Income Forecasting

STAT 4502 Project

Mansoo Cho, Delin Shao, Jin Yao, Minghao Yin, Chi Zhang, Darry Zhang

May 12, 2020

Contents

Contents

1 Introduction

As of May 9, the number of US COVID-19 diagnoses was 1,346,336, which is the largest in the world. The outbreak of COVID-19 caused a huge blow to the US economy. In March, the United States imposed travel restrictions in almost all 50 states to reduce the spread of the virus. Approximately 330 million Americans are paralyzed in public life. Many shops, shopping centers and businesses are closed, and restaurants and hotels are empty. Many employees of these companies are forced to apply for unemployment benefits because they have lost their jobs. According to The Labor Department report, most travel businesses were closed, and leisure and hospitality work was reduced by 7.6 million. The retail and healthcare sectors each declined by 2.1 million. Manufacturing lost 1.3 million and government jobs decreased by 980,000. Now 20.5 million jobs are lost last month as the nation locked down against the coronavirus. The jobless rate soared to 14.7% which is the highest level since the Great Depression. (Horsley, 2020) In this difficult period, everyone needs a certain amount of savings to overcome the difficulties. So we are interested in people's income. We found data from the Machine Learning Repository. This data describes the information whether the Americans have an annual income of more than 50,000 US dollars or not, based on their education level, marital status, race, gender, etc. In this project, we want to predict if these factors would affect people's annual income to be higher than 50,000 dollars or lower. We will emphasize the study of the process of building the random forest.

2 Method

In this project, we intend to use random forest. Random forest is a classification method that consists of a "forest" of decision trees. The method basically creates the forest by bootstrapping the dataset and choosing only a subset of predictors for each split in the trees. To understand the concept of random forest, we first need to discuss "bagging."

Decision trees tend to have high variance, and this can be problematic. Suppose we split a dataset and fit a decision tree on each training data. If the trees have high variance, then the outcomes we get from each tree can be quite different. This makes it harder to classify the data. Bagging (also known as Bootstrap aggregating) seeks to address such an issue by reducing the variance. It reduces the variance by averaging a set of observations. Ideally, obtaining many training data from the population will be suitable for the reduction, but this is not practical. Instead, we can repeatedly sample from one training data (i.e. bootstrap) and take the average. Suppose we bootstrap to create B training sets and build our prediction model (in our case a decision tree) $\hat{f}(x)$ on the b-th training set. We then take the average of the predictions to get a "bagged tree"

$$\hat{f}_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{b}(x)$$

(James, 2017).

Random forest is quite similar to bagging in the sense that it creates bootstrap samples and builds a collection of trees. However, random forest makes the trees uncorrelated by considering only a subset of predictors when building a tree. That is, random forest only considers m best predictors out of all p predictors at each node. The subset of preditors are chosen at random, and typically $m \approx \sqrt{p}$. (James, 2017) After building a forest of decision trees (i.e. a random forest), we may use the forest to predict an outcome. Suppose we want to classify a new observation x from a test data. Then, we use each tree in the forest to predict the outcome for x. We record for each tree what category x is categorized into. Lastly, we take a majority vote for the category for x. According to Hastie et al, (Hastie et al, 2009) the whole algorithm is summarized in this way:

1. For $b = 1$ to B:

(a) Draw a bootstrap sample Z of size N from the training data.

(b) Grow a random forest tree T_b of to the bootstrapped data, by repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached.

  i. Select m predictors at random from the p predictors.
  ii. Pick the best predictor or split-point among the m predictors.
  iii. Split the node into two daughter nodes.

2. Output the collection of trees $T_{b_1}^{B}$.

3. To make a prediction at a new observation x: Let $\hat{C}_b(x)$ be the class prediction of the $b-$th random forest tree. Then $\hat{C}_{rf}^{B}(x) =$ the majority vote$\hat{C}_b(x)_1^B$. That is, take the majority vote of the outcomes of each tree.

