Final\_HW4

Coffy Andrews-Guo

2023-05-18

# Exploratory Analysis using Ensemble Machine Learning Algorithms

[Github Source Code](https://github.com/candrewxs/D622)

### Introduction

The [Tuition Assistance Program (TAP) Recipients & Dollars by Income, Age Group and Program Information: Beginning 2000](https://data.ny.gov/Education/Tuition-Assistance-Program-TAP-Recipients-Dollars-/2t78-bs45) [dataset](https://data.ny.gov/resource/2t78-bs45.csv) contains information on the financial aid provided by New York State to eligible students for attending college and it is New York’s largest student financial aid grant program. Since 2020, the Tuition Assistance Program (TAP) has been providing financial aid grants to eligible residents of New York who are pursuing their post-secondary education within the state. TAP grants are awarded based on the New York State taxable income of the applicant and their family. This dataset contains information on the TAP grant recipients from July 1 through June 30, including the dollar amount of the grant, grouped by various factors such as income and age group. It is New York’s largest student financial aid grant program.

***Prediction using Binary Classification***  
*What is the likelihood of a student to receive an above average tuition assistance based on the dataset variables (features), such as age, financial status, or program of study?*

### Exploring and Preparing the Data

The TAP data set is a comma-separated values (CSV) file containing 130.3 Kb of information. The data set information includes 1000 observations and 16 features. These features are categorized based on their categorical (discrete) form and integer/real number (continuous) form.

The dataset includes several features for the analysis, listed here:  
- *Academic Year*: Academic Year is from July 1 through June 30  
- *Level*: U = Undergraduate G = Graduate

- TAP Level of Study: Student’s Level of Study  
- 2 yr Undergrad = Undergraduate 2 Year Program of Study

- 4 yr Undergrad = Undergraduate 4 Year Program of Study

- 5 yr Undergrad = Approved Undergraduate 5 Year Program of Study

- STAP = Supplemental Tuition Assistance Program (authorized additional aid for remedial courses)

- Grad = Graduate Level Program of Study

- *Sector Type*: Type of Institution is either Public or Private  
- *TAP Sector Group*: Sector Group of Institution:

1-CUNY SR = CUNY Senior Colleges

2-CUNY CC = CUNY Community Colleges

3-SUNY SO = SUNY State Operated

4-SUNY CC = SUNY Community Colleges

5-INDEPENDENT = Independent Colleges

6-BUS. DEGREE = Business Degree Granting Institutions

7-BUS. NON-DEG = Non-Degree Business Schools

8-OTHER = All Other Institutions  
9-CHAPTER XXII = Chapter XXII TAP Schools

- *Recipient Age Group*: Age of student as of July 1 start of academic year

- *Tap Financial Status*: Financial Status is either Financial\_Dependent or Financial\_Independent Character

- *Tap Award Schedule*: There are 3 awards schedules:  
Dependent Schedule,  Independent Schedule,  or Married No Dependents Schedule  
- *Tap Degree or NonDegree*:

Degree = Program of study is classified as degree granting

Non Degree = Program of study is classified as non degree  
- *Tap Schedule Letter*: Refer to Tuition Assistance Program award schedule documentation. The link is provided in the Additional Resources section after selecting the About tab.  
- *Income by $1,000 Range*: When performing data analysis, one of three income ranges can be selected. The $1,000 income range is the lowest level of granularity that is available in this dataset. Recipient New York State Net Taxable Income by Category  
- *Income by $5,000 Range*: When performing data analysis, one of three income ranges can be selected. The $5,000 income range is the middle level of granularity that is available in this dataset. Recipient New York State Net Taxable Income by Category  
- *Income by $10,000 Range*: When performing data analysis, one of three income ranges can be selected. The $10,000 income range is the highest level of granularity that is available in this dataset. Recipient New York State Net Taxable Income by Category  
- *Tap Recipient Headcount*: Number of recipients as measured by students receiving at least one term award during the academic year.  
- *Tap Recipient FTEs*: Number of recipients as measured by academic year Full-Time Equivalents: Full Time Equivalent is a unit that indicates the enrollment of a student in credit-bearing courses in a way that makes it comparable across contexts. An FTE of 1.0 means that the person is equivalent to 1 full-time student, while an FTE of 0.5 signals that a student is enrolled half-time.  
- *Tap Recipient Dollars:* Total TAP award dollars provided on behalf of TAP recipients attending an Institution.

