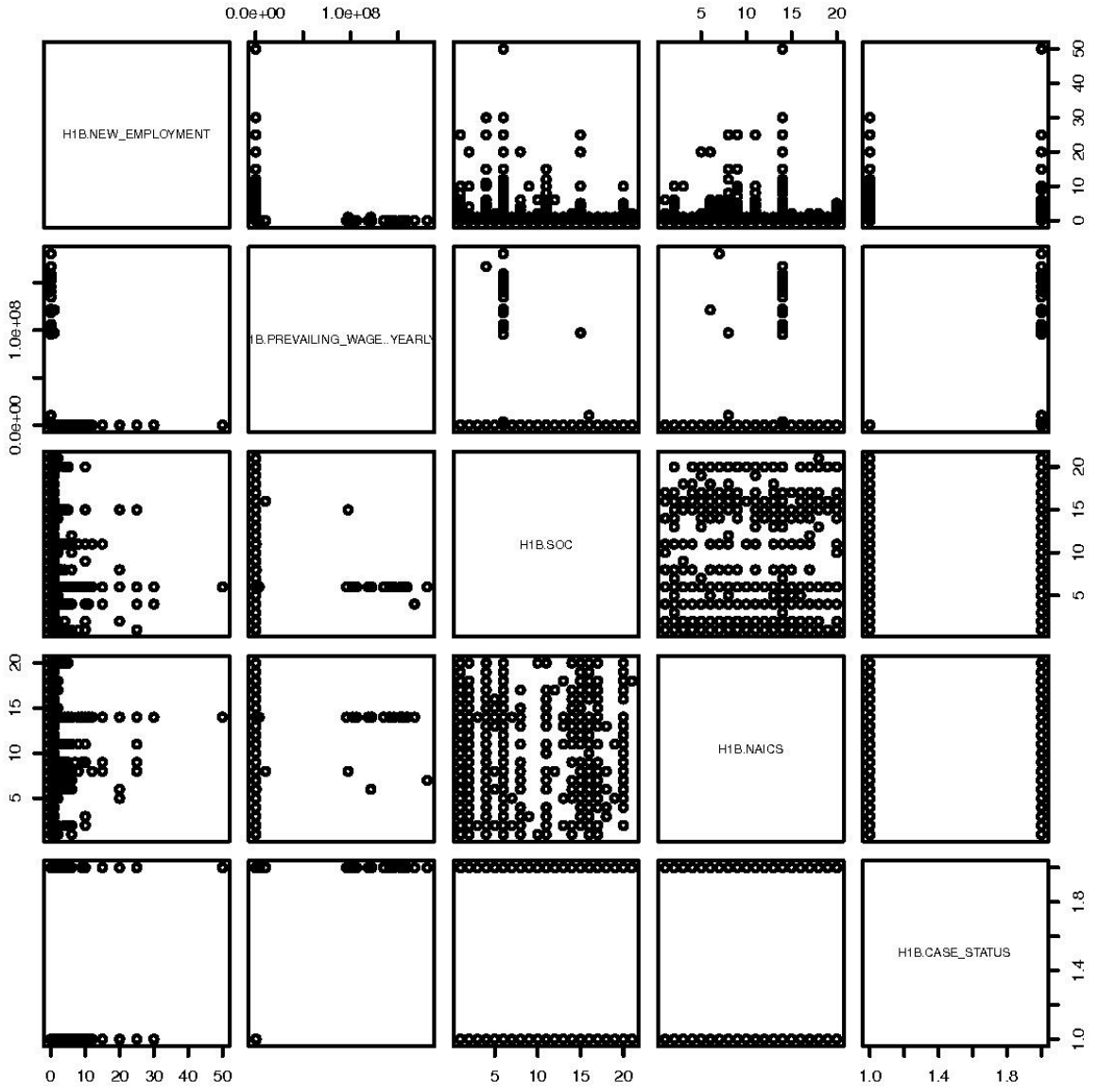


Figure 2: Data exploration of important attribute using Pairs

C. Principal Component Analysis

Principal Component Analysis (PCA) was performed on 71 numeric predictor so as to reduce the dimensionality and analyze if more variance can be explained in the model by including the variables that explains maximum variance. The proportion of variance explained in our dataset can be seen from the Figure XX as below.

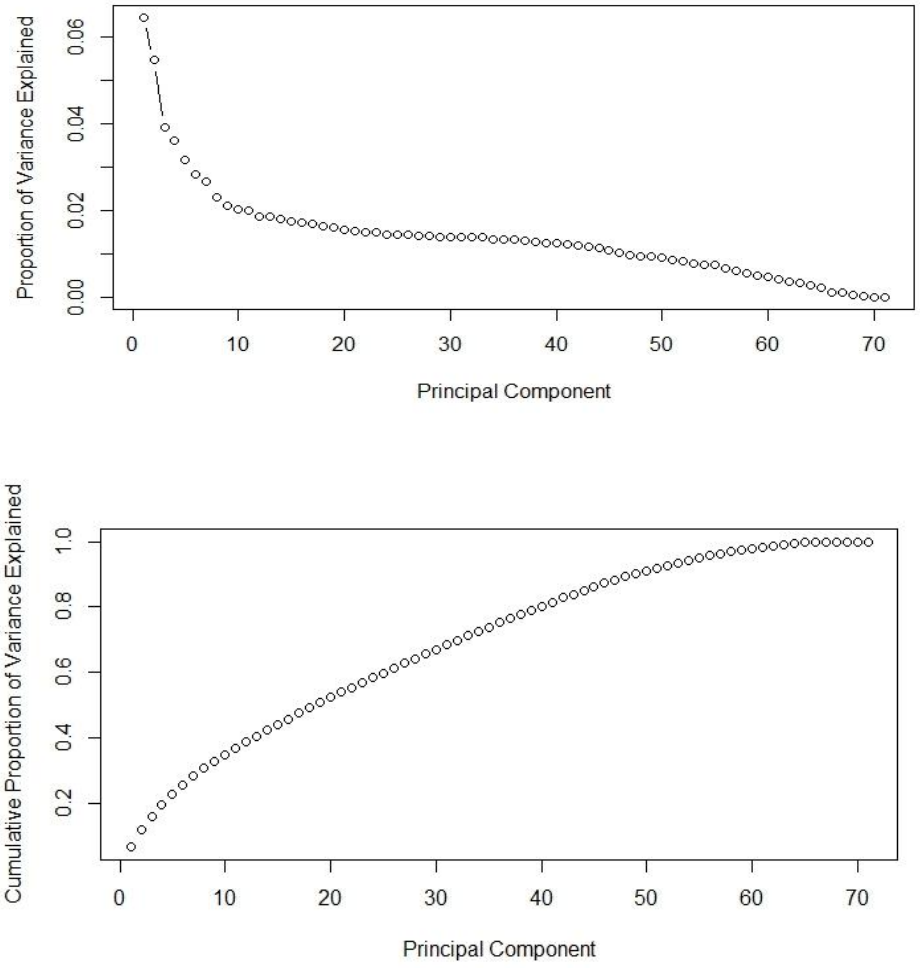


Figure 3: Principal components

The above graph clearly shows that PCA is of very not much helpful for our data since we would have to include many components. To have maximum cumulative proportion of variance of components, we have to include large number of principal components. Since the 90% of cumulative proportion of variance would have maximum components, we have decided to include 50 principal components to train our model using PCA.

D. Model Selection

Since we are predicting a classification class with two levels - “Certified” and “Denied”. We selected classification models which included models with and without PCA. The models selected are as below:

1. Logistic Regression
2. Linear Discriminant Analysis (LDA)
3. Decision Tree
4. Naive Bayes
5. Generalized Additive Model (GAM)
6. Partial Least Square (PLS)
7. Glmnet
8. Random Forest
9. Bagging
10. Boosting

V. Model Building

In order to run the models decided in model planning stage we first need to split the data into train and test set. The models will be trained on the training data and then the performance of those trained model will be tested on data held aside which is known as test data. In this project we decided to have a split of 70-30. 70% of the data from the dataset was used for modeling and remaining 30% was held out for testing the performance.

Number of rows after splitting the dataset into train and test	
Train Data	Test Data
95,672	41,002

Before going ahead with model building, we observed a severe class imbalance in the H1B dataset. Initially, we had 98% of our response had the case status as “CERTIFIED” and only 2% of our response had their case status status as “DENIED”.

Number of imbalanced cases before resampling	
Certified	Denied
94,549	1,123

As we already know that it is not advisable to train our models on such an unbalanced data since such a model will not show us the clear picture because the prediction accuracy of such a data will generally result in high accuracy for the train as well as test dataset since the classifier will train its model in such a way that it will predict the majority class, i.e. the positive class “CERTIFIED” in our case. And hence the accuracy and sensitivity of such a data would be high on almost all the models built which indicates that the model is overfit. Hence in order to deal with this we decided to resample our training data, the test data was untouched. In order to test each of these technique we built flexible as well as less flexible models, viz, Decision Tree, Logistic Regression, Random Forest, and GAM. The following techniques were tried for balancing the classes:

a. ROSE - The ROSE function in R tries to create synthetic data by Randomly Over Sampling Examples (ROSE). It produces a synthetic, balanced sample of data simulated as per the smoothed-bootstrap approach. ROSE was the first balancing technique we tried on our dataset since it randomly creates a 50-50 balance in the

classes. However, the results indicated an overfit model, i.e. accuracy of the decision tree was 99%. Hence we moved on and tried other balancing techniques.

b. Ovun.Sample - Ovun.Sample function in R creates possibly balanced data by random oversampling minority examples, under-sampling majority examples or combination of over and undersampling. We tried combination of over and undersample on our data. Even after balancing the class with combination of over and undersample the performance were not satisfactory.

c. Upsample - Upsampling function in R tries to create a balanced data by increasing the minority class and upsampling it to number of occurrences same as majority class. We tried upsampling because since we had 90k observations it was computationally feasible to upsample the minority class. However, no improvement in the performance of the models test were observed. The accuracy of decision tree was as high as 97%.

d. SMOTE - Synthetic Minority Over-sampling Technique tries to balance the data by increasing the minority class randomly and decreasing the majority class randomly. We tried several combination of percentage, i.e. first we tried to balance the classes as 60-40. However, no improvement in the performance was observed. Then we tried a perfect 50-50 balance which seemed to worked well for out dataset and hence we decided to go ahead with this split.

Number of balanced cases after resampling using SMOTE	
Certified	Denied
57,273	57,273

Ten folds cross validation was performed using the caret package in R on the training data so as to ensure optimal performance of the model.

Threshold adjustments were made to increase the performance (increase specificity) of each model. Since our data was highly imbalanced and it is the tendency of the highly imbalanced data to have a performance high sensitivity and overall high accuracy. To balance this situation, we adjusted the threshold of each model accordingly.

For purposes of this project, correct identification of certified as case status is “true positive”, correct identification of denied as case status is “true negative”, and the error rates are false positives and false negatives. A confusion matrix depicts the numbers of true/false positives and true/false negatives used to calculate the sensitivity and specificity.

After this model specified in the model planning stage were executed keeping in mind the above mentioned considerations. Appendix G contains the R code used to build these models.

Decision Tree

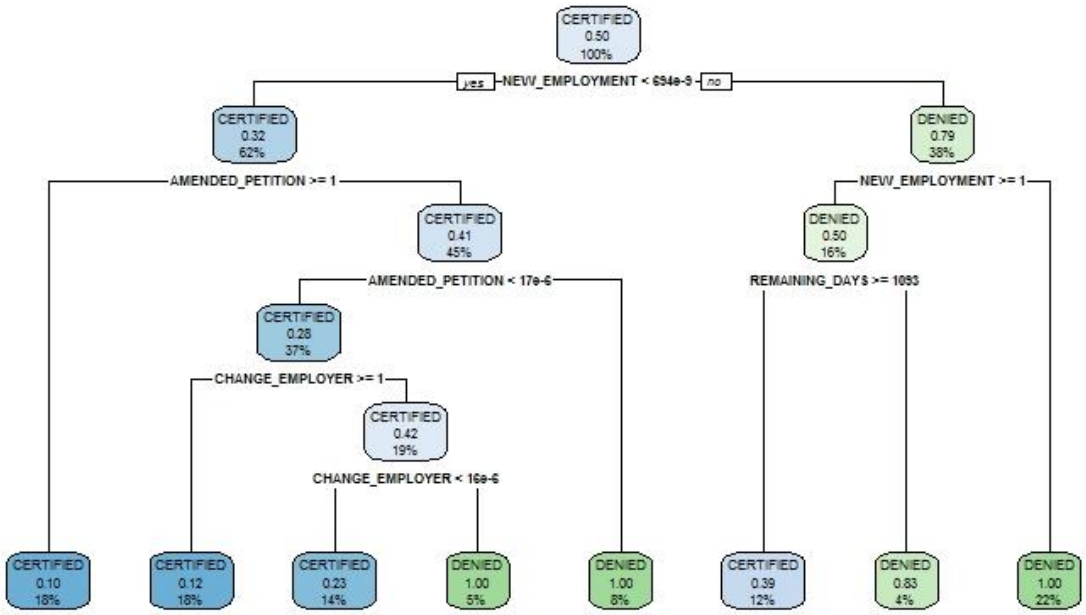


Figure 4: Decision Tree

Decision Tree was firstly executed to get the overall idea about the data and the relationship between predictors. Decision Tree helped us to infer the important predictors on which the tree would split since the algorithm of the decision tree finds the predictor with most informative attribute and then splits on that attribute. Hence, the goal was to find important predictors, explore the data, and to understand the relationships amongst the predictors. The important predictors as per pruned decision tree can be seen from the above Figure 4. Important predictors as per Decision Tree are New Employment, Amended Petition, Remaining Days, and Change Employers.

Later, less flexible models such as logistic regression, decision tree followed by little flexible classifiers such as Linear Discriminant Analysis, Naive Bayes, followed by flexible models such as General Additive models, Partial Least Square, Glment lastly followed by ensembles such as Random Forest, Bagging, Boosting.

VI. Results and Performance

To assess the performance of the models we first evaluated the train performance of all the models specified in the model planning stage. We assessed the performance of the models with and without PCA separately.