3 Data Exploration and Overview

We have 15 columns and 32560 rows in dataset including 14 predictors and an outcome of the level of the income, either higher than 50,000 dollars or lower than 50,000. The predictors include 6 continuous and 9 categorical variables. See figure 1 for an overview of the income level:

List of response:

income level:>50K, <=50K.

List of predictors:

Continuous:

age, fnlwgt, education-num, capital-gain, capital-loss, hours-per-week

Categorical:

workclass: Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked.

education: Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool.

marital-status: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse.

occupation: Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces.

relationship: Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried.

race: White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black.

sex: Female, Male.

native-country: United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad&Tobago, Peru, Hong, Holand-Netherlands.

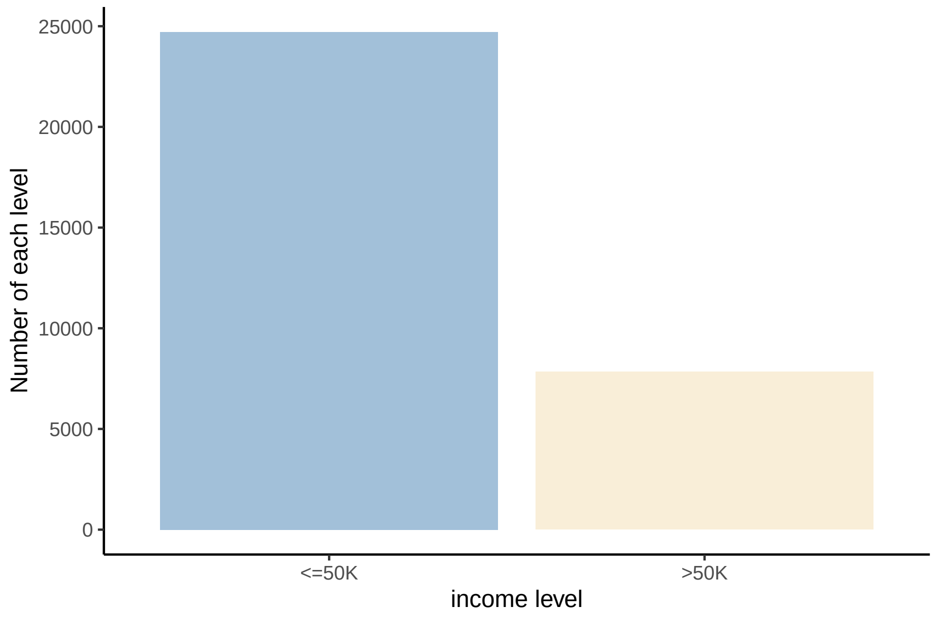

Figure 1:

4 Analysis

Since a random forest does not have any distributional assumptions, our next step was to build a random forest directly. R has a function called **randomForest** which constructs a random forest based on the algorithm described earlier. One of the arguments that **randomForest** takes in is called m_{try}. This is the number of predictors that are considered when building a tree out of all p predictors. Since our data has $p = 14$ predictors, we orginally wanted to consider $\sqrt{p} = 3.74 \approx 4$ predictors. However, we wanted to try out different values of m_{try} to see what value yielded the best model. We varied m_{try} from 1 to 13 and built random forest models keeping number of trees ($ntree$) at 50. We then calculated the model accuracy to see which value of m_{try} yielded the best model. We calculated the model accuracy (number of correct predictions/total predictions) from the confusion matrix that the **randomForest** generates. After getting the best number of predictors, we wanted to see what number of trees yielded the best model. We chose $n_{tree} = 50, 100, 200, 500$ to test.

