INFX 573 Problem Set 8 - Prediction

Miloni Desai

Due: Tuesday, November 26, 2019

Collaborators: Stack Overflow

Instructions:

Before beginning this assignment, please ensure you have access to R and RStudio.

1. Download the `problemset8.rmd` file from Canvas. Open `problemset8.rmd` in RStudio and supply your solutions to the assignment by editing `problemset8.rmd`.
2. Replace the "Insert Your Name Here" text in the `author:` field with your own full name. Any collaborators must be listed on the top of your assignment.
3. Be sure to include well-documented (e.g. commented) code chucks, figures and clearly written text chunk explanations as necessary. Any figures should be clearly labeled and appropriately referenced within the text.
4. Collaboration on problem sets is acceptable, and even encouraged, but each student must turn in an individual write-up in his or her own words and his or her own work. The names of all collaborators must be listed on each assignment. Do not copy-and-paste from other students' responses or code.
5. When you have completed the assignment and have **checked** that your code both runs in the Console and knits correctly when you click `Knit PDF`, rename the R Markdown file to `ps8_YourLastName_YourFirstName.rmd`, knit a PDF and submit the PDF file on Canvas.

Setup:

In this problem set you will need, at minimum, the following R packages.

```
# Load standard libraries
library(tidyverse)
library(gridExtra)
library(MASS)
library(pROC)
library(arm)
library(randomForest)
library(Metrics)
```

Data: In this problem set we will use the `flights` and `titanic` datasets used previously in class. The flights dataset (via the the *nycflights13* library) contains information on flight delays and weather. Titanic text file contains data about the survival of passengers aboard the Titanic. Table 1 contains a description of this data.

```
#Load data
titanic_data <- read.csv("imt573/Data/titanic.csv")
str(titanic_data) # explore data structure
```

```
## 'data.frame':    1309 obs. of  14 variables:
##  $ pclass   : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ survived : int  1 1 0 0 0 1 1 0 1 0 ...
##  $ name     : Factor w/ 1307 levels "Abbing, Mr. Anthony",..: 22 24 25 26 27 31 46 47 51 55 ...
```

```
## $ sex      : Factor w/ 2 levels "female","male": 1 2 1 2 1 2 1 2 1 2 ...
## $ age      : num  29 0.917 2 30 25 ...
## $ sibsp    : int  0 1 1 1 1 0 1 0 2 0 ...
## $ parch    : int  0 2 2 2 2 0 0 0 0 0 ...
## $ ticket   : Factor w/ 929 levels "110152","110413",..: 188 50 50 50 50 125 93 16 77 826 ...
## $ fare     : num  211 152 152 152 152 ...
## $ cabin    : Factor w/ 187 levels "","A10","A11",..: 45 81 81 81 81 151 147 17 63 1 ...
## $ embarked : Factor w/ 4 levels "","C","Q","S": 4 4 4 4 4 4 4 4 4 2 ...
## $ boat     : Factor w/ 28 levels "","1","10","11",..: 13 4 1 1 1 14 3 1 28 1 ...
## $ body     : int  NA NA NA 135 NA NA NA NA NA 22 ...
## $ home.dest: Factor w/ 370 levels "","?Havana, Cuba",..: 310 232 232 232 232 238 163 25 23 230 ...
```

```r
summary(titanic_data)
```

```
##     pclass         survived                                    name
##  Min.   :1.000   Min.   :0.000   Connolly, Miss. Kate            :   2
##  1st Qu.:2.000   1st Qu.:0.000   Kelly, Mr. James                :   2
##  Median :3.000   Median :0.000   Abbing, Mr. Anthony             :   1
##  Mean   :2.295   Mean   :0.382   Abbott, Master. Eugene Joseph   :   1
##  3rd Qu.:3.000   3rd Qu.:1.000   Abbott, Mr. Rossmore Edward     :   1
##  Max.   :3.000   Max.   :1.000   Abbott, Mrs. Stanton (Rosa Hunt):   1
##                                  (Other)                         :1301
##      sex           age              sibsp            parch
##  female:466   Min.   : 0.1667   Min.   :0.0000   Min.   :0.000
##  male  :843   1st Qu.:21.0000   1st Qu.:0.0000   1st Qu.:0.000
##               Median :28.0000   Median :0.0000   Median :0.000
##               Mean   :29.8811   Mean   :0.4989   Mean   :0.385
##               3rd Qu.:39.0000   3rd Qu.:1.0000   3rd Qu.:0.000
##               Max.   :80.0000   Max.   :8.0000   Max.   :9.000
##               NA's   :263
##       ticket          fare                        cabin      embarked
##  CA. 2343:  11   Min.   :  0.000                 :1014    :  2
##  1601    :   8   1st Qu.:  7.896   C23 C25 C27    :   6   C:270
##  CA 2144 :   8   Median : 14.454   B57 B59 B63 B66:   5   Q:123
##  3101295 :   7   Mean   : 33.295   G6             :   5   S:914
##  347077  :   7   3rd Qu.: 31.275   B96 B98        :   4
##  347082  :   7   Max.   :512.329   C22 C26        :   4
##  (Other) :1261   NA's   :1         (Other)        : 271
##       boat          body                      home.dest
##         :823   Min.   :  1.0                       :564
##  13     : 39   1st Qu.: 72.0   New York, NY        : 64
##  C      : 38   Median :155.0   London              : 14
##  15     : 37   Mean   :160.8   Montreal, PQ        : 10
##  14     : 33   3rd Qu.:256.0   Cornwall / Akron, OH:  9
##  4      : 31   Max.   :328.0   Paris, France       :  9
##  (Other):308   NA's   :1188    (Other)             :639
```

```r
titanic_data$pclass <- as.factor(titanic_data$pclass)
titanic_data$survived <- as.factor(titanic_data$survived)
```