This analysis will investigate the probability of a student obtaining tuition assistance that exceeds the average amount. The dataset that will be used for the analysis is imbalanced, containing only a small number of positive cases (high award) compared to negative ones (average/low award). Specifically, there are a total of **330** (33%) high award cases and **670** (67%) average/low award cases. I will utilize an ensemble of machine-learning methods to address this issue.

### Preparation

The dataset was prepared and manipulated to resolve issues such as:  
- Changing:

* character variables to factor variables
* integer variables to numeric variables
* numeric variables that are known to be factor variables.
* variables name to a shorter description

- Outliers – the entire data was kept, and log transformation was applied to modify the data structure to meet the requirements of the machine learning approach.  
- Feature extraction – removed features that have no impact on the machine learning approach.

**DESCRIPTIVE STATISTICS**

The summary statistics indicates two different formats: one format for categorical features and the other for continuous features. The summary statistics for the categorical features, such as *sector\_type*, show the feature values along with the frequency for each value. The second format applies to continuous features, showing the mean, median, minimum, maximum, and the first and third quartile values.

The *summary()* function list the number of missing values (NAs) for the features and NAs are non-existent in the data set.

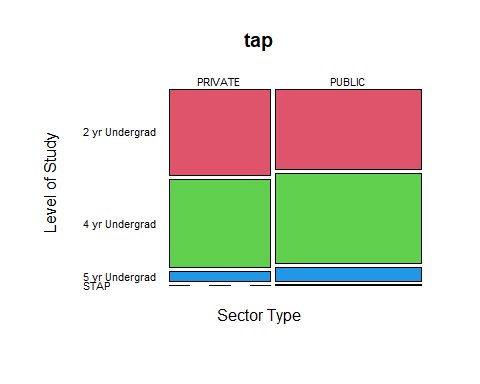
### Visualizing the Data

**Relationship between two categorical variables**

The contingency table using the table() function shows the number of cases in each *sector\_type* subgroup compared to *level\_of\_study*:

|  |  |  |
| --- | --- | --- |
|  | PRIVATE | PUBLIC |
| 2 yr Undergrad | 192 | 258 |
| 4 yr Undergrad | 196 | 289 |
| 5 yr Undergrad | 21 | 43 |
| STAP | 0 | 1 |

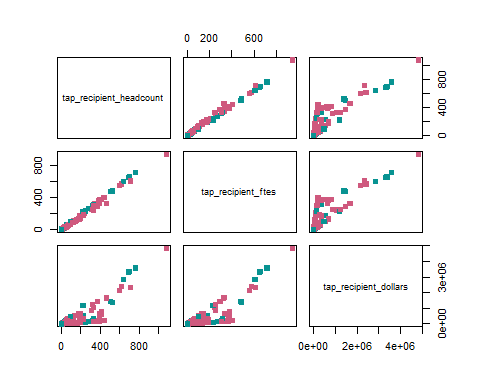
*Visualizing the contingency table:*



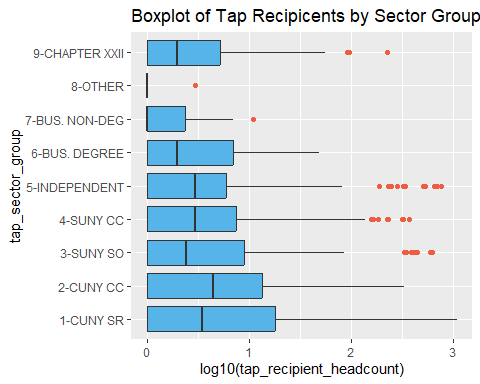
The contingency table shows the *Public* sector type have more TAP recipients than the *Private* sector type among the various *Level of Study*.