5 Outcome and Conclusion

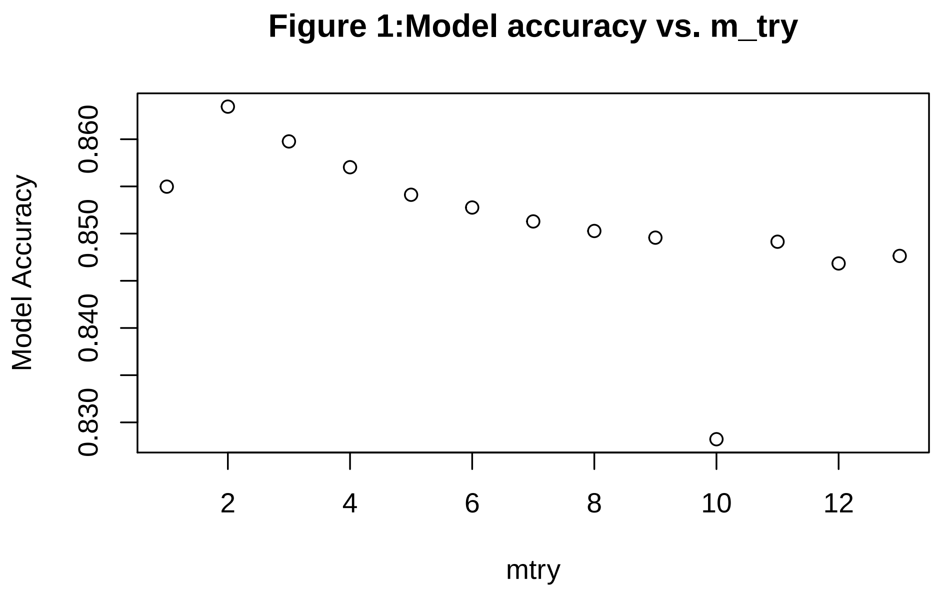

Figure 2:

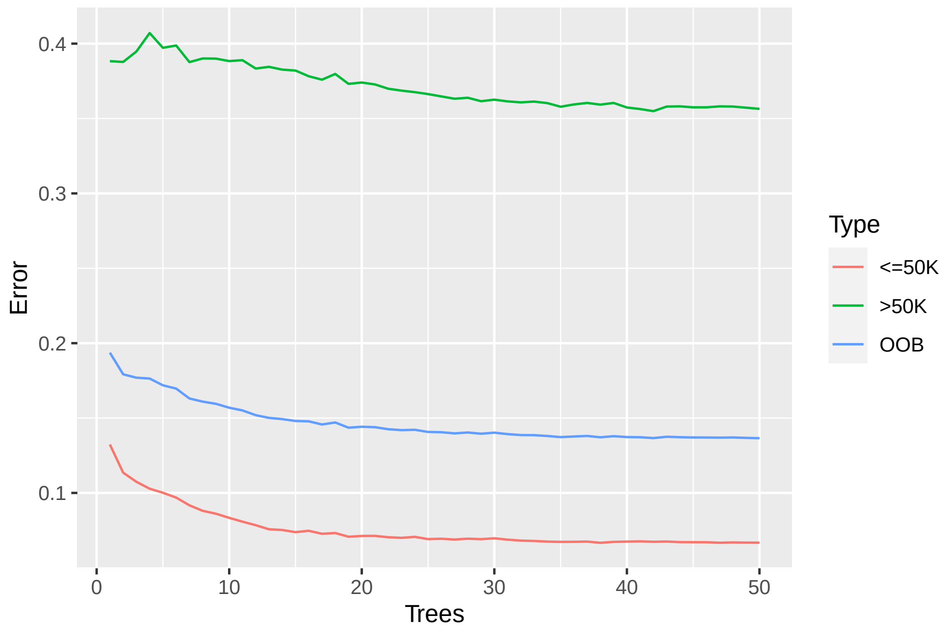

Figure 3:

Table 1: Confusion matrices of different m_{try}

		Actual	Class			Actual	Class
mtry=1		$\leq 50K$	$> 50K$	mtry=2		$\leq 50K$	$> 50K$
Predicted	$\leq 50K$	23620	1099	Predicted	$\leq 50K$	23068	1651
Class	$> 50K$	3623	4218	Class	$> 50K$	2795	5046

		Actual	Class			Actual	Class
mtry=3		$\leq 50K$	$> 50K$	mtry=4		$\leq 50K$	$> 50K$
Predicted	$\leq 50K$	22984	1735	Predicted	$\leq 50K$	22860	1859
Class	$> 50K$	2831	5010	Class	$> 50K$	2796	5045

From the plot of the accuracy above we see that at $m_{try} = 2$ the model accuracy is the highest, and thus we chose $m_{try} = 2$. Also, we can see the confusion matrices and find that $m_{try} = 2$ has the best accuracy. After $m_{try} = 2$, the error increases.