Variable	Description
pclass	Passenger Class (1 = 1st; 2 = 2nd; 3 = 3rd)
survived	Survival (0 = No; 1 = Yes)
name	Name
sex	Sex
age	Age
sibsp	Number of Siblings/Spouses Aboard
parch	Number of Parents/Children Aboard
ticket	Ticket Number
fare	Passenger Fare
cabin	Cabin
embarked	Port of Embarkation (C = Cherbourg; Q = Queenstown; S = Southampton)
boat	Lifeboat
body	Body Identification Number
home.dest	Home/Destination

Table 1: Description of variables in the Titanic Dataset

As part of this assignment, we will evaluate the performance of several statistical learning methods. We will fit our learning models using a set of *training* observations and measure its performance on a set of *test* observations.

1. Discuss the advantages of using a training/test split when evaluating statistical models.

#The advantages of the training/test split is that it lets us cross validate the statistical model. Without this validation neither can we assess the model's performance, nor can we determine if the predictions it produces are accurate.Additionally, this cross validation in random forest is also used for optimal model selection. The random splitting adn set seed function allow for randomization and reproducibility.

Predictions with a continuous output variable

2. Load in the flights dataset. Join the flights data to the weather data based on the departure location, date, and hour of the flight. Exclude data entries which cannot be joined to weather data. Copy the joined data so we can refer to it later.

```
# Load data
library(nycflights13)

fw<-flights%>%
  left_join(weather)
```

```
## Joining, by = c("year", "month", "day", "origin", "hour", "time_hour")
```

3. From the joined data, keep only the following columns as we build our first model: departure delay, origin, departure time, temperature, wind speed, precipitation, and visibility. Omit observations that do not have all of these variables present.

```
#Getting rid of rows with null values
fw2<-fw[complete.cases(fw), ]
```