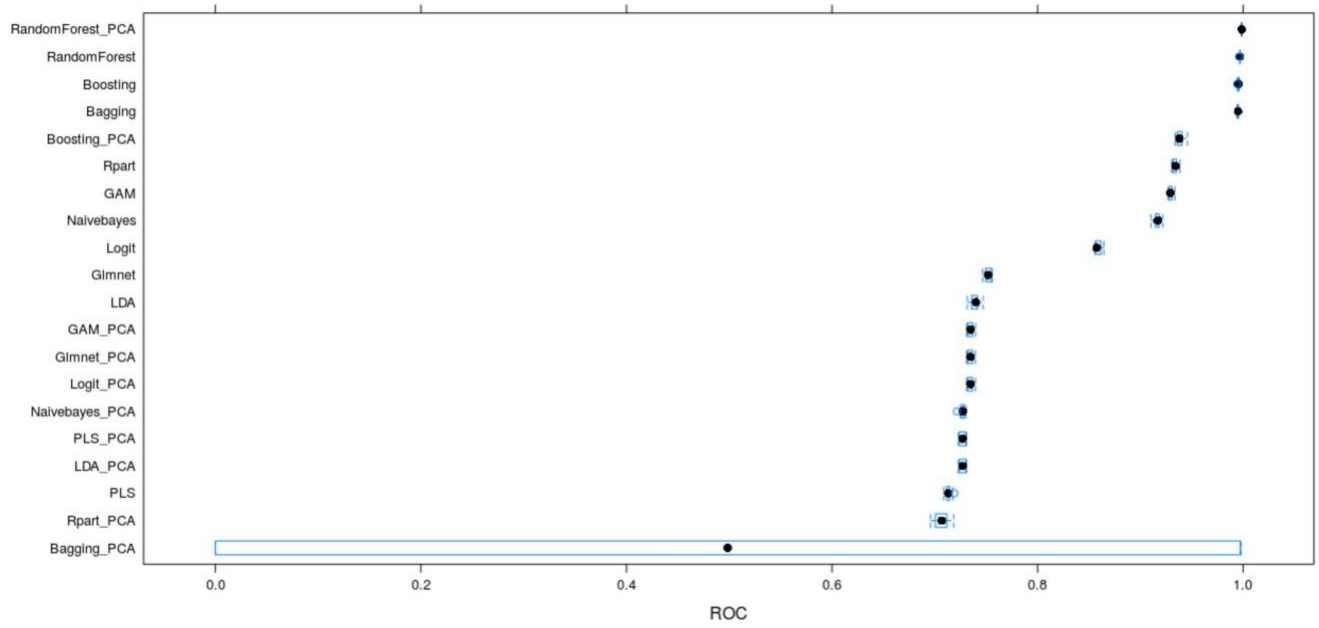


Figure 5: Train performance of ROC

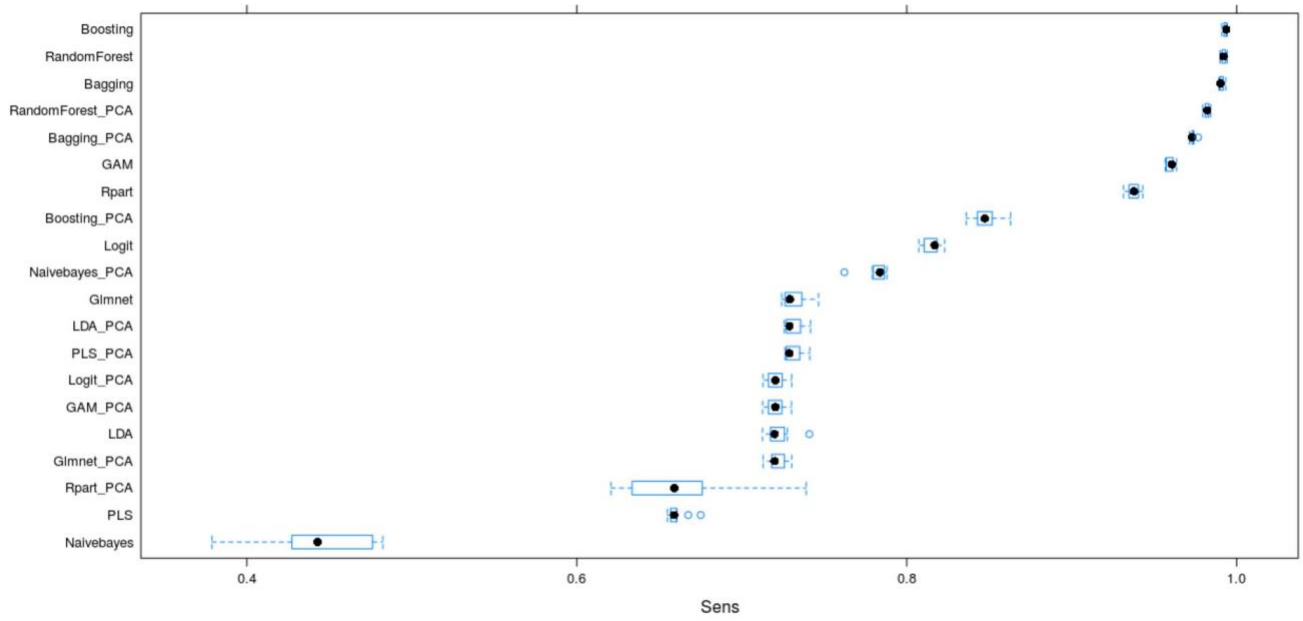


Figure 6: Train performance of Sensitivity

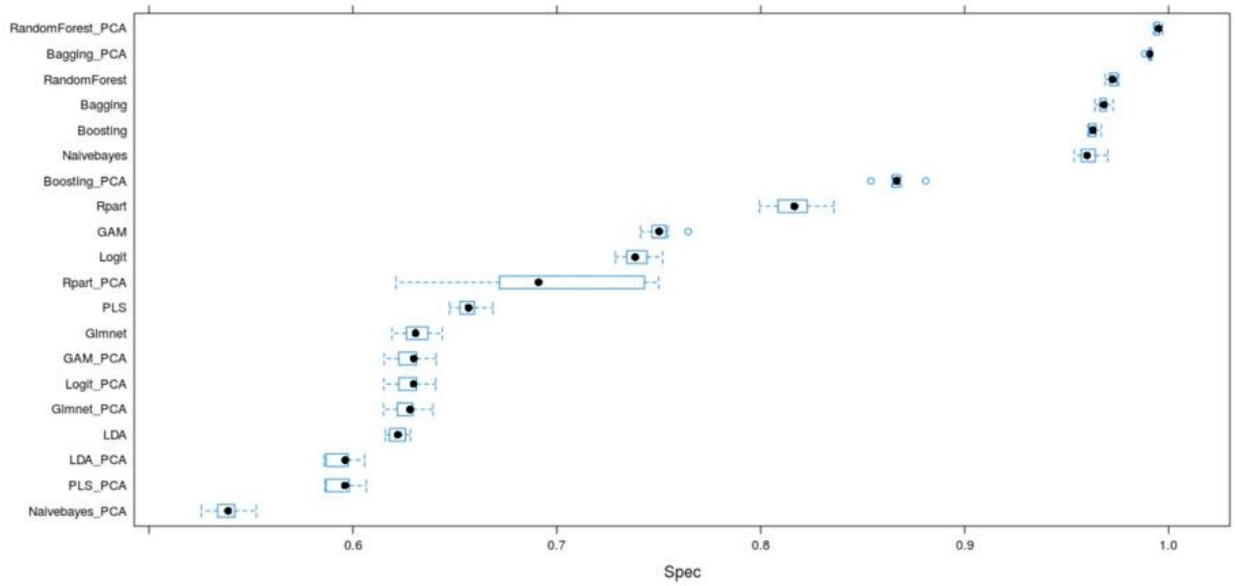


Figure 7: Train performance of Sensitivity

Looking at the train performance graphs of ROC (see figure 5), sensitivity (see figure 6), specificity (see figure 7) of all the models we can clearly infer that ensemble, specifically

random forest consistently performed well on all the three metric and outperformed as compared to other models.

Test Performance of the models

As we know that train performance is not a correct indicator of measuring the performance of the model. Rather, we assess the test performance for each of our models. The metric to assess the test performance used were ROC, Sensitivity, Specificity, AUC, and Accuracy to some extent. We decided to use these metrics instead of only accuracy since with imbalanced data accuracy gives a misleading results, i.e. the high accuracy is actually an overfit model. Apart from that we also considered precision and recall for assessing the performance of our models. For individual ROC of each model please refer to Appendix H.

Below graph shows the performance based on Sensitivity.

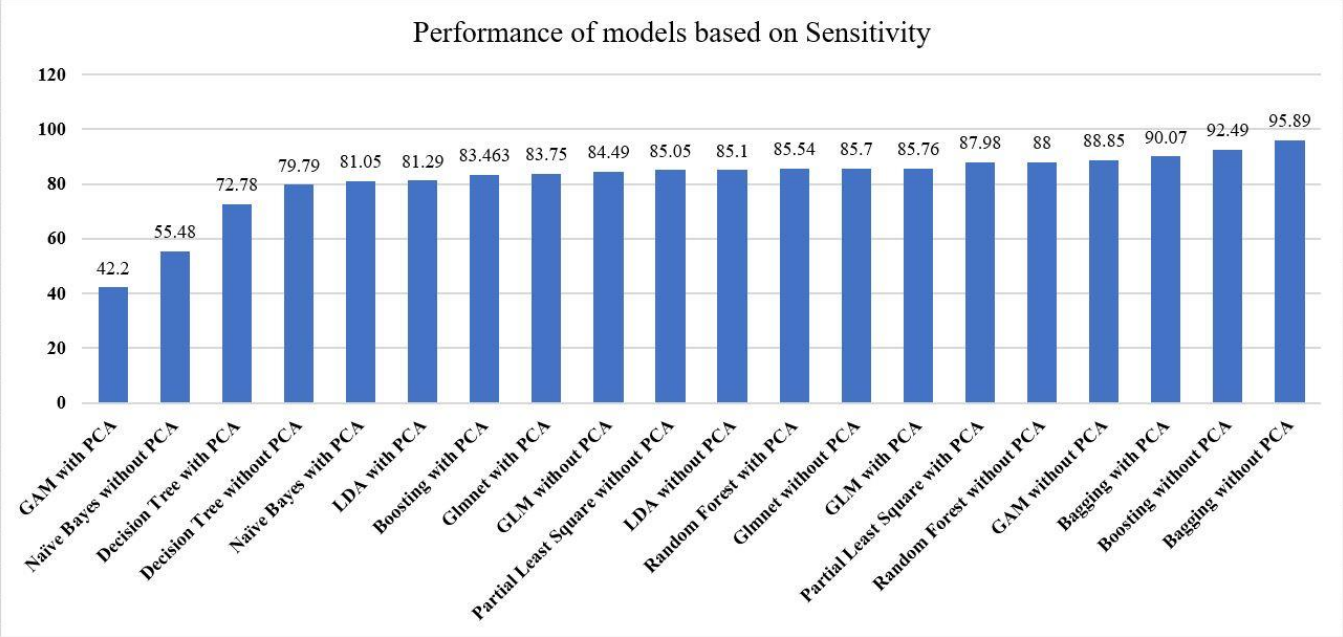


Figure 8: Test performance based on Sensitivity of all the models

From the above Figure 8,based on sensitivity we can see that the performance of Bagging without PCA, Boosting without PCA, Bagging with PCA, GAM without PCA, and

Random Forest without PCA had better performance as compared to other models. However, the performance of Bagging without PCA, Boosting without PCA, Bagging with PCA seems to be overfit. On the other hand GAM with PCA and Naive Bayes without PCA didn't perform well in terms of sensitivity.

Below graph shows the performance based on Specificity

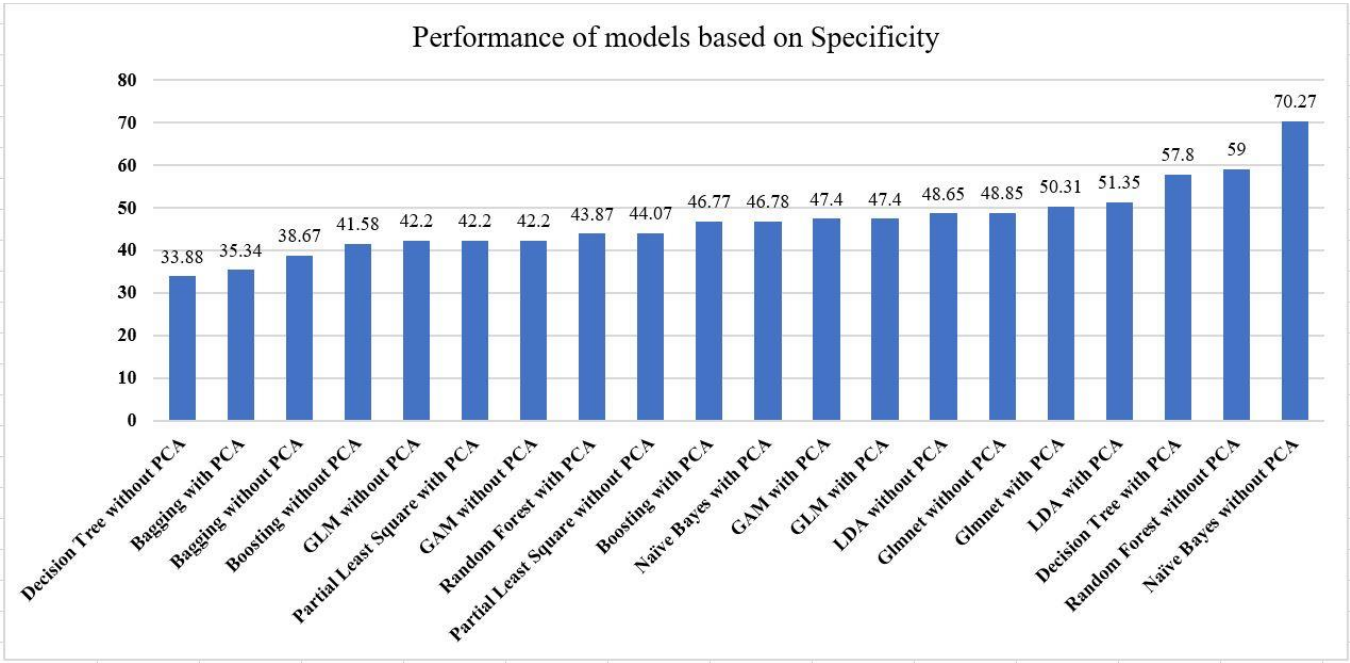


Figure 9: Test performance based on Specificity of all the models

From the above graph (see figure 9), based specificity on we can see that the performance of Naive Bayes without PCA, Random Forest without PCA, and Decision Tree with PCA had better performance as compared to other models. On the other hand Decision Tree without PCA, Bagging with PCA, Bagging without PCA didn't perform well in terms of Specificity.

Below graph shows the performance based on Accuracy.

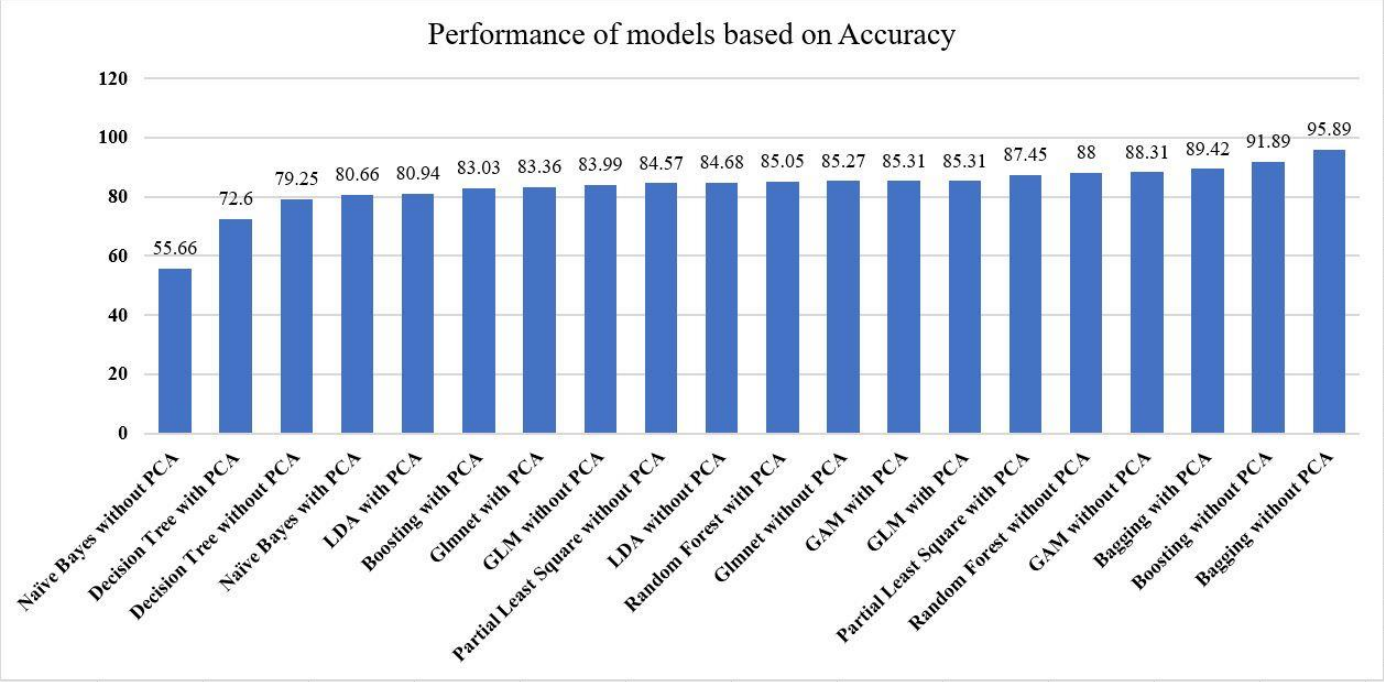


Figure 10: Test performance based on Accuracy of all the models

From the Figure 10, we can see that for our model the results of accuracy were similar to the results of sensitivity. Hence, models which performed well on Accuracy are Bagging without PCA, Boosting without PCA, Bagging with PCA, GAM without PCA, and Random Forest without PCA. However, the performance of Bagging without PCA, Boosting without PCA, Bagging with PCA seems to be overfit. On the other hand Naive Bayes without PCA, Decision Tree with PCA, Decision Tree without PCA performed terribly.

Below graph shows the performance based on AUC.

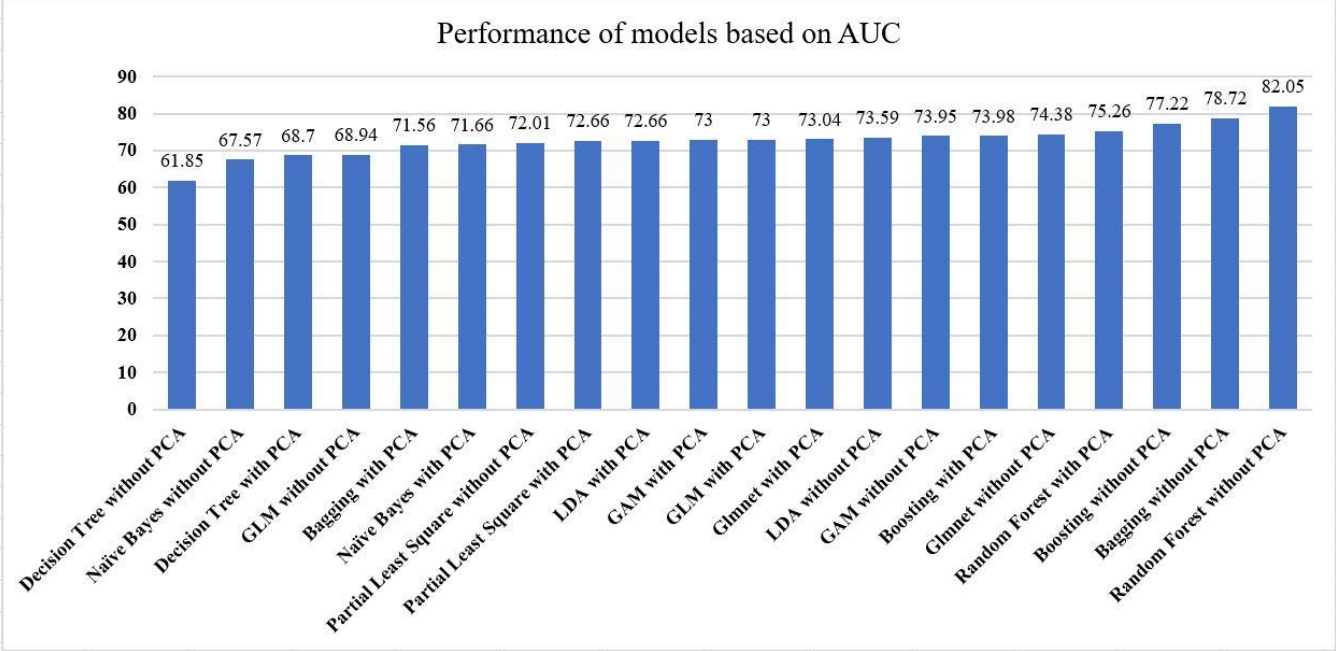


Figure 11: Test performance based on AUC of all the models

From the above graph (see figure 11), based on AUC we can see that the performance of Bagging without PCA, Boosting without PCA, Random Forest without PCA, Random Forest with PCA had better performance as compared to other models. On the other hand Naive Bayes without PCA, Decision Tree with PCA, Decision Tree without PCA and, GLM with PCA performed terribly.

Below graph based on Precision and Recall.