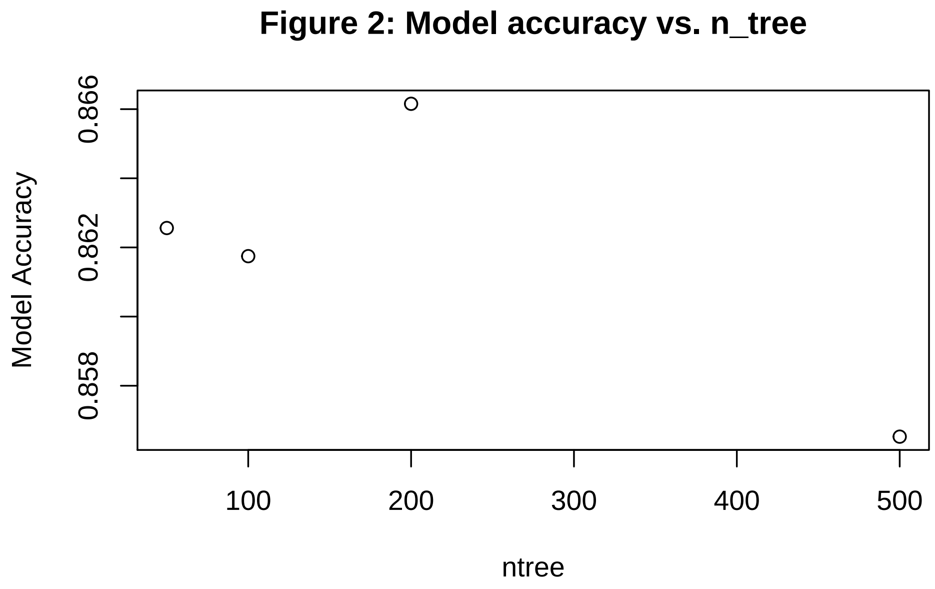

Figure 4:

From the plot we see that $n_{tree} = 200$ yields the highest model accuracy; we chose $n_{tree} = 200$. Therefore, we concluded that the best model was generated at $m_{try} = 2$ and $n_{tree} = 200$. This result is supported from **Figure 5** below as the minimum error is reached at $n_{tree} = 200$.

Table 3: Confusion matrices of different n_{tree}

		Actual	Class			Actual	Class
ntree=50		$\leq 50K$	$> 50K$	ntree=100		$\leq 50K$	$> 50K$
Predicted	$\leq 50K$	23169	1550	Predicted	$\leq 50K$	23180	1539
Class	$> 50K$	2925	4916	Class	$> 50K$	2866	4975

		Actual	Class			Actual	Class
ntree=200		$\leq 50K$	$> 50K$	ntree=500		$\leq 50K$	$> 50K$
Predicted	$\leq 50K$	23164	1555	Predicted	$\leq 50K$	23233	1486
Class	$> 50K$	2803	5038	Class	$> 50K$	2866	4975