*Relationship between numeric variables*

The correlation between the numeric variables show several pairs of variable are positively linked in response to a each other. Also, the relationship between the pairs of variables shows *outlier* in the data and probable bias.



In the boxplot, we see the distribution of values between *tap recipients* and *sector group*. The outliers are represented by red circles that extend beyond either whisker.



### BUILDING THE CLASSIFICATION ENSEMBLE MODELS

An ensemble machine learning algorithm is a technique that combines multiple individual models to make predictions or decisions. Instead of relying on a single model, an ensemble leverages the collective knowledge of multiple models to improve overall performance and accuracy. The idea behind ensemble models is that by combining the predictions of diverse models, the weaknesses of individual models can be mitigated, and the strengths can be amplified.

This analysis will apply the ensemble machine learning algorithms: *(1) Random Forest, (2) Bagged Trees, (3)eXtreme Gradient Boosting “XGBoost” (4) C5.0, (5) Stacking (includes: CART “rpart”, k-Nearest Neighbors “knn”, svmRadial, Generalized Linear Model “glm”, and Naïve Bayes “naïve\_bayes”).*

#### Random Forest (p. 355)

Random Forest is an ensemble learning approach that combines multiple decision trees. It works by creating bootstrap samples from the training data and constructing decision trees on these samples. Each tree considers a random subset of features at each node. During prediction, the trees’ outputs are aggregated through voting or averaging. Random Forest reduces overfitting, improves generalization, and provides feature importance measures. It is known for its accuracy, robustness, and versatility in handling classification and regression tasks.

#### Bagging

Bagging, or bootstrap aggregating, is an ensemble learning technique that combines multiple models to make predictions. It works by creating subsets of the training data through bootstrapping and training individual models on these subsets. During prediction, the models’ outputs are combined to obtain the final prediction. Bagging reduces variance, improves performance, and helps with overfitting by capturing different aspects of the data. It is a powerful technique that increases accuracy and stability in machine learning algorithms.

#### Boosting (XGBoost)

Extreme Gradient Boosting (XGBoost) is an ensemble learning approach that combines multiple decision trees using gradient boosting. It starts with an initial model and calculates gradients and residuals to build new trees capturing the errors. The model is updated iteratively by combining the predictions from the new trees with a learning rate. XGBoost applies regularization techniques to prevent overfitting and produces a final prediction by combining the predictions from all the trees. It is known for its speed, performance, and ability to handle complex relationships in the data.

#### C5.0

C5.0 is an ensemble learning approach that combines decision trees to create a powerful predictive model. It starts by splitting the data based on attribute values to create pure subsets. Decision trees are constructed using information gain, and pruning is applied to reduce overfitting. Multiple decision trees are generated using different subsets of data or attributes, and their predictions are combined through voting. C5.0 improves accuracy, handles complex relationships, and is widely used for its interpretability and feature selection capabilities.

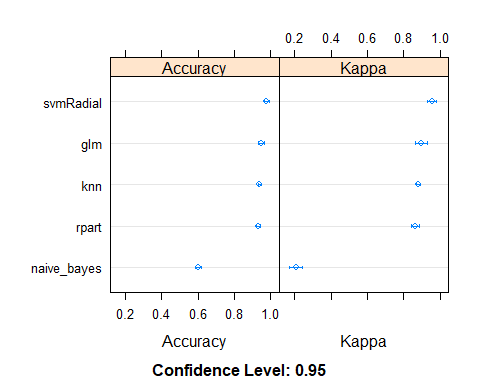
#### Stacking

Stacking is an ensemble learning approach that combines the predictions of multiple models using a meta-model. It involves training diverse base models and using their predictions as input features for training a meta-model. The meta-model learns to combine and weigh the predictions of the base models to make the final prediction. Stacking leverages the strengths of different models and captures complex patterns in the data. It can outperform individual models and requires careful model selection, dataset partitioning, and meta-model tuning.