4. Split your data into a *training* and *test* set based on an 80-20 split. In other words, 80% of the observations will be in the training set and 20% will be in the test set. Remember to set the random seed.

```
#split dataset
train_size <- floor(0.80 * nrow(fw2))
set.seed(250)
train_index <- sample(seq_len(nrow(fw2)), size = train_size)
flight.train <- fw2[train_index, ]
flight.test <- fw2[-train_index, ]
```

5. Build a linear regression model to predict departure delay using the subset of variables indicated in (3.). What is the RMSE on the training set? What is the RMSE on the test set? Which is higher and is this expected?

```
#departure delay, origin, departure time, temperature, wind speed, precipitation, and visibility.
dep_delay_test.model <- (lm(dep_delay ~ temp + wind_speed + precip + visib, data = flight.test))
test<-summary(dep_delay_test.model)
RSS <- c(crossprod(test$residuals))
MSE <- RSS / length(test$residuals)
RMSE <- sqrt(MSE)
RMSE
```

```
## [1] 37.50664
```

```
#training data
dep_delay_train.model <- (lm(dep_delay ~ temp + wind_speed + precip + visib, data = flight.train))
summary(dep_delay_train.model)
```

```
##
## Call:
## lm(formula = dep_delay ~ temp + wind_speed + precip + visib,
##     data = flight.train)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -59.82 -17.04 -12.50  -0.53 738.87
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 33.458867   1.835009  18.234   <2e-16 ***
## temp         0.154404   0.008646  17.858   <2e-16 ***
## wind_speed   0.297415   0.033889   8.776   <2e-16 ***
## precip       1.239533  13.294749   0.093    0.926
## visib       -3.502656   0.162691 -21.530   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 37.94 on 58182 degrees of freedom
## Multiple R-squared:  0.01558,    Adjusted R-squared:  0.01551
## F-statistic: 230.2 on 4 and 58182 DF,  p-value: < 2.2e-16
```

#Thus, we can see that the RMSE value for test data is 37.51 while that for training data is 37.94. Thus RMSE_training > RMSE_testing. Here in our case, this is expected, as we do not expect the model to overfit.

6. Now, improve upon these prediction results by including additional variables in your model. Make sure you keep at least 95% of original data (i.e. about 320K observations across both the training and test datasets). Do not include the arrival time, scheduled arrival time, or the arrival delay in your model. Use the same observations as above for the training and test sets (i.e. keep the same rows but add different variables/columns at your discretion). Can you improve upon the training RMSE? Once you have a model that you feel adequately improves the training RMSE, does your model improve the test RMSE? Which variables did you include in your model?

```r
dep_delay1.model <- (lm(dep_delay ~ dep_time + sched_dep_time + temp + pressure + wind_speed + precip +
summary(dep_delay1.model)
```

```
##
## Call:
## lm(formula = dep_delay ~ dep_time + sched_dep_time + temp + pressure +
##     wind_speed + precip + visib + wind_dir + wind_gust + dewp,
##     data = flight.test)
##
## Residuals:
##    Min      1Q Median     3Q    Max
## -53.89 -14.65  -7.33   1.55 882.95
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    133.590151  46.508767   2.872 0.004080 **
## dep_time         0.115020   0.002412  47.682  < 2e-16 ***
## sched_dep_time  -0.097598   0.002538 -38.454  < 2e-16 ***
## temp            -0.346080   0.038518  -8.985  < 2e-16 ***
## pressure        -0.129810   0.044778  -2.899 0.003750 **
## wind_speed      -0.069028   0.119373  -0.578 0.563101
## precip          41.474431  22.816118   1.818 0.069120 .
## visib           -1.167807   0.322423  -3.622 0.000293 ***
## wind_dir        -0.013908   0.003990  -3.486 0.000492 ***
## wind_gust        0.225225   0.103413   2.178 0.029428 *
## dewp             0.421277   0.039077  10.781  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 34.12 on 14536 degrees of freedom
## Multiple R-squared:  0.1844, Adjusted R-squared:  0.1838
## F-statistic: 328.6 on 10 and 14536 DF,  p-value: < 2.2e-16
```

```r
dep_delay2.model <- (lm(dep_delay ~ dep_time + sched_dep_time + temp + pressure + wind_speed + precip +
summary(dep_delay2.model)
```

```
##
## Call:
## lm(formula = dep_delay ~ dep_time + sched_dep_time + temp + pressure +
##     wind_speed + precip + visib + wind_dir + wind_gust + dewp,
##     data = flight.train)
##
## Residuals:
##    Min      1Q Median     3Q    Max
## -70.61 -14.26  -7.37   1.38 882.53
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    166.893010  23.498053   7.102 1.24e-12 ***
## dep_time         0.132021   0.001265 104.341  < 2e-16 ***
```

```
## sched_dep_time  -0.115493   0.001329 -86.913  < 2e-16 ***
## temp            -0.312157   0.019375 -16.111  < 2e-16 ***
## pressure        -0.153187   0.022610  -6.775 1.25e-11 ***
## wind_speed      -0.143188   0.060021  -2.386  0.01705 *
## precip          38.422656  12.050654   3.188  0.00143 **
## visib           -2.058932   0.154107 -13.360  < 2e-16 ***
## wind_dir        -0.011626   0.001987  -5.851 4.92e-09 ***
## wind_gust        0.250597   0.051983   4.821 1.43e-06 ***
## dewp             0.375264   0.019589  19.157  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 34.12 on 58176 degrees of freedom
## Multiple R-squared:  0.2038, Adjusted R-squared:  0.2036
## F-statistic:  1489 on 10 and 58176 DF,  p-value: < 2.2e-16
```