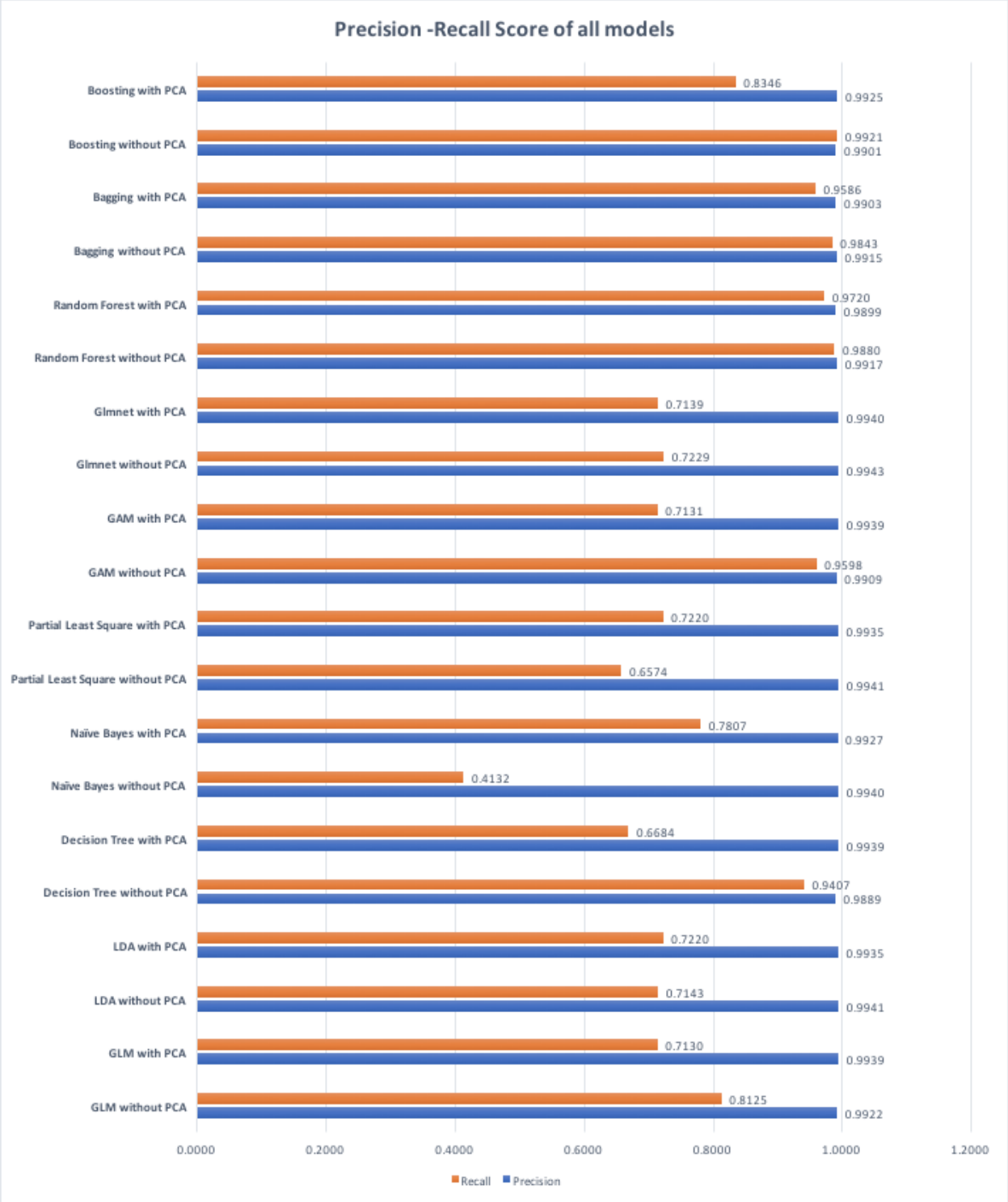


Figure 12: Precision-Recall Score of all models

Precision and Recall are useful metrics to assess the performance of the models especially in the case of class imbalance.

Precision refers to the ratio of correctly predicted positive observations to the total predicted positive observations. High precision relates to the low false positive rate. On the other hand recall refers to the ratio of correctly predicted positive observations to the all observations in actual class.

From the above graph(see figure 12), which plots the precision and recall scores for all the models we can see that performance of ensembles such as Random Forest without PCA, Random Forest with PCA, Bagging without PCA, Bagging with PCA, Boosting without PCA is better in terms of both precision as well as recall as compared to other models.

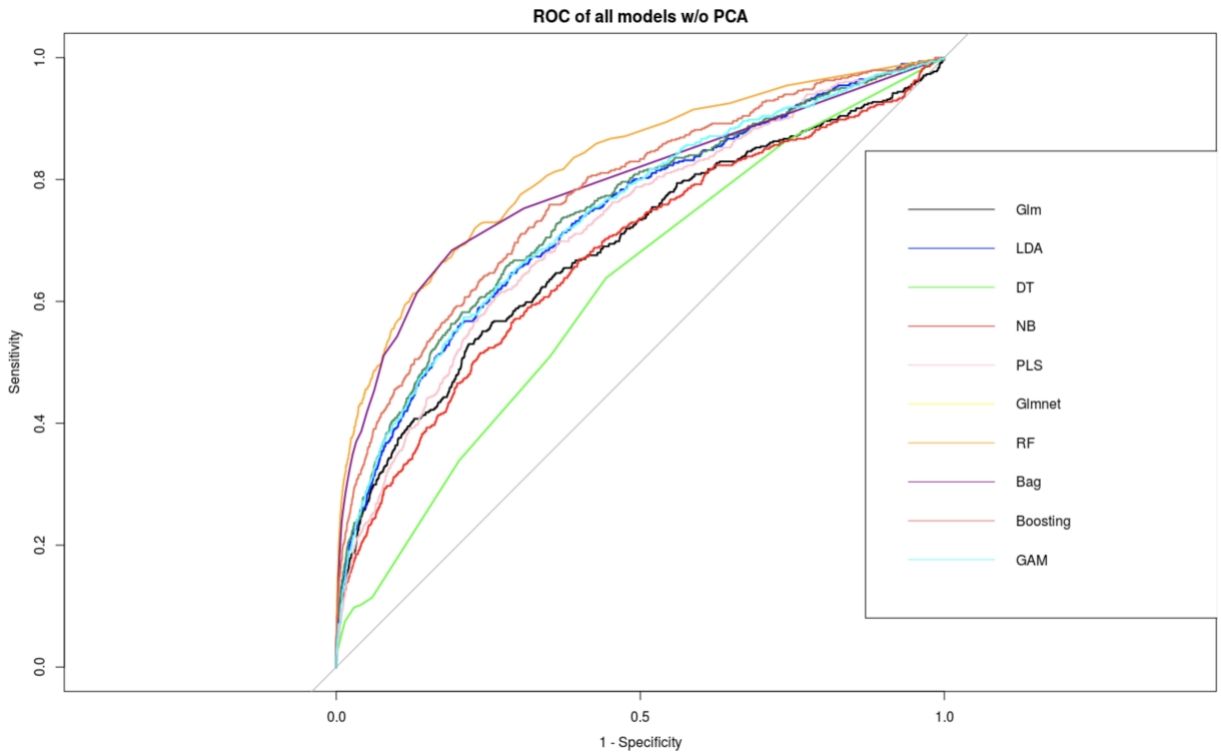


Figure 13: ROC of all model without PCA

The above Figure 13, shows that Random forest with PCA model has maximum Area under the curve. Next, Boosting and Bagging also has higher area under the curve.

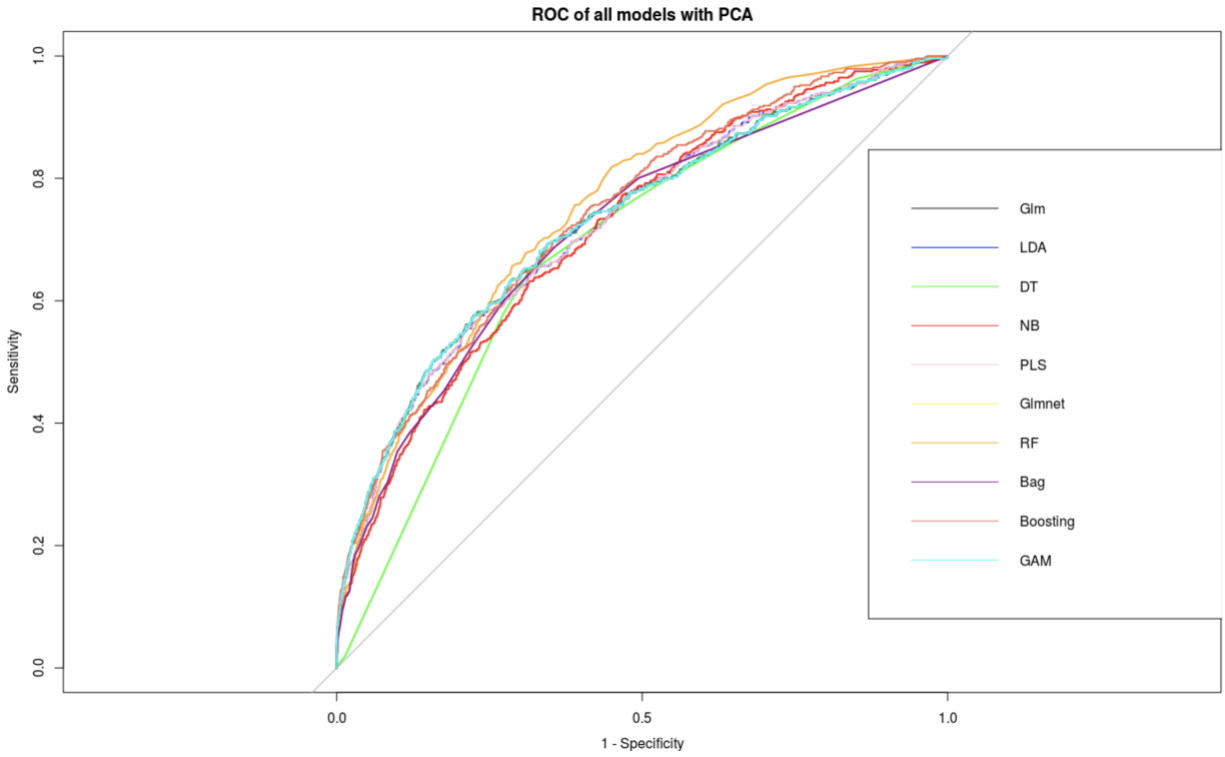


Figure 14: ROC of all model with PCA

In the above Figure 14, we can see that Random forest with PCA has maximum area under the curve. Boosting with PCA has next higher area under the curve.

Out all of the above models (Refer Figures 9-14), Bagging without PCA model has performed really outstanding in terms of Sensitivity (95.89%) as compared to other models like Bagging with PCA (89.42%), Random Forest without PCA (88%), Random Forest with PCA (85.05%), and GLMNET without PCA (85.27%). Test performance of Accuracy metrics is same as that of sensitivity in all the models.

The specificity of Naive Bayes is maximum i.e. 70.27% but it didn't perform on sensitivity metrics. We also see that sensitivity of Naive Bayes without PCA is minimum i.e. 55.66%. The shift in the performance of Naive bayes model can be seen in with PCA model where we got sensitivity of

81.05%. Thus, naive bayes model is very dimension sensitive so application of PCA on naive bayes model boosted its test performance.

In terms of Specificity metrics is Random Forest without PCA performed greatly after Naive Bayes. Bagging without PCA gave specificity of 38.67% and Bagging with PCA gave 35.34%. In the AUC metrics Bagging without PCA model gave 78.72% which less compared to the AUC of Random Forest without PCA i.e. 82.05%. Even though Bagging without PCA and Bagging with PCA models performed well on Sensitivity and Accuracy metrics, didn't perform well on Specificity and AUC metrics. Thus, Bagging without PCA and Bagging with PCA models are overfitted on H1B visa case status data.

After considering and comparing all the metrics of all the models, Random Forest without PCA performed best on our data and outperformed other 19 models by giving us the sensitivity of 88%, specificity of 59%, accuracy of 88% and Area under the curve of 82.05%.

Below is the detailed performance for Random Forest without PCA (see figure 15).

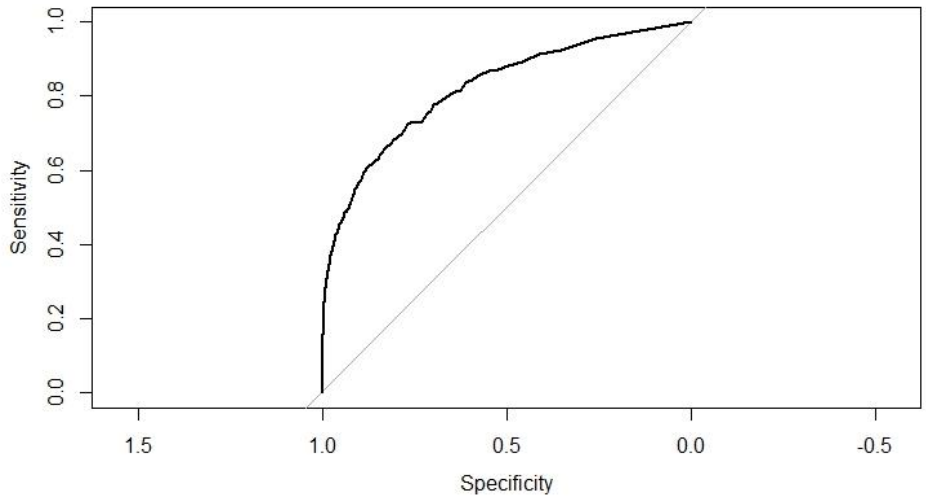


Figure 15: ROC curve for Random Forest without PCA

AUC: 0.8205 Precision: 0.9916 Recall: 0.9880

Confusion Matrix and Statistics

Reference

Prediction CERTIFIED DENIED
CERTIFIED 35936 195
DENIED 4585 286

Accuracy : 0.8834
95% CI : (0.8803, 0.8865)
No Information Rate : 0.9883
P-Value [Acc > NIR] : 1

Kappa : 0.0874
Mcnemar's Test P-Value : <2e-16

Sensitivity : 0.88685
Specificity : 0.59459
Pos Pred Value : 0.99460
Neg Pred Value : 0.05871
Prevalence : 0.98827
Detection Rate : 0.87645
Detection Prevalence : 0.88120
Balanced Accuracy : 0.74072

'Positive' Class : CERTIFIED

Important Variables

As one of the goal for this project is to identify important variables affecting the case status for H1B applicant since as per our hypothesis there is a random stratified pattern in which the case status our declared. We wanted to identify these attributes so as to help the employers filing the visa application to understand which factors would increase the chances of their applicant to gain as case status certified. For this purpose, we used varImp function from the caret package in R. Based on the results of our modeling we interpreted that the most important variables affecting the case status are as below.

- New Employment
- Amendment Petition

- SOC
- NAICS
- Change Employer

VII. Discussion and Recommendations

Our model predicts results with 88% of accuracy. Even if there is a possibility of 12% of inaccurate prediction, we still recommend to use our model, as this would help our clients (i.e employer) to understand the important factors that has significant effect on H-1B status. This would increase chances of employees to receive H-1B status certified. Sponsoring a wrong candidate would result in loss of time and funds reserved for H-1B sponsorship. This would eventually hamper revenue of company. Secondly, service based companies might lose their clients due to unavailability of H-1B certified resources. Our model would help to reduce the risk of sponsoring the wrong candidate and to invest on right candidates whose profile has potential to receive H-1B visa. Additionally, choosing the right candidate would speed up the process of documentation and selection and would help them to efficiently manage their time. Choosing right candidates (specifically for service based industry) could increase their client retention and increase their revenue.

We recommend our client to perform test run on our model with sample data before deploying to production. This would help them to perform environmental check, reduce errors, check the initial performance and understand the functionality. Additionally, H-1B status cycle is processed once a year but we recommend our client to run our model quarterly. As quarterly result will help to receive precise results, reduce variations and help them to understand the repetitive inferential factors affecting the status.

Reference

1. https://www.foreignlaborcert.doleta.gov/performancedata.cfm
2. http://dataaspirant.com/2018/01/15/feature-selection-techniques-r/
3. https://www.rdocumentation.org/packages/ROSE/versions/0.0-3/topics/ROSE
4. https://www.rdocumentation.org/packages/ROSE/versions/0.0-3/topics/ovun.sample
5. https://www.analyticsvidhya.com/blog/2017/03/imbalanced-classification-problem/

APPENDICES

Appendix A: A complete attributes

CASE_NUMBER	Unique identifier assigned to each application submitted for processing to the Chicago National Processing Center.
CASE_SUBMITTED	Date and time the application was submitted.
DECISION_DATE	Date on which the last significant event or decision was recorded by the Chicago National Processing Center.
VISA_CLASS	Indicates the type of temporary application submitted for processing. R = H-1B; A = E-3 Australian; C = H1B1 Chile; S = H-1B1 Singapore. Also, referred to as "Program" in prior years.
EMPLOYMENT_START_DATE	Beginning date of employment.
EMPLOYMENT_END_DATE	Ending date of employment.
EMPLOYER_NAME	Name of employer submitting labor condition application.
EMPLOYER_BUSINESS_DBA	Trade Name or dba name of employer submitting labor condition application, if applicable.
EMPLOYER_ADDRESS	Contact information of the Employer requesting temporary labor certification.
EMPLOYER_CITY	
EMPLOYER_STATE	
EMPLOYER_POSTAL_CODE	
EMPLOYER_COUNTRY	
EMPLOYER_PROVINCE	

EMPLOYER_PHONE	
EMPLOYER_PHONE_EXT	
AGENT_REPRESENTING_EMPLOYER	Is agent representing? Yes/No
AGENT_ATTORNEY_NAME	Name of Agent or Attorney filing an H-1B application on behalf of the employer.
AGENT_ATTORNEY_CITY	City information for the Agent or Attorney filing an H-1B application on behalf of the employer.
AGENT_ATTORNEY_STATE	State information for the Agent or Attorney filing an H1B application on behalf of the employer.
JOB_TITLE	Title of the job
SOC_CODE	Occupational code associated with the job being requested for temporary labor condition, as classified by the Standard Occupational Classification (SOC) System.
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).
NEW_EMPLOYMENT	Indicates requested worker(s) will begin employment for new employer as defined by USCIS I-29.
CONTINUED_EMPLOYMENT	Indicates requested worker(s) will be continuing employment with same employer, as defined by USCIS I-29.
CHANGE_PREVIOUS_EMPLOYMENT	Indicates requested worker(s) will be continuing employment with same employer without material change to job duties, as defined by USCIS I-29.