*Using only caret models: CART(rpart), k-Nearest Neighbors(knn), svmRadial, Generalized Linear Model(glm), Naive Bayes(naive\_bayes).*

After training our base models, the subsequent step involves training the meta-model, which acts as the combination function. However, before proceeding, it is essential to analyze the performance of our base models against the training data. To achieve this, we utilize the “resamples()” function to gather results from each model and then employ the “summary()” function to obtain summary statistics of these results.

Call:  
summary.resamples(object = ans)  
  
Models: rpart, knn, svmRadial, glm, naive\_bayes   
Number of resamples: 10   
  
Accuracy   
 Min. 1st Qu. Median Mean 3rd Qu. Max. NA's  
rpart 0.8989899 0.9257906 0.9294706 0.9324451 0.9396224 0.9648241 0  
knn 0.9242424 0.9346734 0.9419192 0.9405182 0.9444444 0.9597990 0  
svmRadial 0.9343434 0.9810606 0.9848866 0.9788158 0.9849246 0.9949495 0  
glm 0.9040404 0.9457071 0.9546597 0.9495685 0.9623116 0.9798995 0  
naive\_bayes 0.5707071 0.5845960 0.5979899 0.6028095 0.6243719 0.6363636 0  
  
Kappa   
 Min. 1st Qu. Median Mean 3rd Qu. Max. NA's  
rpart 0.7979798 0.8516068 0.8589465 0.8648959 0.8792424 0.9296642 0  
knn 0.8484848 0.8693500 0.8838384 0.8810392 0.8888889 0.9196041 0  
svmRadial 0.8686869 0.9621212 0.9697720 0.9576315 0.9698500 0.9898990 0  
glm 0.8080808 0.8914141 0.9093068 0.8991378 0.9246349 0.9598020 0  
naive\_bayes 0.1414141 0.1691919 0.1960313 0.2056677 0.2477757 0.2727273 0

Visualize the results: 

The analysis of the results indicates that the four models exhibit similar average performance, with the svmRadial model,svm(Radial), achieving the highest performance among the five.

We are prepared to construct the ultimate component of our stacking ensemble, known as the meta-model. To accomplish this, we employ the random forest ensemble method as our chosen machine learning algorithm. The *caretEnsemble* package offers us a convenient function called *caretStack()*, enabling us to combine multiple predictive models through the stacking technique.

### Results and Discussion

This section presents the results of experiments conducted to study the performance of various decision tree ensembles.As the data is imbalanced, classification accuracy is not an appropriate performance measure to compare different classifiers. F-measure, precision, recall, accuracy, and kappa are employed to compare the performances of different classification ensemble models.

Precision Recall F1 Accuracy Kappa  
Random Forest 0.8823529 0.9146341 0.8982036 0.9317269 0.8468690  
Bagging 0.8823529 0.9146341 0.8982036 0.9317269 0.8468690  
XGBoost 0.8941176 0.9268293 0.9101796 0.9397590 0.8648844  
**C5.0 0.9743590 0.9268293 0.9500000 0.9678715 0.9263531**  
GLM-Stack 0.9285714 0.9512195 0.9397590 0.9598394 0.9096451

**Model Evaluation:** C5.0 performed the best among the other ensemble models. The C5.0 performance using cross-validation helped to assess the accuracy and generalization ability of the model. Thefore, the ensemble approach in *C5.0* improved the model’s accuracy, robustness, and ability to handle complex relationships in the data. It reduced the risk of overfitting and provided a more reliable prediction by combining the outputs of multiple ensemble learning approach as shown in the above table.

C5.0 is widely used for its effectiveness, interpretability, and feature selection capabilities. It has been applied to various domains and is considered one of the prominent algorithms in the field of machine learning.