#Thus we improved both the training and testing RMSE. We did this by including other variables that might influence departure delay. The variables included are : departure time, scheduled departure time, pressure, wind direction, gust of wind and dew. The new RMSEs for both testing and training data are 34.12, smaller than before

Predictions with a categorical output (classification)

7. Load in the titanic data. Split your data into a *training* and *test* set based on an 80-20 split. In other words, 80% of the observations will be in the training set and 20% will be in the test set. Remember to set the random seed.

```
titanic_data <- read.csv('imt573/Data/titanic.csv')
train_size <- floor(0.80 * nrow(titanic_data))
set.seed(250)
train_index <- sample(seq_len(nrow(titanic_data)), size = train_size)
titanic.train <- titanic_data[train_index, ]
titanic.test <- titanic_data[-train_index, ]
```

In this problem set our goal is to predict the survival of passengers. First, let's train a logistic regression model for survival that controls for the socioeconomic status of the passenger.

8. Fit the model described above (i.e. one that only takes into account socioeconomic status) using the `glm` function in R.

```
fit.glm <- glm(survived ~ pclass, family = "binomial", data = titanic.train)
summary(fit.glm)
```

```
##
## Call:
## glm(formula = survived ~ pclass, family = "binomial", data = titanic.train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3800  -0.7812  -0.7812   0.9875   1.6344
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   1.2120     0.1842   6.580 4.71e-11 ***
## pclass       -0.7475     0.0781  -9.572  < 2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1398.3  on 1046  degrees of freedom
## Residual deviance: 1301.5  on 1045  degrees of freedom
## AIC: 1305.5
##
## Number of Fisher Scoring iterations: 4
```

9. What might you conclude based on this model about the probability of survival for lower class passengers?

##Survival = -0.7753 x pclass + 1.2564, p-value << alpha - 0.05 giving us the significance of the association. The slope of the fitted regression model is negative, hence the passengers' classes (level) are negatively associated with chance of survival. Higher levels are represented by smaller numbers, the larger the class variable ,lower class of the passengers. Thus,lower class passengers have lower probability of survival.

Next, let's consider the performance of this model.