NEW_CONCURRENT_EMPLOYMENT	Indicates requested worker(s) will begin employment with additional employer, as defined by USCIS I-29.
CHANGE_EMPLOYER	Indicates requested worker(s) will begin employment for new employer, using the same classification currently held, as defined by USCIS I-29.
AMENDED_PETITION	Indicates requested worker(s) will be continuing employment with same employer with material change to job duties, as defined by USCIS I-29.
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_WAGE_LEVEL	Variables include "I", "II", "III", "IV" or "N/A."
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", "BiWeekly", "Month", or "Year".
H1B_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
SUPPORT_H1B	Y = Employer will use the temporary labor condition application only to support H-1B petitions or extensions of status of exempt H-1B worker(s); N = Employer will not use the temporary labor condition application to support H-1B petitions or extensions of status for exempt H-1B worker(s); N/A = not applicable.
LABOR_CON_AGREE	Y = Employer agrees to the responses to the Labor Condition Statements as in the subsection; N = Employer does not agree to the responses to the Labor Conditions Statements in the subsection.
PUBLIC_DISCLOSURE_LOCATION	Variables include "Place of Business" or "Place of Employment."
WORKSITE_CITY	City information of the foreign worker's intended area of employment.
WORKSITE_COUNTY	County information of the foreign worker's intended area of employment.
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.
ORIGINAL_CERT_DATE	Original Certification Date for a Certified_Withdrawn application.
CASE_STATUS	Case Status - Categories: "CERTIFIED", "DENIED", "WITHDRAWN", and "CERTIFIED WITHDRAWN"

Appendix B: Mapping of States to Regions in United States

States	Region
AL	South Region
AK	West Region
AZ	West Region
AR	South Region
CA	West Region
CO	West Region
CT	Northeast Region
DE	South Region
FL	South Region
GA	South Region
HI	West Region
ID	West Region
IL	Midwest Region
IN	Midwest Region
IA	Midwest Region
KS	Midwest Region
KY	South Region
LA	South Region
ME	Northeast Region
MD	South Region
MA	Northeast Region
MI	Midwest Region
MN	Midwest Region
MS	South Region
MO	Midwest Region
MT	West Region
NE	Midwest Region
NV	West Region
NH	Northeast Region
NJ	Northeast Region
NM	West Region
NY	Northeast Region
NC	South Region
ND	Midwest Region

OH	Midwest Region
OK	South Region
OR	West Region
PA	Northeast Region
RI	Northeast Region
SC	South Region
SD	Midwest Region
TN	South Region
TX	South Region
UT	West Region

VT	Northeast Region
VA	South Region
WA	West Region
WV	South Region
WI	Midwest Region
WY	West Region

Appendix C: Details of mapping SOC code

SOC_Code	Occupations
11	Management Occupations
13	Business and Financial Operations Occupations
15	Computer and Mathematical Occupations
17	Architecture and Engineering Occupations
19	Life, Physical, and Social Science Occupations
21	Community and Social Service Occupations
23	Legal Occupations
25	Education, Training, and Library Occupations
27	Arts, Design, Entertainment, Sports, and Media Occupations
29	Healthcare Practitioners and Technical Occupations
31	Healthcare Support Occupations
33	Protective Service Occupations
35	Food Preparation and Serving Related Occupations
37	Building and Grounds Cleaning and Maintenance Occupations
39	Personal Care and Service Occupations
41	Sales and Related Occupations
43	Office and Administrative Support Occupations

45	Farming, Fishing, and Forestry Occupations
47	Construction and Extraction Occupations
49	Installation, Maintenance, and Repair Occupations
51	Production Occupations
53	Transportation and Material Moving Occupations
55	Military Specific Occupations

Appendix D: Details of mapping NAICS code

2012 NAICS Code	2012 NAICS Title
11	Agriculture, Forestry, Fishing and Hunting
21	Mining, Quarrying, and Oil and Gas Extraction
22	Utilities
23	Construction
31-33	Manufacturing
42	Wholesale Trade
44-45	Retail Trade
48-49	Transportation and Warehousing
51	Information
52	Finance and Insurance
53	Real Estate and Rental and Leasing
54	Professional, Scientific, and Technical Services
55	Management of Companies and Enterprises
56	Administrative and Support and Waste Management and Remediation Services
61	Educational Services
62	Health Care and Social Assistance

71	Arts, Entertainment, and Recreation
72	Accommodation and Food Services
81	Other Services (except Public Administration)
92	Public Administration

Appendix E: List of Predictors after Dummy coding

WORKING_DAYS

REMAINING_DAYS

EMPLOYER_REGION.Northeast.Region

EMPLOYER_REGION.South.Region

EMPLOYER_REGION.West.Region

AGENT_REPRESENTING_EMPLOYER.Y

SOC.Arts..Design..Entertainment..Sports..and.Media.Occupations

SOC.Building.and.Grounds.Cleaning.and.Maintenance.Occupations

SOC.Business.and.Financial.Operations.Occupations

SOC.Community.and.Social.Service.Occupations

SOC.Computer.and.Mathematical.Occupations

SOC.Construction.and.Extraction.Occupations

SOC.Education..Training..and.Library.Occupations

SOC.Farming..Fishing..and.Forestry.Occupations

SOC.Food.Preparation.and.Serving.Related.Occupations

SOC.Healthcare.Practitioners.and.Technical.Occupations

SOC.Healthcare.Support.Occupations

SOC.Installation..Maintenance..and.Repair.Occupations

SOC.Legal.Occupations

SOC.Life..Physical..and.Social.Science.Occupations

SOC.Management.Occupations

SOC.Office.and.Administrative.Support.Occupations

SOC.Personal.Care.and.Service.Occupations

SOC.Production.Occupations

SOC.Sales.and.Related.Occupations

SOC.Transportation.and.Material.Moving.Occupations

NAICS.Administrative.and.Support.and.Waste.Management.and.Remediation.Services

NAICS.Agriculture..Forestry..Fishing.and.Hunting

NAICS.Arts..Entertainment..and.Recreation

NAICS.Construction

NAICS.Educational.Services

NAICS.Finance.and.Insurance

NAICS.Health.Care.and.Social.Assistance

NAICS.Information

NAICS.Management.of.Companies.and.Enterprises

NAICS.Manufacturing

NAICS.Mining..Quarrying..and.Oil.and.Gas.Extraction

NAICS.Other.Services..except.Public.Administration.

NAICS.Professional..Scientific..and.Technical.Services

NAICS.Public.Administration

NAICS.Real.Estate.and.Rental.and.Leasing

NAICS.Retail.Trade

NAICS.Transportation.and.Warehousing

NAICS.Utilities

NAICS.Wholesale.Trade

TOTAL_WORKERS

NEW_EMPLOYMENT

CONTINUED_EMPLOYMENT

CHANGE_PREVIOUS_EMPLOYMENT

NEW_CONCURRENT_EMPLOYMENT

CHANGE_EMPLOYER

AMENDED_PETITION

FULL_TIME_POSITION.Y

PREVAILING_WAGE..YEARLY.

PW_WAGE_LEVEL.Level.II

PW_WAGE_LEVEL.Level.III

PW_WAGE_LEVEL.Level.IV

PW_WAGE_LEVEL.N.A

PW_SOURCE.DBA

PW_SOURCE.OES

PW_SOURCE.Other

PW_SOURCE.SCA

PW_SOURCE_YEAR

WAGE_RATE_OF_PAY_FROM.YEARLY.

H1B_DEPENDENT.Y

WILLFUL_VIOLATOR.Y

SUPPORT_H1B.N.A

SUPPORT_H1B.Y

WORKSITE_REGION.Northeast.Region

WORKSITE_REGION.South.Region

WORKSITE_REGION.West.Region

Appendix F: Correlation of predictors to Response Variable

Name of the Attribute	Value	Absolute Value
CASE_STATUS.DENIED (Response Variable)	***1***	***1***
DECISION_DAYS	-0.218070304	0.218070304
PREVAILING_WAGE..YEARLY.	0.094854924	0.094854924
SOC.Food.Preparation.and.Serving.Related.Occupations	0.071698237	0.071698237
PW_WAGE_LEVEL.N.A	0.067511155	0.067511155
SOC.Computer.and.Mathematical.Occupations	-0.05506503	0.05506503
SOC.Construction.and.Extraction.Occupations	0.052601383	0.052601383
NAICS.Construction	0.050930308	0.050930308
H1B_DEPENDENT.Y	-0.048300945	0.048300945
SOC.Production.Occupations	0.046725022	0.046725022
PW_SOURCE.OES	-0.04440122	0.04440122
SUPPORT_H1B.Y	-0.044289259	0.044289259
SUPPORT_H1B.N.A	0.042716765	0.042716765
PW_SOURCE.Other	0.041959934	0.041959934
NAICS.Professional..Scientific..and.Technical.Services	-0.037676238	0.037676238
SOC.Building.and.Grounds.Cleaning.and.Maintenance.Occupations	0.037085948	0.037085948

SOC.Office.and.Administrative.Support.Occupations	0.035438251	0.035438251
PW_SOURCE.SCA	0.033038976	0.033038976
PW_WAGE_LEVEL.Level.II	-0.028498626	0.028498626
PW_SOURCE.DBA	0.028491943	0.028491943
NAICS.Arts..Entertainment..and.Recreation	0.026244101	0.026244101
REMAINING_DAYS	-0.024128342	0.024128342
NEW_EMPLOYMENT	0.02410268	0.02410268
SOC.Legal.Occupations	0.024094551	0.024094551
FULL_TIME_POSITION.Y	-0.02299164	0.02299164
WAGE_RATE_OF_PAY_FROM.YEARLY.	0.022836851	0.022836851
SOC.Personal.Care.and.Service.Occupations	0.021765249	0.021765249
SOC.Healthcare.Support.Occupations	0.020731442	0.020731442
NEW_CONCURRENT_EMPLOYMENT	0.020442023	0.020442023
WORKING_DAYS	0.019931959	0.019931959
SOC.Farming..Fishing..and.Forestry.Occupations	0.019786044	0.019786044
SOC.Transportation.and.Material.Moving.Occupations	0.019786044	0.019786044
SOC.Management.Occupations	0.019327221	0.019327221
SOC.Installation..Maintenance..and.Repair.Occupations	0.018314631	0.018314631

SOC.Education..Training..and.Library.Occupations	0.017917113	0.017917113
NAICS.Agriculture..Forestry..Fishing.and.Hunting	0.017738924	0.017738924
NAICS.Other.Services..except.Public.Administration.	0.016913163	0.016913163
NAICS.Public.Administration	0.015991187	0.015991187
SOC.Arts..Design..Entertainment..Sports..and.Media.Occupations	0.014553801	0.014553801
SOC.Healthcare.Practitioners.and.Technical.Occupations	0.014523267	0.014523267
SOC.Community.and.Social.Service.Occupations	0.01317539	0.01317539
NAICS.Manufacturing	0.013162793	0.013162793
NAICS.Educational.Services	0.012634554	0.012634554
NAICS.Real.Estate.and.Rental.and.Leasing	0.01256425	0.01256425
SOC.Business.and.Financial.Operations.Occupations	0.012374101	0.012374101
PW_WAGE_LEVEL.Level.III	-0.011614172	0.011614172
NAICS.Wholesale.Trade	0.011017035	0.011017035
NAICS.Health.Care.and.Social.Assistance	0.010929078	0.010929078
WILLFUL_VIOLATOR.Y	0.009166449	0.009166449
AGENT_REPRESENTING_EMPLOYER.Y	-0.008132627	0.008132627
EMPLOYER_REGION.West.Region	0.008039216	0.008039216

EMPLOYER_REGION.South.Region	-0.006779713	0.006779713
NAICS.Transportation.and.Warehousing	0.006654729	0.006654729
NAICS.Administrative.and.Support.and.Waste.Management.and.Remediation.Services	0.006459504	0.006459504
SOC.Life..Physical..and.Social.Science.Occupations	0.005589181	0.005589181
NAICS.Information	0.004908372	0.004908372
PW_WAGE_LEVEL.Level.IV	0.004221799	0.004221799
SOC.Sales.and.Related.Occupations	0.00406326	0.00406326
PW_SOURCE_YEAR	-0.003364739	0.003364739
AMENDED_PETITION	-0.002374318	0.002374318
WORKSITE_REGION.South.Region	0.002347434	0.002347434
WORKSITE_REGION.West.Region	-0.002252419	0.002252419
EMPLOYER_REGION.Northeast.Region	0.002218789	0.002218789
NAICS.Utilities	0.002084125	0.002084125
CONTINUED_EMPLOYMENT	-0.001990847	0.001990847
NAICS.Management.of.Companies.and.Enterprises	-0.001743746	0.001743746
NAICS.Mining..Quarrying..and.Oil.and.Gas.Extraction	-0.001159065	0.001159065
TOTAL_WORKERS	-0.001023082	0.001023082
NAICS.Finance.and.Insurance	-0.000960686	0.000960686

CHANGE_PREVIOUS_EMPLOYMENT	0.000544156	0.000544156
NAICS.Retail.Trade	-0.000523958	0.000523958
WORKSITE_REGION.Northeast.Region	-0.000466524	0.000466524
CHANGE_EMPLOYER	-0.000293055	0.000293055