**In Conclusion:**

A student inquiry on the likelihood of receiving an above average tuition assistance based variables (features), such as age, financial status, or program of study will have a **97% precision** with the C5.0 decision tree ensemble algorithm. Although some institutions may utilize machine learning algorithms such as C5.0 or other decision tree-based approaches, their calculations typically involve complex considerations beyond just algorithmic models.

## REFERENCES

* [An Introduction to Statistical Learning with Applications in R](https://hastie.su.domains/ISLR2/ISLRv2_website.pdf)
* Nwanganga, F., & Chapple, M. (2020). Practical machine learning in R. John Wiley & Sons.
* [Ensemble Learning in R](https://github.com/kmutya/Ensemble-Learning-in-R)
* [Chapter 27 Ensemble Methods](https://daviddalpiaz.github.io/r4sl/ensemble-methods.html)
* [Foley, M. (2020, July 26). 8.6 Summary | My Data Science Notes.](https://bookdown.org/mpfoley1973/data-sci/summary.html)

## Appendix: All code for this report

knitr::opts\_chunk$set(echo = FALSE, comment = NA, warning = FALSE, message = FALSE)  
suppressPackageStartupMessages({  
 #data import, tidying, manipulation, visualization, and programming  
 library("tidyverse")   
 library("flextable") #pretty table  
 library("mlbench") #frequency table  
 library("caTools") #data partitioning  
 library("DMwR") #balance dataset  
 #R for Statistical Learning  
 library("rpart")  
 library("rpart.plot")  
 library("randomForest")  
 library("gbm")  
 library("caret") #data partitioning  
 library("MASS")  
 library("ISLR")  
 library("xgboost")  
 library("caretEnsemble")  
 library("cowplot") #arrange multiple plots  
 library("ROSE")  
})  
# load the dataset and renamed it tap  
tap <- read.csv("https://data.ny.gov/resource/2t78-bs45.csv")  
format(object.size(tap), units = "auto")  
glimpse(tap)  
# convert all character columns to factor  
# convert academic\_year variable to factor (qualitative measure)  
tap <- tap %>%  
 mutate\_if(is.character, as.factor) %>%  
 mutate\_if(is.integer, as.numeric) %>%  
 mutate(academic\_year = as.factor(academic\_year))  
# view statistical summary of the updated data frame  
summary(tap)  
#find total NA values in data frame  
sum(is.na(tap))  
# Descriptive Analysis - Contingency Table  
knitr::kable(table(tap$tap\_level\_of\_study, tap$sector\_type))  
# Mosaic Plot  
# allow to visualize a contingency table of two qualitative variables  
mosaicplot(~ sector\_type + tap\_level\_of\_study, data = tap, color = 2:4, las = 1,   
 ylab = "Level of Study",  
 xlab = "Sector Type")   
# remove categorical variable  
  
tap.num <- tap %>% dplyr::select(tap\_recipient\_headcount, tap\_recipient\_ftes, tap\_recipient\_dollars)  
  
# display 5 first obs. of new dataset  
head(tap.num, 5)  
  
#multiple scatterplots  
sector <- tap[, 4]  
l <- length(unique(sector))  
#pairs(tap.num, pch = 19, col = 10, main = "Pairwise Correlation Plot")  
pairs(tap.num, pch = 22, bg = hcl.colors(l, "Temps")[sector],  
 col = hcl.colors(l, "Temps")[sector])  
#boxplot  
tap %>%  
 ggplot() +   
 geom\_boxplot(mapping = aes(x = tap\_sector\_group, y = log10(tap\_recipient\_headcount)), fill = "#56B4E9",  
 outlier.colour = "tomato2") +   
 labs(title = "Boxplot of Tap Recipicents by Sector Group") +  
 coord\_flip()  
#Distribution - show the statistical distribution of the values of a feature. It shows the spread and skewness of data for a particular feature.  
tap %>%  
 ggplot() +   
 geom\_histogram(mapping = aes(x = log10(tap\_recipient\_headcount)), bins = 30, fill = "yellow", color = "black") +   
 labs(title = "Histogram of Tap Recipicents")  
#Composition - shows the component makeup of the data (stacked and pie charts)  
tap %>%  
 ggplot() +   
 geom\_bar(mapping = aes(x = income\_by\_10\_000\_range, fill = tap\_level\_of\_study), color = "black") +   
 labs(title = "Stacked bar chart on level of study by Income Range", x = "Num. of Tap Recipients", y = "Tap Fees") +  
 coord\_flip()  
#create new object  
#remove variables that has no value  
tap2 <- tap %>% dplyr::select(-c(academic\_year, level, tap\_degree\_or\_nondegree))  
  