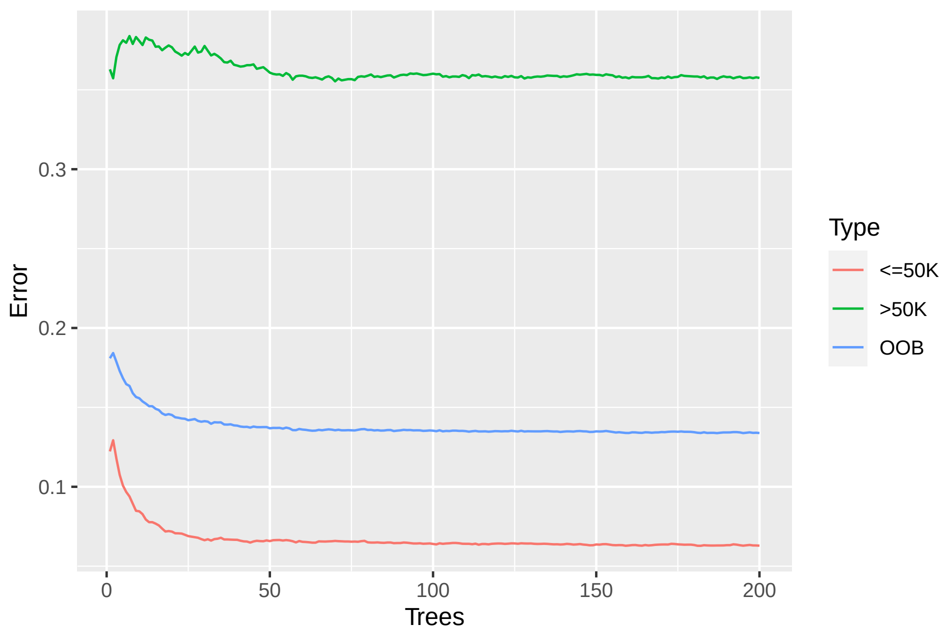

Figure 5:

From the table of confusion matrices, for $n_{tree} = 200$, we see that the model predicted 1555 times incorrectly that an individual had income of less than or equal to 50K when the true income was greater than 50K. This results in an error rate of $\dfrac{1555}{1555 + 2803 + 23164 + 5038} \approx 0.048$. On the other hand the forest predicted 2803 times incorrectely that an individual had an income > 50K when the true income was $\leq$ 50K. This gives us an error rate of $\dfrac{2803}{1555 + 2803 + 23164 + 5038} \approx 0.086$. Overall both error rates are below 10 percent; we concluded that the model performance is decent.

6 Summary

A random forest consists of a large number of decision trees. Each decision tree makes its own prediction, and we combine these predicitons to make a grand prediction on the observation that is being classified. This is called "voting" since the observation will be classified as the category with the most votes.

Random forest also has various parameters that we can adjust in order to optimize the fit of the model on the input data. Their parameters are similar to the one in decision tree along with the number of trees, random splits of the numerical attributes instead of optimization, prediction on subset ratios, subset ratio, voting strategy (confidence vs majority), and so on.

In our project, we emphasize the process of building a random forest model. We can see the random forest gives us a good outcome of classification in terms of error rate. When we choose the tree of $n_{tree} = 200$ and $m_{try} = 2$, we have 86.61% accuracy in terms of our huge data. We found the method is very similar to the bootstrapping method and very powerful without any assumptions or limitations when applying.

Random forest has some drawbacks such as that the running speed is very long. Also, the algorithm of the random forest may be more complex than what we have learned so far, which means we still have a long way to go to study the theory behind that.

7 Reference

Breiman, L. (2001, January). RANDOM FORESTS - Statistics at UC Berkeley. Retrieved from https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf

Hastie, T., Tibshirani, R., & Friedman, J. H. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. (2nd ed.) New York: Springer.

Horsley, S. (2020, May 8). One For The History Books: 14.7% Unemployment, 20.5 Million Jobs Wiped Away. Retrieved from https://www.npr.org/sections/coronavirus-live-updates/2020/05/08/852430930/one-for-the-history-books-14-7-unemployment-20-5-million-jobs-wiped-away

James G., Witten D., Hastie T., Tibshirani R. (2017). An introduction to statistical learning : with applications in R.(8th ed) New York :Springer. Retrieved from https://faculty.marshall.usc.edu/gareth-james/ISL/ISLR%20Seventh%20Printing.pdf

Killian, J. A. (2020, January). Census Income Data Set. Retrieved from https://archive.ics.uci.edu/ml/datasets/Census Income