Appendix G: R Code

```
library(caret)
library(mice)
library(dplyr)
library(missForest)
library(doParallel)
library(lattice)
library(ggplot2)
library(iterators)
library(parallel)
library(ipred)
library(pROC)
library(lubridate)
library(ROSE)
library(e1071)


#Read the CSV
H1B <- read.csv("H1B_Disclosure_Clean_1.6.csv", header = T)
str(H1B)
summary(H1B)


H1B$DECISION_DAYS <- NULL

str(H1B)
#Dummy Variable
#H1B.dmy <- dummyVars( "~ ." , H1B, fullRank = T)
#H1B.d <- data.frame(predict(H1B.dmy, newdata = H1B))
#head(H1B)
#str(H1B)
```

```
#Correlation between predictors and dependent variable
#cor(H1B.d, H1B.d$CASE_STATUS)


#str(H1B)
#str(H1B.d)

## Data Splitting for categorical
set.seed(99)
trainIndex <- createDataPartition(H1B$CASE_STATUS, p=.7, list=F)
nrow(trainIndex)
H1B.imbal.train <- H1B[trainIndex,]
H1B.imbal.test <- H1B[-trainIndex,]
str(H1B.imbal.train)
str(H1B.imbal.test)

#checking the class imbalance
table(H1B.imbal.train$CASE_STATUS)

## Create train data split for DV and Predictors
x.H1B.train <- H1B.imbal.train [,-ncol(H1B.imbal.train)]
y.H1B.train <- H1B.imbal.train [, ncol(H1B.imbal.train)]

## Create test data split for DV and Predictors
y.H1B.test <- H1B.imbal.test$CASE_STATUS
x.H1B.test <- H1B.imbal.test [,-ncol(H1B.imbal.test)]
head(y.H1B.test)
colnames(x.H1B.test)
nrow(x.H1B.test)

# SMOTE Imbalancing
library(grid)
library(DMwR)
set.seed(99)
nrow(H1B.imbal.train)
H1B.smote.train <- SMOTE(CASE_STATUS ~ ., H1B.imbal.train, perc.over = 5020, perc.under=102)
table(H1B.smote.train$CASE_STATUS)
prop.table(table(H1B.smote.train$CASE_STATUS))

## Create test data split for DV and Predictors
y.H1B.smote.train <- H1B.smote.train$CASE_STATUS
x.H1B.smote.train <- H1B.smote.train [,-ncol(H1B.smote.train)]
head(y.H1B.smote.train)
colnames(x.H1B.smote.train)
```

```
#Dummy Variable
H1B.dmy <- dummyVars( "~ ." , H1B[,-24], fullRank = T)
H1B.d <- data.frame(predict(H1B.dmy, newdata = H1B[,-24]),H1B$CASE_STATUS)
head(H1B)
str(H1B.d)


# Data Exploration and ploting pair
library(car)
library(hexbin)

subset <- data.frame(H1B$NEW_EMPLOYMENT, H1B$PREVAILING_WAGE..YEARLY.,
H1B$SOC,
                H1B$NAICS, H1B$CASE_STATUS)

bitmap("Pairs.tiff", height = 4, width = 4, units = 'in', type="tifflzw", res=300)
pairs(subset)
dev.off()


#####Data splitting for numerical
set.seed(99)
trainIndex.d <- createDataPartition(H1B.d$H1B.CASE_STATUS, p=.7, list=F)
nrow(trainIndex.d)
H1B.imbal.train.d <- H1B.d[trainIndex.d,]
H1B.imbal.test.d <- H1B.d[-trainIndex.d,]
str(H1B.imbal.train.d)
str(H1B.imbal.test.d)
colnames(H1B.imbal.train.d)
colnames(H1B.imbal.test.d)

#checking the class imbalance
table(H1B.imbal.train.d$H1B.CASE_STATUS)

## Create train data split for DV and Predictors
x.H1B.train.d <- H1B.imbal.train.d [,-ncol(H1B.imbal.train.d)]
y.H1B.train.d <- H1B.imbal.train.d [, ncol(H1B.imbal.train.d)]
str(x.H1B.train.d)
str(y.H1B.train.d)
colnames(x.H1B.train.d)
names(y.H1B.train.d)
```

```
## Create test data split for DV and Predictors
y.H1B.test.d <- H1B.imbal.test.d [, ncol(H1B.imbal.train.d)]
x.H1B.test.d <- H1B.imbal.test.d [,-ncol(H1B.imbal.test.d)]
head(y.H1B.test.d)
colnames(x.H1B.test.d)


# SMOTE Imbalancing
library(DMwR)
library(grid)
set.seed(99)
nrow(H1B.imbal.train.d)
H1B.smote.train.d <- SMOTE(H1B.CASE_STATUS ~ ., H1B.imbal.train.d, perc.over = 5020,
perc.under=102)
table(H1B.smote.train.d$H1B.CASE_STATUS)
prop.table(table(H1B.smote.train.d$H1B.CASE_STATUS))

## Create test data split for DV and Predictors
y.H1B.smote.train.d <- H1B.smote.train.d$H1B.CASE_STATUS
x.H1B.smote.train.d <- H1B.smote.train.d [,-ncol(H1B.smote.train.d)]
head(y.H1B.smote.train.d)
colnames(x.H1B.smote.train.d)

###### Principal Component Analysis (PCA)
H1B.smote.prin_comp <- prcomp(x.H1B.smote.train.d, scale. = T)
names(H1B.smote.prin_comp)
H1B.smote.prin_comp$center
biplot(H1B.smote.prin_comp, scale = 0)

#compute standard deviation of each principal component
H1B.smote.std_dev <- H1B.smote.prin_comp$sdev

#compute variance
H1B.smote.pr_var <- H1B.smote.std_dev^2

#check variance of first 10 components
H1B.smote.pr_var[1:10]

#proportion of variance explained
H1B.smote.prop_varex <- H1B.smote.pr_var/sum(H1B.smote.pr_var)
H1B.smote.prop_varex[1:20]

#screen plot
plot(H1B.smote.prop_varex, xlab = "Principal Component",
```

```
    ylab = "Proportion of Variance Explained",
    type = "b")
#cumulative scree plot
plot(cumsum(H1B.smote.prop_varex), xlab = "Principal Component",
    ylab = "Cumulative Proportion of Variance Explained",
    type = "b")

#add a training set with principal components
H1B.smote.pcr.train <- data.frame(CASE_STATUS = y.H1B.smote.train.d, H1B.smote.prin_comp$x)

#we are interested in first 30 PCAs
H1B.smote.pcr.train <- H1B.smote.pcr.train[,1:51]

y.H1B.smote.pcr.train <- H1B.smote.pcr.train[, 1]
x.H1B.smote.pcr.train <- H1B.smote.pcr.train[, -1]

#transform test into PCA
H1B.smote.pcr.test <- predict(H1B.smote.prin_comp, newdata = H1B.imbal.test.d)
H1B.smote.pcr.test <- as.data.frame(H1B.smote.pcr.test)

#select the first 50 components
H1B.smote.pcr.test <- H1B.smote.pcr.test[,1:50]

## to run parallel with 4 cores
cl <- makeCluster(4)
registerDoParallel(cl)

#some parameters to control the sampling during parameter tuning and testing
#10 fold crossvalidation, using 10-folds instead of 10 to reduce computation time in class demo, use 10
and with more computation to spare use
#repeated cv
set.seed(99)
ctrl <- trainControl(method="cv", number=10,
                     classProbs=TRUE,
                     #function used to measure performance
                     summaryFunction = twoClassSummary, #multiClassSummary for non binary
                     allowParallel =  TRUE)


#####Logistic without PCA
set.seed(99)
m.H1B.smote.glm <- readRDS("m.H1B.smote.glm.rdata")
m.H1B.smote.glm
getTrainPerf(m.H1B.smote.glm)
```

```
varImp(m.H1B.smote.glm)

m.H1B.smote.glm

#Making prediction on test data
p.H1B.smote.glm <- predict(m.H1B.smote.glm, x.H1B.test)
p.H1B.smote.glm.prob <- predict(m.H1B.smote.glm, x.H1B.test, type="prob")

confusionMatrix(p.H1B.smote.glm,y.H1B.test) #calc accuracies with confuction matrix on test set

p.H1B.smote.glm.newthresh <- factor(ifelse(p.H1B.smote.glm.prob[[1]]>0.45, "CERTIFIED",
"DENIED"))
confusionMatrix(p.H1B.smote.glm.newthresh, y.H1B.test)

test.H1B.smote.glm.roc<- roc(response= y.H1B.test, predictor= p.H1B.smote.glm.prob[[1]])
plot(test.H1B.smote.glm.roc)

newthresh<- coords(test.H1B.smote.glm.roc, x="best", best.method="closest.topleft")

auc(test.H1B.smote.glm.roc)

#Logistic with PCA
set.seed(99)
m.H1B.smote.pca.glm <- train(y=y.H1B.smote.pcr.train, x=x.H1B.smote.pcr.train,
                        method = "glm",
                        metric = "ROC", tuneLength = 7,
                        trControl = ctrl)

m.H1B.smote.pca.glm
getTrainPerf(m.H1B.smote.pca.glm)

varImp(m.H1B.smote.pca.glm)
#Making prediction on test data
p.H1B.smote.pca.glm <- predict(m.H1B.smote.pca.glm,H1B.smote.pcr.test)
p.H1B.smote.pca.glm.prob <- predict(m.H1B.smote.pca.glm, H1B.smote.pcr.test, type="prob")

confusionMatrix(p.H1B.smote.pca.glm,y.H1B.test) #calc accuracies with confuction matrix on test set

p.H1B.smote.pca.glm.newthresh <- factor(ifelse(p.H1B.smote.pca.glm.prob[[1]]>0.40, "CERTIFIED",
"DENIED"))
p.H1B.smote.pca.glm.newthresh
confusionMatrix(p.H1B.smote.pca.glm.newthresh, y.H1B.test)

test.H1B.smote.pca.glm.roc<- roc(response= y.H1B.test, predictor= p.H1B.smote.pca.glm.prob[[1]])
```

```
plot(test.H1B.smote.pca.glm.roc)

newthresh<- coords(test.H1B.smote.pca.glm.roc, x="best", best.method="closest.topleft")
plot(p.H1B.smote.pca.glm.newthresh)
auc(test.H1B.smote.pca.glm.roc)


######Linear Discriminate Analysis without PCA
m.H1B.smote.lda <- readRDS("m.H1B.smote.lda.rdata")
m.H1B.smote.lda
getTrainPerf(m.H1B.smote.lda)
varImp(m.H1B.smote.lda)

p.H1B.smote.lda <- predict(m.H1B.smote.lda,x.H1B.test.d)
p.H1B.smote.lda.prob <- predict(m.H1B.smote.lda, x.H1B.test.d, type="prob")

confusionMatrix(p.H1B.smote.lda,y.H1B.test.d) #calc accuracies with confuction matrix on test set

p.H1B.smote.lda.newthresh <- factor(ifelse(p.H1B.smote.lda.prob[[1]]>0.40, "CERTIFIED",
"DENIED"))
levels(p.H1B.smote.lda.newthresh)
confusionMatrix(p.H1B.smote.lda.newthresh, y.H1B.test.d)

test.H1B.smote.lda.roc<- roc(response= y.H1B.test.d, predictor= p.H1B.smote.lda.prob[[1]])
plot(test.H1B.smote.lda.roc)

newthresh<- coords(test.H1B.smote.lda.roc, x="best", best.method="closest.topleft")

p.H1B.smote.lda.prob [[1]]
levels(p.H1B.smote.lda.newthresh)

auc(test.H1B.smote.lda.roc)


#Linear Discriminate Analysis with PCA
m.H1B.smote.pca.lda <- readRDS("m.H1B.smote.pca.lda.rdata")
m.H1B.smote.pca.lda
getTrainPerf(m.H1B.smote.pca.lda)
varImp(m.H1B.smote.pca.lda)

#Making prediction on test data
p.H1B.smote.pca.lda <- predict(m.H1B.smote.pca.lda,H1B.smote.pcr.test)
p.H1B.smote.pca.lda.prob <- predict(m.H1B.smote.pca.lda, H1B.smote.pcr.test, type="prob")
```

```
confusionMatrix(p.H1B.smote.pca.lda,y.H1B.test) #calc accuracies with confuction matrix on test set

p.H1B.smote.pca.lda.newthresh <- factor(ifelse(p.H1B.smote.pca.lda.prob[[1]]>0.45, "CERTIFIED",
"DENIED"))
p.H1B.smote.pca.lda.newthresh
confusionMatrix(p.H1B.smote.pca.lda.newthresh, y.H1B.test)

test.H1B.smote.pca.lda.roc<- roc(response= y.H1B.test, predictor= p.H1B.smote.pca.lda.prob[[1]])
plot(test.H1B.smote.pca.lda.roc)

newthresh<- coords(test.H1B.smote.pca.lda.roc, x="best", best.method="closest.topleft")
plot(p.H1B.smote.pca.lda.newthresh)
auc(test.H1B.smote.pca.lda.roc)


##Decision Tree without PCA
m.H1B.smote.rpart <- readRDS("m.H1B.smote.rpart.rdata")
m.H1B.smote.rpart

getTrainPerf(m.H1B.smote.rpart)
varImp(m.H1B.smote.rpart)

#Making prediction on test data
p.H1B.smote.rpart <- predict(m.H1B.smote.rpart,x.H1B.test)
p.H1B.smote.rpart.prob <- predict(m.H1B.smote.rpart, x.H1B.test, type="prob")

confusionMatrix(p.H1B.smote.rpart,y.H1B.test) #calc accuracies with confuction matrix on test set

p.H1B.smote.rpart.newthresh <- factor(ifelse(p.H1B.smote.rpart.prob[[1]]>0.69, "CERTIFIED",
"DENIED"))
p.H1B.smote.rpart.newthresh
confusionMatrix(p.H1B.smote.rpart.newthresh, y.H1B.test)

test.H1B.smote.rpart.roc<- roc(response= y.H1B.test, predictor= p.H1B.smote.rpart.prob[[1]])
plot(test.H1B.smote.rpart.roc)

newthresh<- coords(test.H1B.smote.rpart.roc, x="best", best.method="closest.topleft")
plot(p.H1B.smote.rpart.newthresh)
auc(test.H1B.smote.rpart.roc)

#Plotting DT using rpart
library(rpart)
set.seed(99)
smote.fit <- rpart(y.H1B.smote.train~. ,data=x.H1B.smote.train,
```

```
            method="class",
            control=rpart.control(minsplit=1),
            parms=list(split='information'))
plot(smote.fit)
library(rpart.plot)
rpart.plot(smote.fit, type=4, extra=2,clip.right.labs=FALSE, varlen=0, faclen=3)
rpart.plot(smote.fit)


printcp(smote.fit) #display crossvalidated error for each tree size
plotcp(smote.fit) #plot cv error


#we can grab this from the plotcp table automatically with
Smote.opt.cp <- smote.fit$cptable[which.min(smote.fit$cptable[,"xerror"]),"CP"]

#lets prune the tree
Smote.fit.pruned <- prune(smote.fit,cp=0.025553)

#lets review the final tree
rpart.plot(Smote.fit.pruned)

##Decision Tree with PCA
m.H1B.smote.pca.rpart <- readRDS("m.H1B.smote.pca.rpart.rdata")
m.H1B.smote.pca.rpart
getTrainPerf(m.H1B.smote.pca.rpart)
varImp(m.H1B.smote.pca.rpart)

#Making prediction on test data
p.H1B.smote.pca.rpart <- predict(m.H1B.smote.pca.rpart,H1B.smote.pcr.test)
p.H1B.smote.pca.rpart.prob <- predict(m.H1B.smote.pca.rpart, H1B.smote.pcr.test, type="prob")

confusionMatrix(p.H1B.smote.pca.rpart,y.H1B.test) #calc accuracies with confuction matrix on test set

p.H1B.smote.pca.rpart.newthresh <- factor(ifelse(p.H1B.smote.pca.rpart.prob[[1]]>0.33, "CERTIFIED",
"DENIED"))
p.H1B.smote.pca.rpart.newthresh
confusionMatrix(p.H1B.smote.pca.rpart.newthresh, y.H1B.test)

test.H1B.smote.pca.rpart.roc<- roc(response= y.H1B.test, predictor= p.H1B.smote.pca.rpart.prob[[1]])
plot(test.H1B.smote.pca.rpart.roc)

newthresh<- coords(test.H1B.smote.pca.rpart.roc, x="best", best.method="closest.topleft")
plot(p.H1B.smote.pca.rpart.newthresh)
```

```
auc(test.H1B.smote.pca.rpart.roc)

##Naive Bayes without PCA
library(klaR)
m.H1B.smote.nb <- readRDS("m.H1B.smote.nb.rdata")
m.H1B.smote.nb
getTrainPerf(m.H1B.smote.nb)
varImp(m.H1B.smote.nb)

#Making prediction on test data
p.H1B.smote.nb <- predict(m.H1B.smote.nb,x.H1B.test)
p.H1B.smote.nb.prob <- predict(m.H1B.smote.nb, x.H1B.test, type="prob")

confusionMatrix(p.H1B.smote.nb,y.H1B.test) #calc accuracies with confuction matrix on test set

p.H1B.smote.nb.newthresh <- factor(ifelse(p.H1B.smote.nb.prob[[1]]>0.30, "CERTIFIED", "DENIED"))
p.H1B.smote.nb.newthresh
confusionMatrix(p.H1B.smote.nb.newthresh, y.H1B.test)

test.H1B.smote.nb.roc<- roc(response= y.H1B.test, predictor= p.H1B.smote.nb.prob[[1]])
plot(test.H1B.smote.nb.roc)

newthresh<- coords(test.H1B.smote.nb.roc, x="best", best.method="closest.topleft")
plot(p.H1B.smote.nb.newthresh)
auc(test.H1B.smote.nb.roc)


##Naive Bayes with PCA
m.H1B.smote.pca.nb <- readRDS("m.H1B.smote.pca.nb.rdata")
m.H1B.smote.pca.nb
getTrainPerf(m.H1B.smote.pca.nb)
varImp(m.H1B.smote.pca.nb)

#Making prediction on test data
p.H1B.smote.pca.nb <- predict(m.H1B.smote.pca.nb,H1B.smote.pcr.test)
p.H1B.smote.pca.nb.prob <- predict(m.H1B.smote.pca.nb, H1B.smote.pcr.test, type="prob")

confusionMatrix(p.H1B.smote.pca.nb,y.H1B.test) #calc accuracies with confuction matrix on test set

p.H1B.smote.pca.nb.newthresh <- factor(ifelse(p.H1B.smote.pca.nb.prob[[1]]>0.35, "CERTIFIED",
"DENIED"))
p.H1B.smote.pca.nb.newthresh
confusionMatrix(p.H1B.smote.pca.nb.newthresh, y.H1B.test)
```

```
test.H1B.smote.pca.nb.roc<- roc(response= y.H1B.test, predictor= p.H1B.smote.pca.nb.prob[[1]])
plot(test.H1B.smote.pca.nb.roc)

newthresh<- coords(test.H1B.smote.pca.nb.roc, x="best", best.method="closest.topleft")
plot(p.H1B.smote.pca.nb.newthresh)
auc(test.H1B.smote.pca.nb.roc)

####GAM without PCA
m.H1B.smote.gam <- library(splines)
library(gam)
set.seed(99)
m.H1B.smote.gam <- readRDS("m.H1B.smote.gam.rdata")
#saveRDS(m.H1B.smote.gam, file = "m.H1B.smote.gam.rdata")
m.H1B.smote.gam
getTrainPerf(m.H1B.smote.gam)
varImp(m.H1B.smote.gam)

p.H1B.smote.gam <- predict(m.H1B.smote.gam,x.H1B.test.d)
p.H1B.smote.gam.prob <- predict(m.H1B.smote.gam, x.H1B.test.d, type="prob")

confusionMatrix(p.H1B.smote.gam,y.H1B.test.d) #calc accuracies with confuction matrix on test set

p.H1B.smote.gam.newthresh <- factor(ifelse(p.H1B.smote.gam.prob[[1]]>0.60, "CERTIFIED",
"DENIED"))
confusionMatrix(p.H1B.smote.gam.newthresh, y.H1B.test.d)

test.H1B.smote.gam.roc<- roc(response= y.H1B.test.d, predictor= p.H1B.smote.gam.prob[[1]])
plot(test.H1B.smote.gam.roc)

newthresh<- coords(test.H1B.smote.gam.roc, x="best", best.method="closest.topleft")

p.H1B.smote.gam.prob[[1]]
levels(p.H1B.smote.gam.newthresh)
plot(p.H1B.smote.gam)

auc(test.H1B.smote.gam.roc)

##GAM with PCA
m.H1B.smote.pca.gam <- readRDS("m.H1B.smote.pca.gam.rdata")
m.H1B.smote.pca.gam
getTrainPerf(m.H1B.smote.pca.gam)
varImp(m.H1B.smote.pca.gam)

#Making prediction on test data
```

```
p.H1B.smote.pca.gam <- predict(m.H1B.smote.pca.gam,H1B.smote.pcr.test)
p.H1B.smote.pca.gam.prob <- predict(m.H1B.smote.pca.gam, H1B.smote.pcr.test, type="prob")

confusionMatrix(p.H1B.smote.pca.gam,y.H1B.test) #calc accuracies with confuction matrix on test set

p.H1B.smote.pca.gam.newthresh <- factor(ifelse(p.H1B.smote.pca.gam.prob[[1]]>0.40, "CERTIFIED",
"DENIED"))
confusionMatrix(p.H1B.smote.pca.gam.newthresh, y.H1B.test)

test.H1B.smote.pca.gam.roc<- roc(response= y.H1B.test, predictor= p.H1B.smote.pca.gam.prob[[1]])
plot(test.H1B.smote.pca.gam.roc)

newthresh<- coords(test.H1B.smote.pca.gam.roc, x="best", best.method="closest.topleft")
plot(p.H1B.smote.pca.gam.newthresh)
auc(test.H1B.smote.pca.gam.roc)

##PLS without PCA
m.H1B.smote.pls <- readRDS("m.H1B.smote.pls.rdata")
m.H1B.smote.pls
getTrainPerf(m.H1B.smote.pls)
varImp(m.H1B.smote.pls)

p.H1B.smote.pls <- predict(m.H1B.smote.pls,x.H1B.test.d)
p.H1B.smote.pls.prob <- predict(m.H1B.smote.pls, x.H1B.test.d, type="prob")

confusionMatrix(p.H1B.smote.pls,y.H1B.test.d) #calc accuracies with confuction matrix on test set

p.H1B.smote.pls.newthresh <- factor(ifelse(p.H1B.smote.pls.prob[[1]]>0.45, "CERTIFIED",
"DENIED"))
levels(p.smote.pls.newthresh)
confusionMatrix(p.H1B.smote.pls.newthresh, y.H1B.test.d)

test.smote.pls.roc<- roc(response= y.H1B.test.d, predictor= p.H1B.smote.pls.prob[[1]])
plot(test.smote.pls.roc)

newthresh<- coords(test.smote.pls.roc, x="best", best.method="closest.topleft")

p.smote.pls.prob[[1]]
levels(p.smote.pls.newthresh)
plot(p.smote.pls)

auc(test.smote.pls.roc)

##PLS with PCA
```

```
set.seed(99)
m.H1B.smote.pca.pls <- readRDS("m.H1B.smote.pca.pls.rdata")

saveRDS(m.H1B.smote.pca.pls, file = "m.H1B.smote.pca.pls.rdata")
m.H1B.smote.pca.pls
getTrainPerf(m.H1B.smote.pca.pls)
varImp(m.H1B.smote.pca.pls)

#Making prediction on test data
p.H1B.smote.pca.pls <- predict(m.H1B.smote.pca.pls,H1B.smote.pcr.test)
p.H1B.smote.pca.pls.prob <- predict(m.H1B.smote.pca.pls, H1B.smote.pcr.test, type="prob")

confusionMatrix(p.H1B.smote.pca.pls, y.H1B.test) #calc accuracies with confuction matrix on test set

p.H1B.smote.pca.pls.newthresh <- factor(ifelse(p.H1B.smote.pca.pls.prob[[1]]>0.45, "CERTIFIED",
"DENIED"))
confusionMatrix(p.H1B.smote.pca.pls.newthresh, y.H1B.test)

test.H1B.smote.pca.pls.roc<- roc(response= y.H1B.test, predictor= p.H1B.smote.pca.pls.prob[[1]])
plot(test.H1B.smote.pca.pls.roc)

newthresh<- coords(test.H1B.smote.pca.pls.roc, x="best", best.method="closest.topleft")
plot(p.H1B.smote.pca.pls.newthresh)
auc(test.H1B.smote.pca.pls.roc)


##Glmnet without PCA
m.H1B.smote.glmnet <- readRDS("m.H1B.smote.glmnet.rdata")
m.H1B.smote.glmnet
getTrainPerf(m.H1B.smote.glmnet)
varImp(m.H1B.smote.glmnet)
#Making prediction on test data
p.H1B.smote.glmnet <- predict(m.H1B.smote.glmnet,x.H1B.test.d)
p.H1B.smote.glmnet.prob <- predict(m.H1B.smote.glmnet, x.H1B.test.d, type="prob")

confusionMatrix(p.H1B.smote.glmnet,y.H1B.test) #calc accuracies with confuction matrix on test set

p.H1B.smote.glmnet.newthresh <- factor(ifelse(p.H1B.smote.glmnet.prob[[1]]>0.40, "CERTIFIED",
"DENIED"))
p.H1B.smote.glmnet.newthresh
confusionMatrix(p.H1B.smote.glmnet.newthresh, y.H1B.test)

test.H1B.smote.glmnet.roc<- roc(response= y.H1B.test, predictor= p.H1B.smote.glmnet.prob[[1]])
plot(test.H1B.smote.glmnet.roc)
```

```
newthresh<- coords(test.H1B.smote.glmnet.roc, x="best", best.method="closest.topleft")
plot(p.H1B.smote.glmnet.newthresh)
auc(test.H1B.smote.glmnet.roc)

##Glmnet with PCA
m.H1B.smote.pca.glmnet <- readRDS("m.H1B.smote.pca.glm.rdata")
m.H1B.smote.pca.glmnet
getTrainPerf(m.H1B.smote.pca.glmnet)
varImp(m.H1B.smote.pca.glmnet)

#Making prediction on test data
p.H1B.smote.pca.glmnet <- predict(m.H1B.smote.pca.glmnet,H1B.smote.pcr.test)
p.H1B.smote.pca.glmnet.prob <- predict(m.H1B.smote.pca.glmnet, H1B.smote.pcr.test, type="prob")

confusionMatrix(p.H1B.smote.pca.glmnet,y.H1B.test) #calc accuracies with confuction matrix on test set

p.H1B.smote.pca.glmnet.newthresh <- factor(ifelse(p.H1B.smote.pca.glmnet.prob[[1]]>0.42,
"CERTIFIED", "DENIED"))
confusionMatrix(p.H1B.smote.pca.glmnet.newthresh, y.H1B.test)

test.H1B.smote.pca.glmnet.roc<- roc(response= y.H1B.test, predictor=
p.H1B.smote.pca.glmnet.prob[[1]])
plot(test.H1B.smote.pca.glmnet.roc)

newthresh<- coords(test.H1B.smote.pca.glmnet.roc, x="best", best.method="closest.topleft")
plot(p.H1B.smote.pca.glmnet.newthresh)
auc(test.H1B.smote.pca.glmnet.roc)


##Random Forest without PCA
m.H1B.smote.rf <- readRDS("m.H1B.smote.rf.rdata")
m.H1B.smote.rf

getTrainPerf(m.H1B.smote.rf)
varImp(m.H1B.smote.rf)
plot(m.H1B.smote.rf)

p.H1B.smote.rf <- predict(m.H1B.smote.rf ,x.H1B.test)
plot(p.H1B.smote.rf)
confusionMatrix(p.H1B.smote.rf ,y.H1B.test)

p.H1B.smote.rf.prob<- predict(m.H1B.smote.rf , x.H1B.test ,type="prob")
p.H1B.smote.rf.newthresh <- factor(ifelse(p.H1B.smote.rf.prob[[1]]>0.87, "CERTIFIED", "DENIED"))
```

```
p.H1B.smote.rf.newthresh
confusionMatrix(p.H1B.smote.rf.newthresh, y.H1B.test )

test.H1B.smote.rf.roc<- roc(response= y.H1B.test, predictor= p.H1B.smote.rf.prob [[1]])
plot(test.H1B.smote.rf.roc)

p.H1B.smote.rf.prob [[1]]
levels(p.H1B.smote.rf.newthresh)

auc(test.H1B.smote.rf.roc)

##Random Forest with PCA
m.H1B.smote.pca.rf <- readRDS("m.H1B.smote.pca.rf.rdata")
m.H1B.smote.pca.rf
getTrainPerf(m.H1B.smote.pca.rf)
varImp(m.H1B.smote.pca.rf)

#Making prediction on test data
p.H1B.smote.pca.rf <- predict(m.H1B.smote.pca.rf,H1B.smote.pcr.test)
p.H1B.smote.pca.rf.prob <- predict(m.H1B.smote.pca.rf, H1B.smote.pcr.test, type="prob")

confusionMatrix(p.H1B.smote.pca.rf,y.H1B.test) #calc accuracies with confuction matrix on test set

p.H1B.smote.pca.rf.newthresh <- factor(ifelse(p.H1B.smote.pca.rf.prob[[1]]>0.85, "CERTIFIED",
"DENIED"))
confusionMatrix(p.H1B.smote.pca.rf.newthresh, y.H1B.test)

test.H1B.smote.pca.rf.roc<- roc(response= y.H1B.test, predictor= p.H1B.smote.pca.rf.prob[[1]])
plot(test.H1B.smote.pca.rf.roc)

newthresh<- coords(test.H1B.smote.pca.rf.roc, x="best", best.method="closest.topleft")
plot(p.H1B.smote.pca.rf.newthresh)
auc(test.H1B.smote.pca.rf.roc)


##Bagging without PCA
library(ipred)
set.seed(99)
m.H1B.smote.bag <- readRDS("m.H1B.smote.bag.rdata")
#saveRDS(m.H1B.smote.bag, file = "m.H1B.smote.bag.rdata")
m.H1B.smote.bag

getTrainPerf(m.H1B.smote.bag)
varImp(m.H1B.smote.bag)
```

```
plot(m.H1B.smote.bag)

p.H1B.smote.bag<- predict(m.H1B.smote.bag,x.H1B.test)
confusionMatrix(p.H1B.smote.bag,y.H1B.test)

p.H1B.smote.bag.prob <- predict(m.H1B.smote.bag,x.H1B.test,type="prob")
p.H1B.smote.bag.newthresh <- factor(ifelse(p.H1B.smote.bag.prob[[1]]>0.70, "CERTIFIED",
"DENIED"))
p.H1B.smote.bag.newthresh
confusionMatrix(p.H1B.smote.bag.newthresh, y.H1B.test)

test.H1B.smote.bag.roc<- roc(response= y.H1B.test, predictor=p.H1B.smote.bag.prob[[1]])
plot(test.H1B.smote.bag.roc)

auc(test.H1B.smote.bag.roc)

##Bagging with PCA
set.seed(99)
m.H1B.smote.pca.bag <- readRDS("m.H1B.smote.pca.bag.rdata")
#saveRDS(m.H1B.smote.pca.bag, file = "m.H1B.smote.pca.bag.rdata")
m.H1B.smote.pca.bag
getTrainPerf(m.H1B.smote.pca.bag)
varImp(m.H1B.smote.pca.bag)

#Making prediction on test data
p.H1B.smote.pca.bag <- predict(m.H1B.smote.pca.bag,H1B.smote.pcr.test)
p.H1B.smote.pca.bag.prob <- predict(m.H1B.smote.pca.bag, H1B.smote.pcr.test, type="prob")

confusionMatrix(p.H1B.smote.pca.bag,y.H1B.test) #calc accuracies with confuction matrix on test set

p.H1B.smote.pca.bag.newthresh <- factor(ifelse(p.H1B.smote.pca.bag.prob[[1]]>0.70, "CERTIFIED",
"DENIED"))
p.H1B.smote.pca.bag.newthresh
confusionMatrix(p.H1B.smote.pca.bag.newthresh, y.H1B.test)

test.H1B.smote.pca.bag.roc<- roc(response= y.H1B.test, predictor= p.H1B.smote.pca.bag.prob[[1]])
plot(test.H1B.smote.pca.bag.roc)

newthresh<- coords(test.H1B.smote.pca.bag.roc, x="best", best.method="closest.topleft")

auc(test.H1B.smote.pca.bag.roc)

##Boosting without PCA
m.H1B.smote.ada <- readRDS("m.H1B.smote.ada.rdata")
```

```
m.H1B.smote.ada
getTrainPerf(m.H1B.smote.ada)
varImp(m.H1B.smote.ada)

#Making prediction on test data
p.H1B.smote.ada <- predict(m.H1B.smote.ada,x.H1B.test)
p.H1B.smote.ada.prob <- predict(m.H1B.smote.ada, x.H1B.test, type="prob")

confusionMatrix(p.H1B.smote.ada,y.H1B.test) #calc accuracies with confuction matrix on test set

p.H1B.smote.ada.newthresh <- factor(ifelse(p.H1B.smote.ada.prob[[1]]>0.90, "CERTIFIED",
"DENIED"))
confusionMatrix(p.H1B.smote.ada.newthresh, y.H1B.test)

test.H1B.smote.ada.roc<- roc(response= y.H1B.test, predictor= p.H1B.smote.ada.prob[[1]])
plot(test.H1B.smote.ada.roc)

newthresh<- coords(test.H1B.smote.ada.roc, x="best", best.method="closest.topleft")
plot(p.H1B.smote.ada.newthresh)
auc(test.H1B.smote.ada.roc)

##Boosting with PCA
m.H1B.smote.pca.ada <- readRDS("m.H1B.smote.pca.ada.rdata")
m.H1B.smote.pca.ada
getTrainPerf(m.H1B.smote.pca.ada)
varImp(m.H1B.smote.pca.ada)

#Making prediction on test data
p.H1B.smote.pca.ada <- predict(m.H1B.smote.pca.ada,H1B.smote.pcr.test)
p.H1B.smote.pca.ada.prob <- predict(m.H1B.smote.pca.ada, H1B.smote.pcr.test, type="prob")

confusionMatrix(p.H1B.smote.pca.ada,y.H1B.test) #calc accuracies with confuction matrix on test set

p.H1B.smote.pca.ada.newthresh <- factor(ifelse(p.H1B.smote.pca.ada.prob[[1]]>0.90, "CERTIFIED",
"DENIED"))
confusionMatrix(p.H1B.smote.pca.ada.newthresh, y.H1B.test)

test.H1B.pca.smote.ada.roc<- roc(response= y.H1B.test, predictor= p.H1B.smote.pca.ada.prob[[1]])
plot(test.H1B.pca.smote.ada.roc)

newthresh<- coords(test.H1B.pca.smote.ada.roc, x="best", best.method="closest.topleft")
plot(p.H1B.smote.pca.ada.newthresh)
auc(test.H1B.pca.smote.ada.roc)
```

```
#compare training performance
#create list of cross validation runs (resamples) for without PCA models
rValues <- resamples(list(Logit=m.H1B.smote.glm, LDA=m.H1B.smote.lda, Rpart=m.H1B.smote.rpart,
                    Naivebayes=m.H1B.smote.nb, GAM=m.H1B.smote.gam, PLS=m.H1B.smote.pls,
                    Glmnet=m.H1B.smote.glmnet, RandomForest=m.H1B.smote.rf,
                    Bagging=m.H1B.smote.bag, Boosting=m.H1B.smote.ada))


#create list of cross validation runs (resamples) for with PCA models
rValues_PCA <- resamples(list(Logit_PCA=m.H1B.smote.pca.glm, LDA_PCA=m.H1B.smote.pca.lda,
                      Rpart_PCA=m.H1B.smote.pca.rpart, Naivebayes_PCA=m.H1B.smote.pca.nb,
                      GAM_PCA=m.H1B.smote.pca.gam, PLS_PCA=m.H1B.smote.pca.pls,
                      Glmnet_PCA=m.H1B.smote.pca.glmnet, RandomForest_PCA=m.H1B.smote.pca.rf,
                      Bagging_PCA=m.H1B.smote.pca.bag, Boosting_PCA=m.H1B.smote.pca.ada))

##create list of cross validation runs (resamples) for all PCA models
rvalues_All <- resamples(list(Logit=m.H1B.smote.glm, LDA=m.H1B.smote.lda,
Rpart=m.H1B.smote.rpart,
                      Naivebayes=m.H1B.smote.nb, GAM=m.H1B.smote.gam, PLS=m.H1B.smote.pls,
                      Glmnet=m.H1B.smote.glmnet, RandomForest=m.H1B.smote.rf,
                      Bagging=m.H1B.smote.bag, Boosting=m.H1B.smote.ada,
                      Logit_PCA=m.H1B.smote.pca.glm, LDA_PCA=m.H1B.smote.pca.lda,
                      Rpart_PCA=m.H1B.smote.pca.rpart, Naivebayes_PCA=m.H1B.smote.pca.nb,
                      GAM_PCA=m.H1B.smote.pca.gam, PLS_PCA=m.H1B.smote.pca.pls,
                      Glmnet_PCA=m.H1B.smote.pca.glmnet, RandomForest_PCA=m.H1B.smote.pca.rf,
                      Bagging_PCA=m.H1B.smote.pca.bag, Boosting_PCA=m.H1B.smote.pca.ada))

#create plot for without PCA models comparing them
bwplot(rValues, metric="ROC")
bwplot(rValues, metric="Sens") #Sensitvity
bwplot(rValues, metric="Spec")

#create dot plot comparing them
dotplot(rValues, metric="ROC")
dotplot(rValues, metric="Sens") #Sensitvity
dotplot(rValues, metric="Spec")

xyplot(rValues, metric="ROC")
xyplot(rValues, metric="Sens") #Sensitvity
```

```
xyplot(rValues, metric="Spec")

##create plot for with PCA models comparing them
bwplot(rValues_PCA, metric="ROC")
bwplot(rValues_PCA, metric="Sens") #Sensitvity
bwplot(rValues_PCA, metric="Spec")

#create dot plot comparing them
dotplot(rValues_PCA, metric="ROC")
dotplot(rValues_PCA, metric="Sens") #Sensitvity
dotplot(rValues_PCA, metric="Spec")

xyplot(rValues_PCA, metric="ROC")
xyplot(rValues_PCA, metric="Sens") #Sensitvity
xyplot(rValues_PCA, metric="Spec")

#create plot for all the models comparing them
bwplot(rvalues_All, metric="ROC")
bwplot(rvalues_All, metric="Sens") #Sensitvity
bwplot(rvalues_All, metric="Spec")

#create dot plot comparing them
dotplot(rvalues_All, metric="ROC")
dotplot(rvalues_All, metric="Sens") #Sensitvity
dotplot(rvalues_All, metric="Spec")

xyplot(rvalues_All, metric="ROC")
xyplot(rvalues_All, metric="Sens") #Sensitvity
xyplot(rvalues_All, metric="Spec")

summary(rvalues_All)

##build to combined ROC plot with resampled ROC curves
bitmap("ROC w/o PCA.tiff", height = 4, width = 4, units = 'in', type="tifflzw", res=300)

plot(test.H1B.smote.glm.roc, legacy.axes=T, ylim=c(0:1) , main = "ROC of all models w/o PCA")
plot(test.H1B.smote.lda.roc, add=T, col="Blue")
plot(test.H1B.smote.rpart.roc, add=T, col="Green")
plot(test.H1B.smote.nb.roc, add=T, col="Red")
plot(test.smote.pls.roc, add=T, col="Pink")
plot(test.H1B.smote.glmnet.roc, add=T, col="seagreen4")
plot(test.H1B.smote.rf.roc, add=T, col="Orange")
plot(test.H1B.smote.bag.roc, add=T, col="darkmagenta")
plot(test.H1B.smote.ada.roc, add=T, col="coral2")
```