names(tap2)[8] <- "income\_1k"  
names(tap2)[9] <- "income\_5k"  
names(tap2)[10] <- "income\_10k"  
  
#For skewed distributions and data with values that range over several orders of magnitude, the log transformation is usually more suitable  
#Applying log transformation on the continuous features/numeric values  
  
tap2 <- tap2 %>%  
 #select(tap\_recipient\_headcount, tap\_recipient\_ftes, tap\_recipient\_dollars) %>%  
 mutate(tap\_recipient\_headcount = log10(tap\_recipient\_headcount)) %>%  
 mutate(tap\_recipient\_ftes = log10(tap\_recipient\_ftes)) %>%  
 mutate(tap\_recipient\_dollars = log10(tap\_recipient\_dollars))  
  
tap2 %>%  
 keep(is.numeric) %>%  
 summary()  
p2 <- ggplot(tap2, aes(tap\_recipient\_dollars)) +   
 geom\_histogram(bins = 15, fill = "yellow", color = "black") +   
 geom\_vline(xintercept = mean(tap2$tap\_recipient\_dollars), col = "red", lwd = 3) +  
 labs(title = "Histogram of Tap Recipicent Dollars")   
  
p2 +   
 annotate("text", # Add text for mean  
 x = mean(tap2$tap\_recipient\_dollars) \* 1.3,  
 y = mean(tap2$tap\_recipient\_dollars) \* 40,  
 label = paste("Mean =", round(mean(tap2$tap\_recipient\_dollars), 2)),  
 col = "red",  
 size = 6)  
#Modify variable, tap\_recipient\_dollars, from numeric to categorical  
tap2$tap\_recipient\_dollars <- as.factor(ifelse(tap2$tap\_recipient\_dollars <= "4", "Low", "High"))  
summary(tap2)  
#split dataset into training and test sets  
set.seed(1234)  
sample\_set2 <- createDataPartition(y = tap2$tap\_recipient\_dollars, p = 0.75, list = FALSE)  
dollars\_train <- tap2[sample\_set2,]  
dollars\_test <- tap2[-sample\_set2, ]  
round(prop.table(table(dplyr::select(dollars\_train, tap\_recipient\_dollars), exclude = NULL)), 4) \* 100  
#balance the training data  
set.seed(1234)  
dollars\_train <- SMOTE(tap\_recipient\_dollars ~ ., data.frame(dollars\_train),  
 perc.over = 100, perc.under = 200)  
round(prop.table(table(dplyr::select(dollars\_train, tap\_recipient\_dollars), exclude = NULL)), 4) \* 100  
#Implementation of random forest using caret  
set.seed(9)  
control = trainControl(method="repeatedcv", number=5, repeats=2, savePredictions=TRUE, classProbs=TRUE)  
rf = train(tap\_recipient\_dollars~., data = dollars\_train, method = 'rf', metric = 'Accuracy', trControl = control)  
rf  
set.seed(1239)  
rf\_mod <- train(tap\_recipient\_dollars ~., data = dollars\_train, metric = "Accuracy",   
 method = "rf", trControl = trainControl(method = "repeatedcv"),  
 tuneGrid = expand.grid(.mtry = 66))  
rf\_mod  
rf\_pred <- predict(rf\_mod, dollars\_test)  
(rf\_cm <- confusionMatrix(rf\_pred, dollars\_test$tap\_recipient\_dollars, positive = "High"))  
set.seed(9)  
bg <- train(tap\_recipient\_dollars ~., data = dollars\_train, method = "treebag", metric = "Accuracy", trControl = control)  
bg  
#evaluate how well Bagged performs  
bg\_pred <- predict(bg, dollars\_test)  
(bg\_cm <- confusionMatrix(bg\_pred, dollars\_test$tap\_recipient\_dollars, positive = "High"))  
#XGBoost algorithm("xgboost")  
  