8 Appendix

```
library(dplyr)
library(tidyverse)
library(ggplot2)
library(rpart)
library(rattle)
library(RColorBrewer)
library(randomForest)

dat=read.csv("income.csv")
dat=data.frame(dat)
dat$X..50K=as.factor(dat$X..50K)
dat$State.gov=as.factor(dat$State.gov)
dat$Bachelors=as.factor(dat$Bachelors)
dat$Never.married=as.factor(dat$Never.married)
dat$Adm.clerical=as.factor(dat$Adm.clerical)
dat$Not.in.family=as.factor(dat$Not.in.family)
dat$White=as.factor(dat$White)
dat$Male=as.factor(dat$Male)
dat$United.States=as.factor(dat$United.States)
set.seed(4052)

ggplot(dat, aes(x = factor(X..50K))) +
    geom_bar(fill = c("steelblue","wheat"), alpha=0.5) +
    theme_classic() +

    xlab("income level") +

    ylab("Number of each level")


#head(dat)


m = c(1,2,3,4,5,6,7,8,9,10,11,12,13)


#mod 1 mtry =1, ntree =50

mod1=randomForest(X..50K~.,data=dat,ntree=50,mtry=1,importance=TRUE)


#mod 2 mtry = 2, n tree = 50

mod2=randomForest(X..50K~.,data=dat,ntree=50,mtry=2,importance=TRUE)


#mod 3 mtry = 3, ntree =50

mod3=randomForest(X..50K~.,data=dat,ntree=50,mtry=3,importance=TRUE)


#mod 4 mtry=4 ntree = 50

mod4=randomForest(X..50K~.,data=dat,ntree=50,mtry=4,importance=TRUE)


#mod 5 mtry =5 ntree =50

mod5=randomForest(X..50K~.,data=dat,ntree=50,mtry=5,importance=TRUE)


#mod 6 mtry =6 ntree =50

mod6=randomForest(X..50K~.,data=dat,ntree=50,mtry=6,importance=TRUE)


#mod 7 mtry =7 ntree =50
```

```
mod7=randomForest(X..50K~.,data=dat,ntree=50,mtry=7,importance=TRUE)

#mod 8 mtry =8 ntree =50
mod8=randomForest(X..50K~.,data=dat,ntree=50,mtry=8,importance=TRUE)

#mod 9 mtry =9 ntree =50
mod9=randomForest(X..50K~.,data=dat,ntree=50,mtry=9,importance=TRUE)

#mod 10 mtry =10 ntree =50
mod10=randomForest(X..50K~.,data=dat,ntree=50,mtry=10,importance=TRUE)


#mod 11 mtry =11 ntree =50
mod11=randomForest(X..50K~.,data=dat,ntree=50,mtry=11,importance=TRUE)
#mod 12 mtry =12 ntree =50
mod12=randomForest(X..50K~.,data=dat,ntree=50,mtry=12,importance=TRUE)
#mod 13 mtry =13 ntree =50
mod13=randomForest(X..50K~.,data=dat,ntree=50,mtry=13,importance=TRUE)


mod1.acc = (mod1$confusion[1,1] + mod1$confusion[2,2])/
(mod1$confusion[1,1] + mod1$confusion[1,2]+ mod1$confusion[2,1]+
mod1$confusion[2,2])

mod2.acc = (mod2$confusion[1,1] + mod2$confusion[2,2])/
(mod2$confusion[1,1] + mod2$confusion[1,2]+ mod2$confusion[2,1]+
mod2$confusion[2,2])
```

```
mod3.acc = (mod3$confusion[1,1] + mod3$confusion[2,2])/
(mod3$confusion[1,1] + mod3$confusion[1,2]+ mod3$confusion[2,1]+
mod3$confusion[2,2])

mod4.acc = (mod4$confusion[1,1] + mod4$confusion[2,2])/
(mod4$confusion[1,1] + mod4$confusion[1,2]+ mod4$confusion[2,1]+
mod4$confusion[2,2])


mod5.acc = (mod5$confusion[1,1] + mod5$confusion[2,2])/
(mod5$confusion[1,1] + mod5$confusion[1,2]+ mod5$confusion[2,1]+