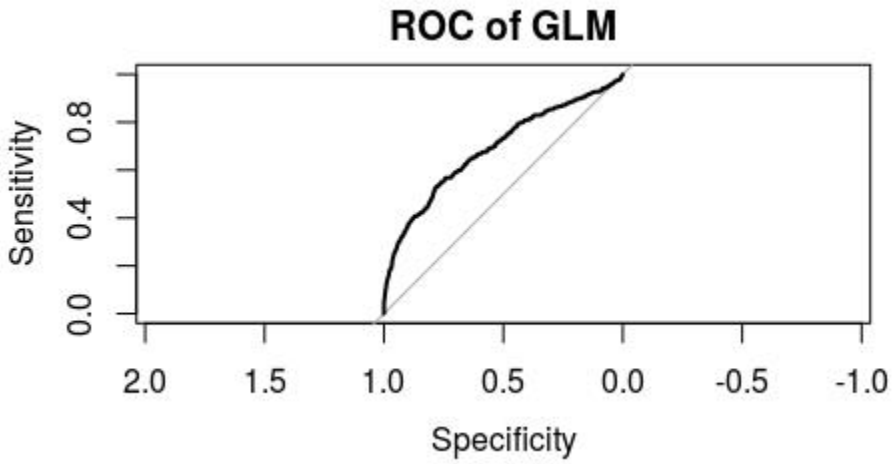
```
plot(test.H1B.smote.gam.roc, add=T, col="cyan")
legend(x=0.1, y=1, legend=c("Glm", "LDA", "DT", "NB", "PLS", "Glmnet", "RF", "Bag", "Boosting",
"GAM"),
      col=c("black","blue","green","red", "pink", "yellow", "Orange", "darkmagenta", "coral2",
"cyan"),lty=1,
      x.intersp = 0.5 , y.intersp = 0.5, xjust = 0.05,  yjust = 1.2)

##build to combined ROC plot of models with PCA with resampled ROC curves
plot(test.H1B.smote.pca.glm.roc, legacy.axes=T, ylim=c(0:1) , main = "ROC of all models with PCA")
plot(test.H1B.smote.pca.lda.roc, add=T, col="Blue" )
plot(test.H1B.smote.pca.rpart.roc, add=T, col="Green")
plot(test.H1B.smote.pca.nb.roc, add=T, col="Red")
plot(test.H1B.smote.pca.pls.roc, add=T, col="Pink")
plot(test.H1B.smote.pca.glmnet.roc, add=T, col="seagreen4")
plot(test.H1B.smote.pca.rf.roc, add=T, col="Orange")
plot(test.H1B.smote.pca.bag.roc, add=T, col="darkmagenta")
plot(test.H1B.pca.smote.ada.roc, add=T, col="coral2")
plot(test.H1B.smote.pca.gam.roc, add=T, col="cyan")
legend(x=0.1, y=1, legend=c("Glm", "LDA", "DT", "NB", "PLS", "Glmnet", "RF", "Bag", "Boosting",
"GAM"),
      col=c("black","blue","green","red", "pink", "yellow", "Orange", "darkmagenta", "coral2",
"cyan"),lty=1,
      x.intersp = 0.5 , y.intersp = 0.5, xjust = 0.05,  yjust = 1.2)

###PRECISION RECALL

# Precision, Recall and F1 Score for GLM without PCA
precision_glm <- posPredValue(p.H1B.smote.glm, y.H1B.test, positive="CERTIFIED")
recall_glm <- sensitivity(p.H1B.smote.glm, y.H1B.test, positive="CERTIFIED")
F1-glm <- (2 * precision * recall) / (precision + recall)

# Precision, Recall and F1 Score for Glm with PCA
precision_pca_glm <- posPredValue(p.H1B.smote.pca.glm, y.H1B.test, positive="CERTIFIED")
recall_pca_glm <- sensitivity(p.H1B.smote.pca.glm, y.H1B.test, positive="CERTIFIED")
F1-pca_glm <- (2 * precision_pca_glm * recall_pca_glm) / (precision_pca_glm + recall_pca_glm)

# Precision, Recall and F1 Score for LDA without PCA
precision_lda <- posPredValue(p.H1B.smote.lda, y.H1B.test, positive="CERTIFIED")
recall_lda <- sensitivity(p.H1B.smote.lda, y.H1B.test, positive="CERTIFIED")
F1-lda <- (2 * precision_lda * recall_lda) / (precision_lda + recall_lda)

# Precision, Recall and F1 Score for LDA with PCA
precision_pca_lda <- posPredValue(p.H1B.smote.pca.lda, y.H1B.test, positive="CERTIFIED")
recall_pca_lda <- sensitivity(p.H1B.smote.pca.lda, y.H1B.test, positive="CERTIFIED")
```