set.seed(9)  
xgb\_mod<- train(tap\_recipient\_dollars ~., data = dollars\_train, method = "xgbTree",  
 metric = "Accuracy", trControl = control, tuneGrid = expand.grid(  
 nrounds = 100, max\_depth = 6, eta = 0.3, gamma = 0.01,   
 colsample\_bytree = 1, min\_child\_weight = 1, subsample = 1  
 ))  
xgb\_mod  
#evaluate how well xgboost performs  
xgb\_pred <- predict(xgb\_mod, dollars\_test)  
(xgb\_cm <- confusionMatrix(xgb\_pred, dollars\_test$tap\_recipient\_dollars, positive = "High"))  
varImp(xgb\_mod)  
#Implementation of AdaBoost and Boosted classification trees using caret  
set.seed(9)  
C5 = train(tap\_recipient\_dollars~., data = dollars\_train, method = 'C5.0', trControl = control)  
C5  
  
C5\_pred <- predict(C5, newdata = dollars\_test)  
(C5\_cm <- confusionMatrix(C5\_pred, dollars\_test$tap\_recipient\_dollars, positive = "High"))  
#caretEnsemble() library - ensemble methods  
#Let's build models rpart, knn, svm(radial), glm & naive\_bayes  
set.seed(9)  
algorithms = c('rpart', 'knn', 'svmRadial', 'glm', 'naive\_bayes')  
set.seed(7)  
models = caretList(tap\_recipient\_dollars~., data = dollars\_train, trControl=control, methodList=algorithms)  
summary(models)  
  
ans = resamples(models) #resamples helps to tabularize the results  
summary(ans)  
  
dotplot(ans)  
#Check correlations between models to ensure the results are uncorrelated and can be ensemble  
modelCor(ans)  
  
# library ("ROSE")  
# stack using Logistics Regression  
set.seed(9)  
stack.glm = caretStack(models, method="glm", metric="Accuracy", trControl=control) #logistic  
print(stack.glm)  
#evaluate how well xgboost performs  
stackglm\_pred <- predict(stack.glm, dollars\_test)  
(stglm\_cm <- confusionMatrix(stackglm\_pred, dollars\_test$tap\_recipient\_dollars, positive = "High"))  
names(bg\_cm$byClass)  
names(bg\_cm$overall)  
rf\_acc <- c(rf\_cm$byClass[5:7], rf\_cm$overall[1:2]) #random forest  
bg\_acc <- c(bg\_cm$byClass[5:7], bg\_cm$overall[1:2]) #bagging  
xgb\_acc <- c(xgb\_cm$byClass[5:7], xgb\_cm$overall[1:2]) #XGBoost  
C5\_acc <- c(C5\_cm$byClass[5:7], C5\_cm$overall[1:2]) #C5.0  
stglm\_acc <- c(stglm\_cm$byClass[5:7], stglm\_cm$overall[1:2]) #glm stacked  
  
tab\_model <- rbind(rf\_acc, bg\_acc, xgb\_acc, C5\_acc, stglm\_acc)  
  
tab.model <- data.frame(tab\_model)  
rownames(tab.model) <- c("Random Forest", "Bagging", "XGBoost", "C5.0", "GLM-Stack")  
tab.model  
#flextable(tab.model) %>% align(align = "center", part = "all") %>% autofit()