mod5$confusion[2,2])

mod6.acc = (mod6$confusion[1,1] + mod6$confusion[2,2])/
(mod6$confusion[1,1] + mod6$confusion[1,2]+ mod6$confusion[2,1]+
mod6$confusion[2,2])

mod7.acc = (mod7$confusion[1,1] + mod7$confusion[2,2])/
(mod7$confusion[1,1] + mod7$confusion[1,2]+ mod7$confusion[2,1]+
mod7$confusion[2,2])

mod8.acc = (mod8$confusion[1,1] + mod8$confusion[2,2])/
(mod8$confusion[1,1] + mod8$confusion[1,2]+ mod8$confusion[2,1]+
mod8$confusion[2,2])

mod9.acc = (mod9$confusion[1,1] + mod9$confusion[2,2])/
```

```
(mod9$confusion[1,1] + mod9$confusion[1,2]+ mod9$confusion[2,1]+
mod9$confusion[2,2])


mod10.acc = (mod10$confusion[1,1] + mod10$confusion[2,2])/
(mod10$confusion[1,1] + mod10$confusion[1,2]+ mod1$confusion[2,1]+
mod10$confusion[2,2])


mod11.acc = (mod11$confusion[1,1] + mod11$confusion[2,2])/
(mod11$confusion[1,1] + mod11$confusion[1,2]+ mod11$confusion[2,1]+
mod11$confusion[2,2])


mod12.acc = (mod12$confusion[1,1] + mod12$confusion[2,2])/
(mod12$confusion[1,1] + mod12$confusion[1,2]+ mod12$confusion[2,1]+
mod12$confusion[2,2])


mod13.acc = (mod13$confusion[1,1] + mod13$confusion[2,2])/
(mod13$confusion[1,1] + mod13$confusion[1,2]+ mod13$confusion[2,1]+
mod13$confusion[2,2])



mod.acc = c(mod1.acc, mod2.acc, mod3.acc, mod4.acc, mod5.acc, mod6.acc,mod7.acc,mod8.acc,mo


#plot of model accuracy
plot(mod.acc ~ m, xlab = "mtry", ylab = "Model Accuracy",
main = "Figure 1:Model accuracy vs. m_try")


ntree = c(50, 100 , 200, 500)
```

```
#mod 1.1 mtry =3, ntree =50

mod1.1=randomForest(X..50K~.,data=dat,ntree=50,mtry=3,importance=TRUE)


#mod 2.1 mtry = 3, n tree = 100

mod2.1=randomForest(X..50K~.,data=dat,ntree=100,mtry=3,importance=TRUE)


#mod 3.3 mtry = 3, ntree =200

mod3.1=randomForest(X..50K~.,data=dat,ntree=200,mtry=3,importance=TRUE)


#mod 4.1 mtry=3 ntree = 500

mod4.1=randomForest(X..50K~.,data=dat,ntree=500,mtry=3,importance=TRUE)


mod1.1.acc = (mod1.1$confusion[1,1] + mod1.1$confusion[2,2])/

(mod1.1$confusion[1,1] + mod1.1$confusion[1,2]+ mod1.1$confusion[2,1]+

mod1.1$confusion[2,2])


mod2.1.acc = (mod2.1$confusion[1,1] + mod2.1$confusion[2,2])/

(mod2.1$confusion[1,1] + mod2$confusion[1,2]+ mod2.1$confusion[2,1]+

mod2.1$confusion[2,2])


mod3.1.acc = (mod3.1$confusion[1,1] + mod3.1$confusion[2,2])/

(mod3.1$confusion[1,1] + mod3.1$confusion[1,2]+ mod3.1$confusion[2,1]+

mod3.1$confusion[2,2])


mod4.1.acc = (mod4.1$confusion[1,1] + mod4.1$confusion[2,2])/

(mod4.1$confusion[1,1] + mod4$confusion[1,2]+ mod4.1$confusion[2,1]+

mod4.1$confusion[2,2])
```

```
mod.acc.1= c(mod1.1.acc, mod2.1.acc, mod3.1.acc, mod4.1.acc)


#plot of model accuracy

plot(mod.acc.1 ~ ntree, xlab = "ntree", ylab = "Model Accuracy",

main = "Figure 2: Model accuracy vs. n_tree")
```