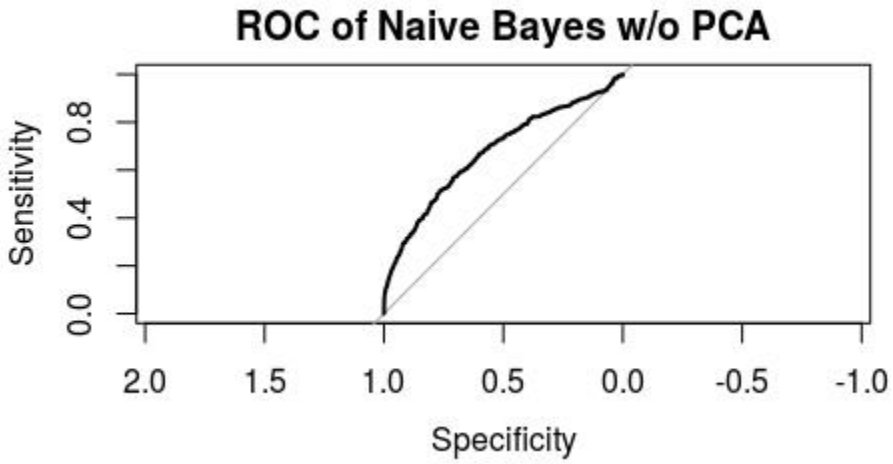
```
F1-pca_lda <- (2 * precision_pca_lda * recall_pca_lda) / (precision_pca_lda + recall_pca_lda)


# Precision, Recall and F1 Score for DT without PCA
precision_rpart <- posPredValue(p.H1B.smote.rpart, y.H1B.test, positive="CERTIFIED")
recall_rpart <- sensitivity(p.H1B.smote.rpart, y.H1B.test, positive="CERTIFIED")
F1-rpart <- (2 * precision_rpart * recall_rpart) / (precision_rpart + recall_rpart)

# Precision, Recall and F1 Score for DT with PCA
precision_pca_rpart <- posPredValue(p.H1B.smote.pca.rpart, y.H1B.test, positive="CERTIFIED")
recall_pca_rpart <- sensitivity(p.H1B.smote.pca.rpart, y.H1B.test, positive="CERTIFIED")
F1-pca_rpart <- (2 * precision_pca_rpart * recall_pca_rpart) / (precision_pca_rpart + recall_pca_rpart)

# Precision, Recall and F1 Score for NB without PCA
precision_nb <- posPredValue(p.H1B.smote.nb, y.H1B.test, positive="CERTIFIED")
recall_nb <- sensitivity(p.H1B.smote.nb, y.H1B.test, positive="CERTIFIED")
F1-nb <- (2 * precision_nb * recall_nb) / (precision_nb + recall_nb)

# Precision, Recall and F1 Score for NB with PCA
precision_pca_nb <- posPredValue(p.H1B.smote.pca.nb, y.H1B.test, positive="CERTIFIED")
recall_pca_nb <- sensitivity(p.H1B.smote.pca.nb, y.H1B.test, positive="CERTIFIED")
F1-pca_nb <- (2 * precision_pca_nb * recall_pca_nb) / (precision_pca_nb + recall_pca_nb)

# Precision, Recall and F1 Score for GAM without PCA
precision_gam <- posPredValue(p.H1B.smote.gam, y.H1B.test, positive="CERTIFIED")
recall_gam <- sensitivity(p.H1B.smote.gam, y.H1B.test, positive="CERTIFIED")
F1-gam <- (2 * precision_gam * recall_gam) / (precision_gam + recall_gam)

# Precision, Recall and F1 Score for GAM with PCA
precision_pca_gam <- posPredValue(p.H1B.smote.pca.gam, y.H1B.test, positive="CERTIFIED")
recall_pca_gam <- sensitivity(p.H1B.smote.pca.gam, y.H1B.test, positive="CERTIFIED")
F1-pca_gam <- (2 * precision_pca_gam * recall_pca_gam) / (precision_pca_gam + recall_pca_gam)

# Precision, Recall and F1 Score for PLS without PCA
precision_pls <- posPredValue(p.H1B.smote.pls, y.H1B.test, positive="CERTIFIED")
recall_pls <- sensitivity(p.H1B.smote.pls, y.H1B.test, positive="CERTIFIED")
F1-pls <- (2 * precision_pls * recall_pls) / (precision_pls + recall_pls)

# Precision, Recall and F1 Score for PLS with PCA
precision_pca_pls <- posPredValue(p.H1B.smote.pca.pls, y.H1B.test, positive="CERTIFIED")
recall_pca_pls <- sensitivity(p.H1B.smote.pca.pls, y.H1B.test, positive="CERTIFIED")
F1-pca_pls <- (2 * precision_pca_pls * recall_pca_pls) / (precision_pca_pls + recall_pca_pls)

# Precision, Recall and F1 Score for GLMNET without PCA
```

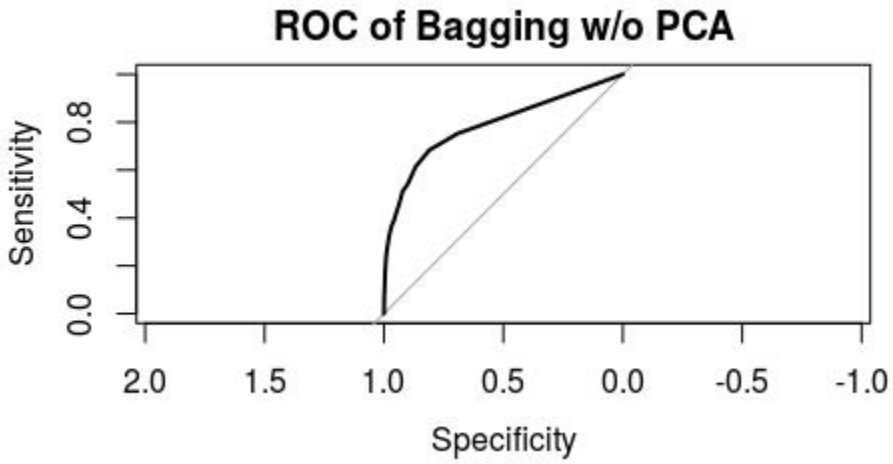
```
precision_glmnet <- posPredValue(p.H1B.smote.glmnet, y.H1B.test, positive="CERTIFIED")
recall_glmnet <- sensitivity(p.H1B.smote.glmnet, y.H1B.test, positive="CERTIFIED")
F1-glmnet <- (2 * precision_glmnet * recall_glmnet) / (precision_glmnet + recall_glmnet)

# Precision, Recall and F1 Score for GLMNET with PCA
precision_pca_glmnet <- posPredValue(p.H1B.smote.pca.glmnet, y.H1B.test, positive="CERTIFIED")
recall_pca_glmnet <- sensitivity(p.H1B.smote.pca.glmnet, y.H1B.test, positive="CERTIFIED")
F1-pca_glmnet <- (2 * precision_pca_glmnet * recall_pca_glmnet) / (precision_pca_glmnet +
recall_pca_glmnet)

# Precision, Recall and F1 Score for Random Forest without PCA
precision_rf <- posPredValue(p.H1B.smote.rf, y.H1B.test, positive="CERTIFIED")
recall_rf <- sensitivity(p.H1B.smote.rf, y.H1B.test, positive="CERTIFIED")
F1-rf <- (2 * precision_rf * recall_rf) / (precision_rf + recall_rf)

# Precision, Recall and F1 Score for Random Forest with PCA
precision_pca_rf <- posPredValue(p.H1B.smote.pca.rf, y.H1B.test, positive="CERTIFIED")
recall_pca_rf <- sensitivity(p.H1B.smote.pca.rf, y.H1B.test, positive="CERTIFIED")
F1-pca_rf <- (2 * precision_pca_rf * recall_pca_rf) / (precision_pca_rf + recall_pca_rf)

# Precision, Recall and F1 Score for Bagging without PCA
precision_bag <- posPredValue(p.H1B.smote.bag, y.H1B.test, positive="CERTIFIED")
recall_bag <- sensitivity(p.H1B.smote.bag, y.H1B.test, positive="CERTIFIED")
F1-bag <- (2 * precision_bag * recall_bag) / (precision_bag + recall_bag)

# Precision, Recall and F1 Score for Bagging with PCA
precision_pca_bag <- posPredValue(p.H1B.smote.pca.bag, y.H1B.test, positive="CERTIFIED")
recall_pca_bag <- sensitivity(p.H1B.smote.pca.bag, y.H1B.test, positive="CERTIFIED")
F1-pca_bag <- (2 * precision_pca_bag * recall_pca_bag) / (precision_pca_bag + recall_pca_bag)

# Precision, Recall and F1 Score for Boosting without PCA
precision_ada <- posPredValue(p.H1B.smote.ada, y.H1B.test, positive="CERTIFIED")
recall_ada <- sensitivity(p.H1B.smote.ada, y.H1B.test, positive="CERTIFIED")
F1-ada <- (2 * precision_ada * recall_ada) / (precision_ada + recall_ada)

# Precision, Recall and F1 Score for Boosting with PCA
precision_pca_ada <- posPredValue(p.H1B.smote.pca.ada, y.H1B.test, positive="CERTIFIED")
recall_pca_ada <- sensitivity(p.H1B.smote.pca.ada, y.H1B.test, positive="CERTIFIED")
F1-pca_ada <- (2 * precision_pca_ada * recall_pca_ada) / (precision_pca_ada + recall_pca_ada)

##Individual ROC
plot(test.H1B.smote.glm.roc, main = "ROC of GLM w/o PCA")
plot(test.H1B.smote.lda.roc, main = "ROC of LDA w/o PCA")
plot(test.H1B.smote.rpart.roc, main = "ROC of Decision Tree w/o PCA")
```

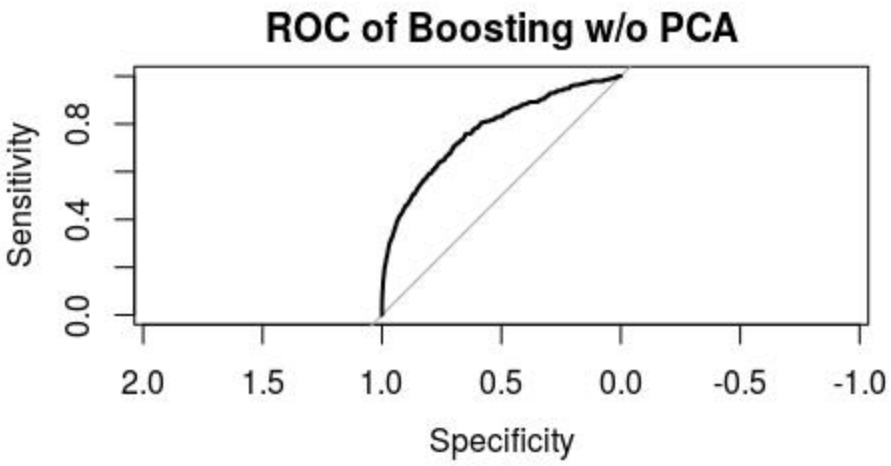
```
plot(test.H1B.smote.nb.roc, main = "ROC of Naive Bayes w/o PCA")
plot(test.smote.pls.roc, main = "ROC of PLS w/o PCA")
plot(test.H1B.smote.glmnet.roc, main = "ROC of Glmnet w/o PCA")
plot(test.H1B.smote.rf.roc, main = "ROC of Random Forest w/o PCA")
plot(test.H1B.smote.bag.roc, main = "ROC of Bagging w/o PCA")
plot(test.H1B.smote.ada.roc, main = "ROC of Boosting w/o PCA")
plot(test.H1B.smote.gam.roc, main = "ROC of GAM w/o PCA")

plot(test.H1B.smote.pca.glm.roc, main = "ROC of GLM with PCA")
plot(test.H1B.smote.pca.lda.roc, main = "ROC of LDA with PCA")
plot(test.H1B.smote.pca.rpart.roc, main = "ROC of Decision Tree with PCA")
plot(test.H1B.smote.pca.nb.roc, main = "ROC of Naive Bayes with PCA")
plot(test.H1B.smote.pca.pls.roc, main = "ROC of PLS with PCA")
plot(test.H1B.smote.pca.glmnet.roc, main = "ROC of Glmnet with PCA")
plot(test.H1B.smote.pca.rf.roc, main = "ROC of Random Forest with PCA")
plot(test.H1B.smote.pca.bag.roc, main = "ROC of Bagging with PCA")
plot(test.H1B.pca.smote.ada.roc, main = "ROC of Boosting with PCA")
plot(test.H1B.smote.pca.gam.roc, main = "ROC of GAM with PCA")
```

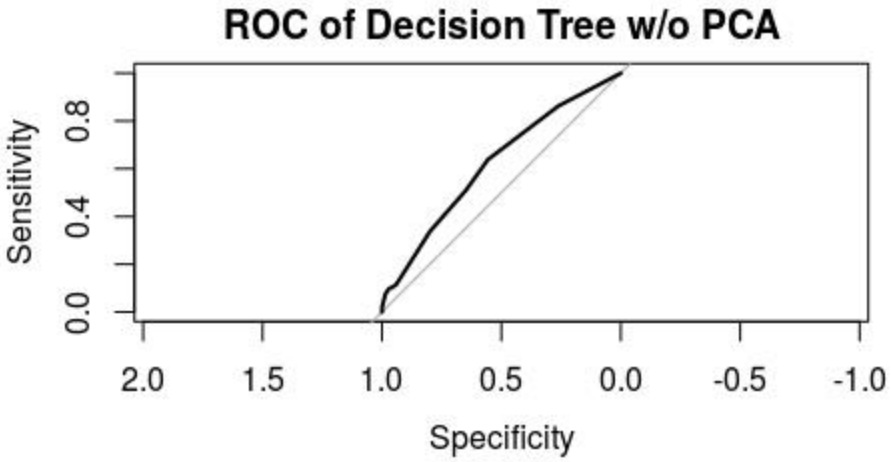
Appendix H - Individual ROC

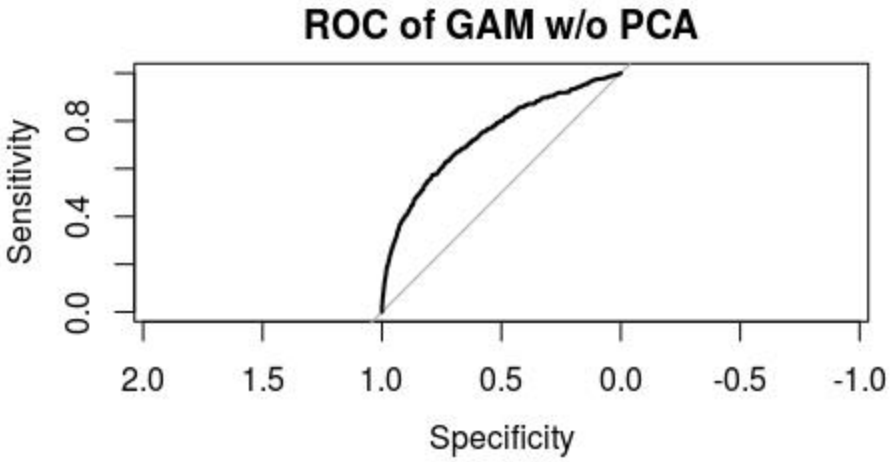


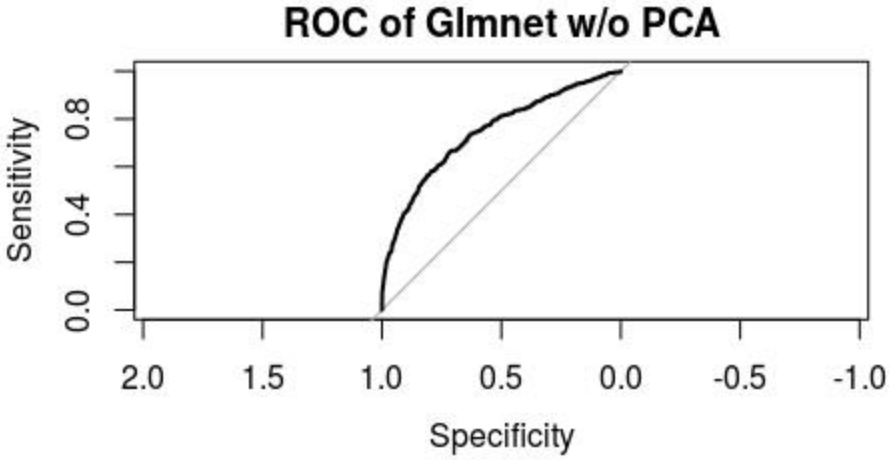


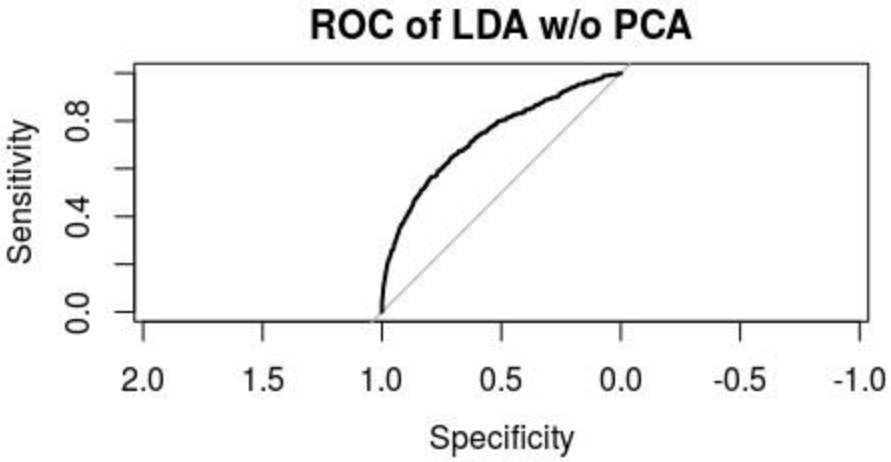


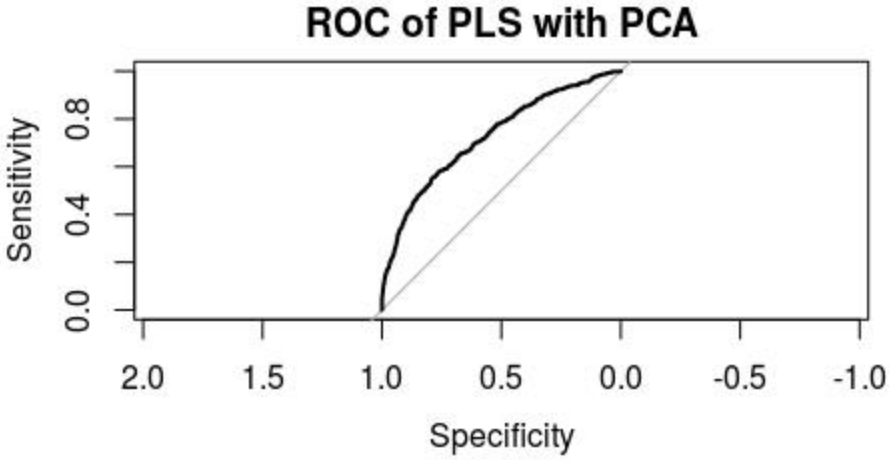

ROC of Boosting w/o PCA
Sensitivity
0.0
0.4
0.8
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
Specificity

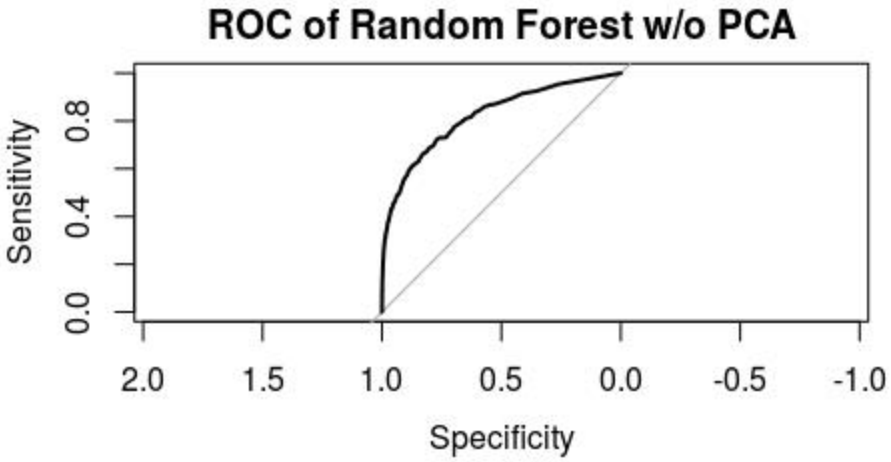

ROC of Decision Tree w/o PCA
Sensitivity
0.0
0.4
0.8
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
Specificity

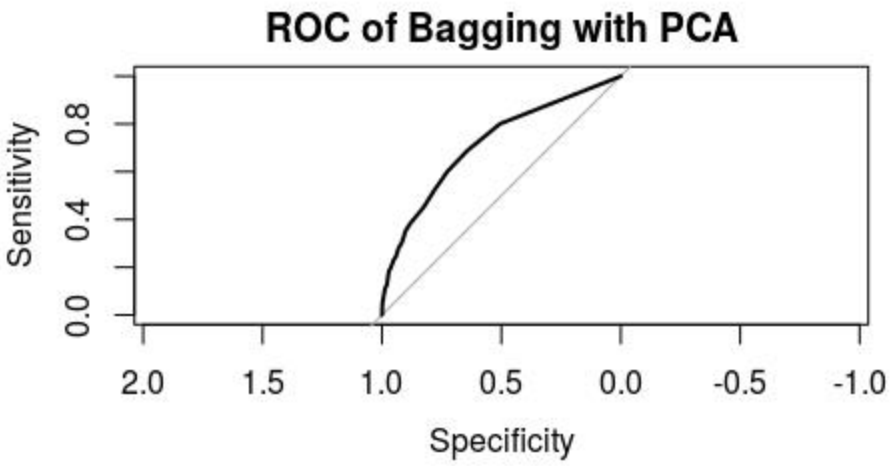

ROC of GAM w/o PCA
Sensitivity
0.0
0.4
0.8
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
Specificity

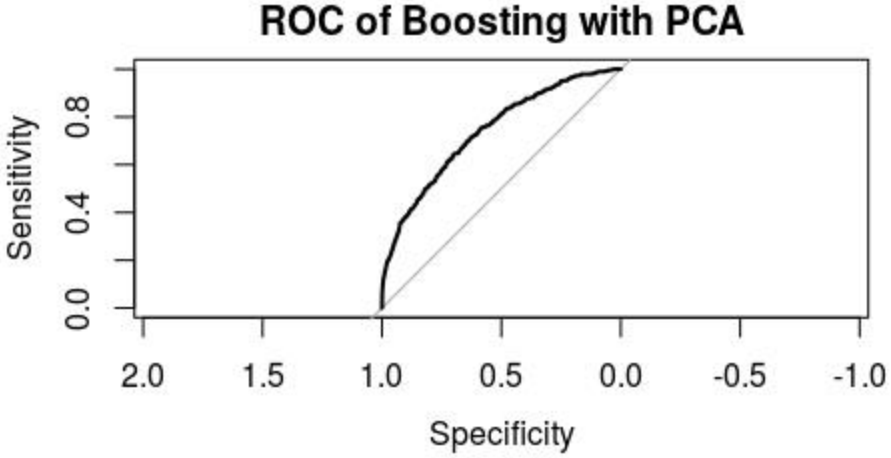

ROC of Glmnet w/o PCA
Sensitivity
0.0
0.4
0.8
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
Specificity

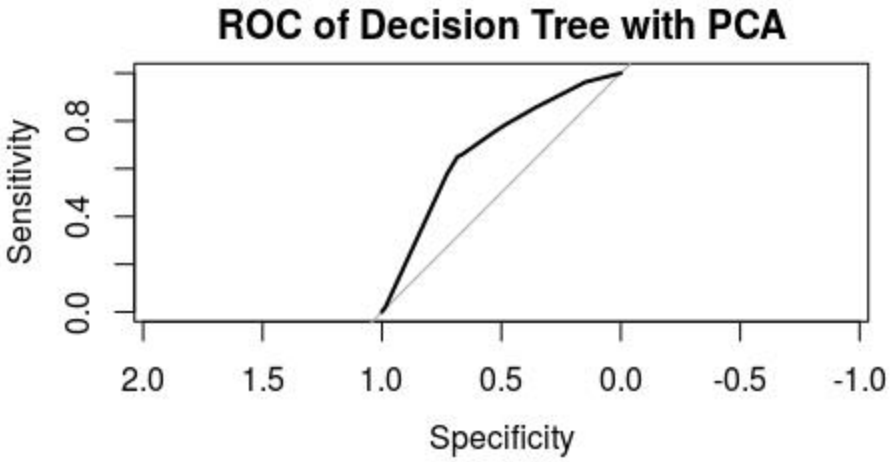

ROC of LDA w/o PCA
Sensitivity
0.0
0.4
0.8
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
Specificity

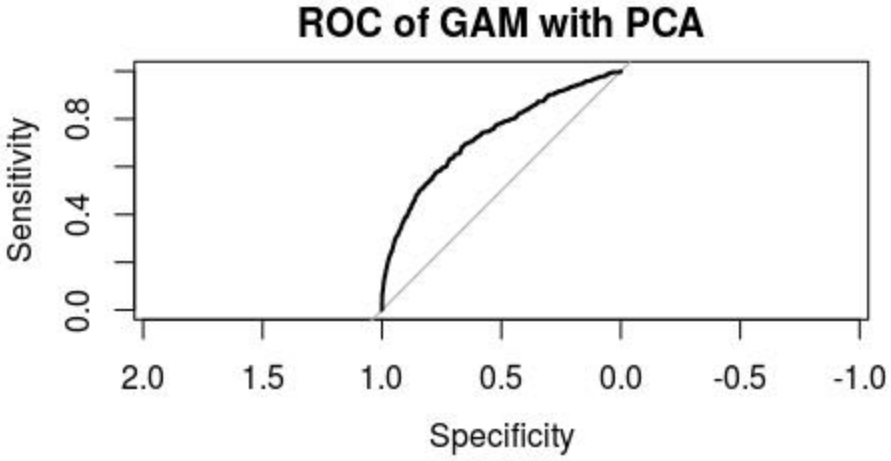

ROC of PLS with PCA
Sensitivity
0.0
0.4
0.8
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
Specificity

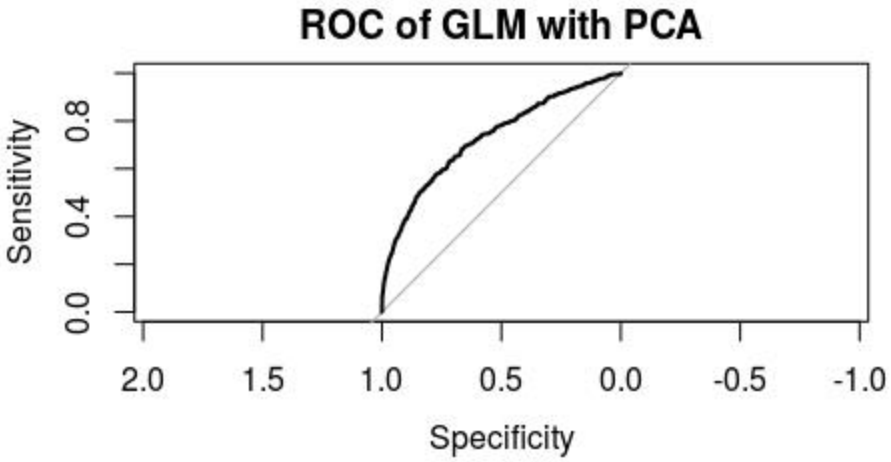

ROC of Random Forest w/o PCA
Sensitivity
0.0
0.4
0.8
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
Specificity

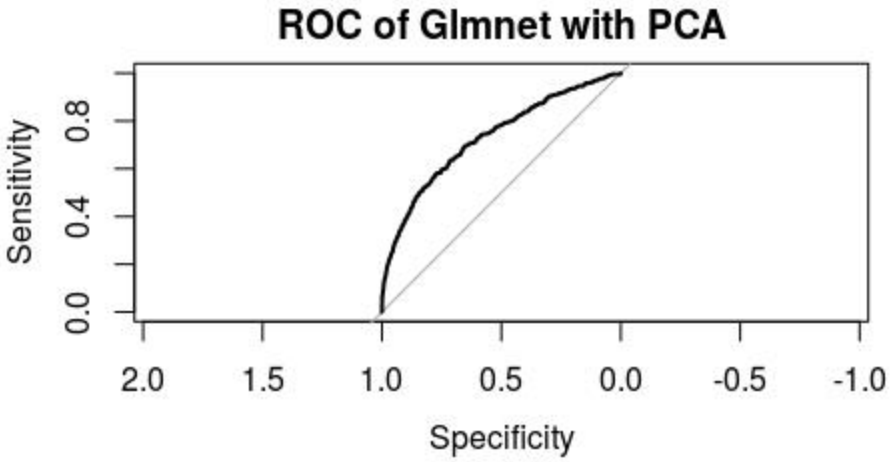

ROC of Bagging with PCA
Sensitivity
0.0
0.4
0.8
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
Specificity

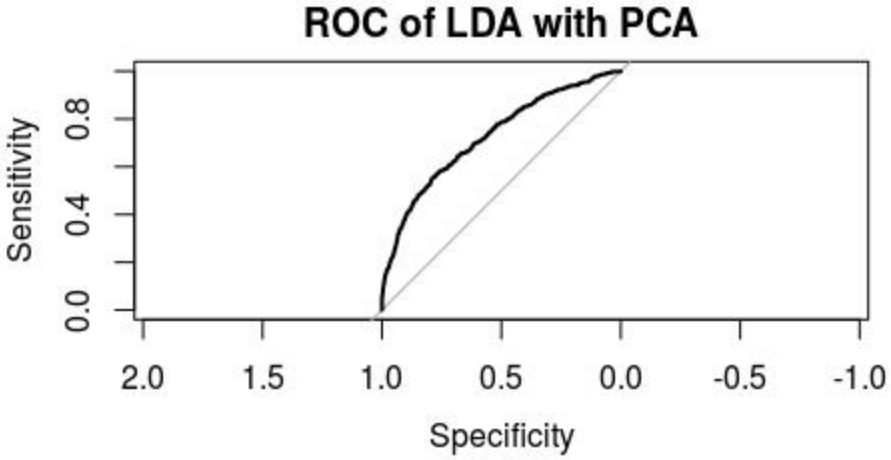

ROC of Boosting with PCA
Sensitivity
0.0
0.4
0.8
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
Specificity

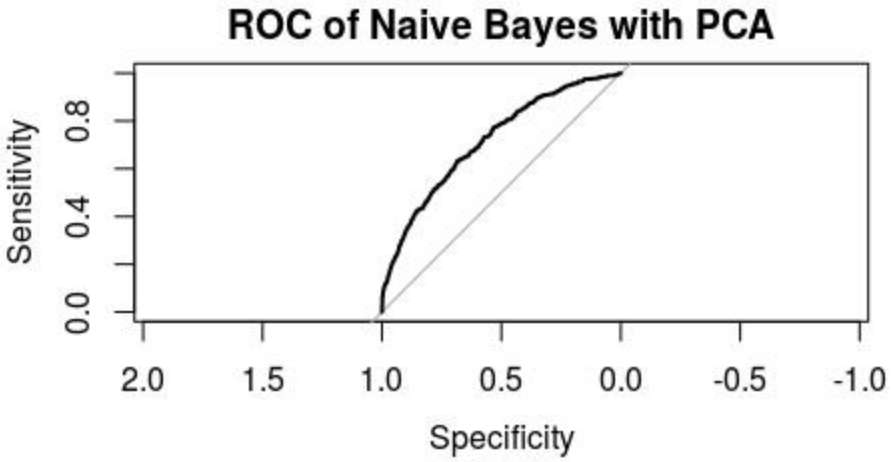

ROC of Decision Tree with PCA
Sensitivity
0.0
0.4
0.8
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
Specificity

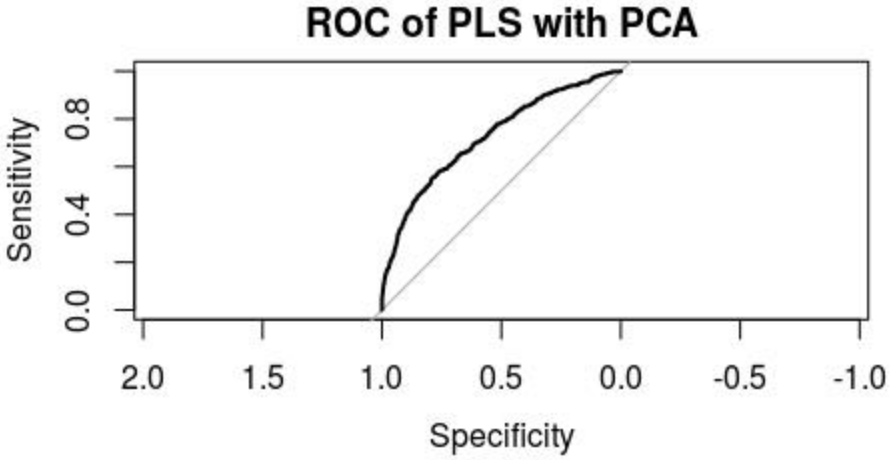

ROC of GAM with PCA
Sensitivity
0.0
0.4
0.8
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
Specificity

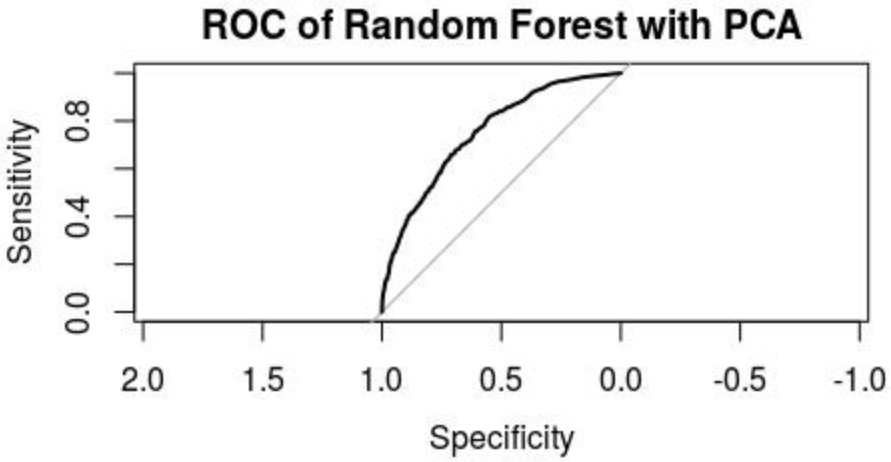

ROC of GLM with PCA
Sensitivity
0.0
0.4
0.8
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
Specificity


ROC of Glmnet with PCA
Sensitivity
0.0
0.4
0.8
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
Specificity


ROC of LDA with PCA
Sensitivity
0.0
0.4
0.8
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
Specificity


ROC of Naive Bayes with PCA
Sensitivity
0.0
0.4
0.8
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
Specificity


ROC of PLS with PCA
Sensitivity
0.0
0.4
0.8
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
Specificity


ROC of Random Forest with PCA
Sensitivity
0.0
0.4
0.8
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
Specificity