10. Predict the survival of passengers for each observation in your test set using the model fit in Problem 2. Save these predictions as `yhat`.

```
prob.glm <- predict(fit.glm, titanic.test, type = "response")
yhat <- rep(0, 262)
yhat[prob.glm >.5] <- 1
obs <- titanic.test$survived
table(yhat, obs)
```

```
##     obs
## yhat   0   1
##    0 150  59
##    1  18  35
```

```
mean(yhat == obs)
```

```
## [1] 0.7061069
```

11. Use a threshold of 0.5 to classify predictions. What is the number of false positives on the test data? Interpret this in your own words.

```
#The number of false positives is 24. In other words, there are 24 misclassifications where the predict
```

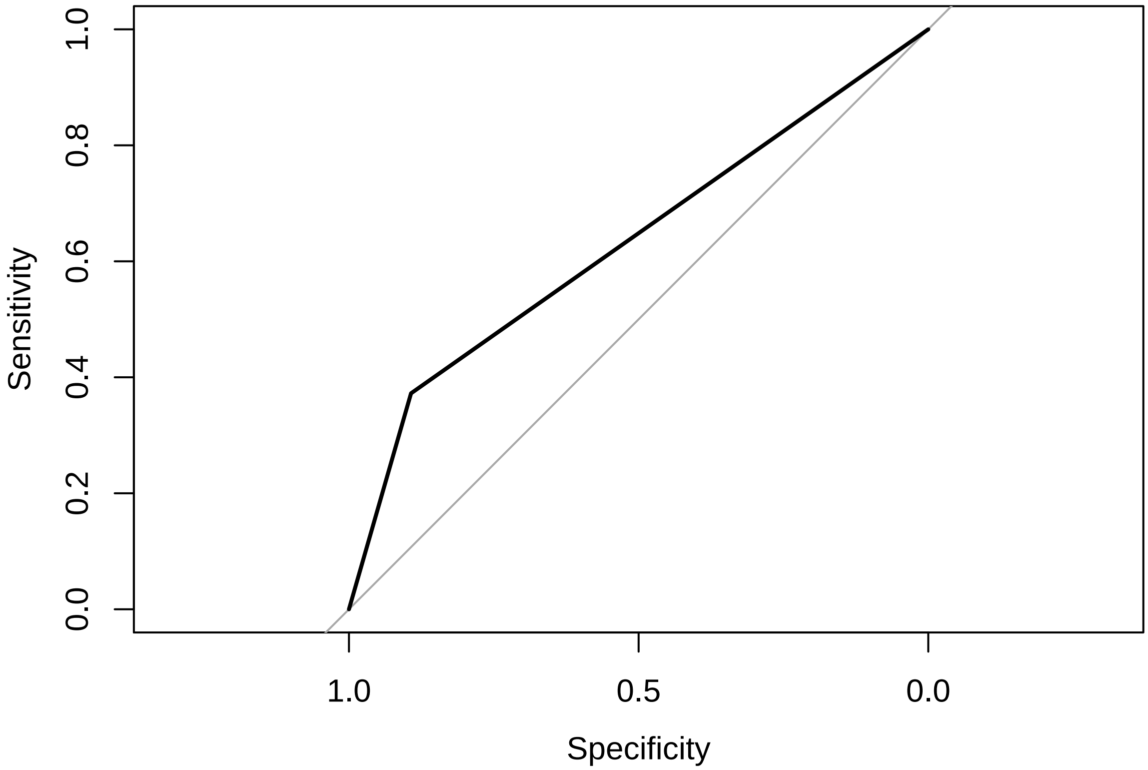
12. Using the `roc` function, plot the ROC curve for this model. Discuss what you find.

```
#plot roc
roc <- roc(obs, yhat)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
plot.roc(roc)
```



##The ROC curve value implies that the prediction accuracy is not very high. ROC curve is close to the 45-degree diagonal line of the ROC space, indicating low accuracy (low sensitivity and specificity) but still better than random guessing basedline. The area under the curve is a measure of text accuracy, which equals to 0.6128. Again, better than random guessing (0.5) but not particularly high.

13. Suppose we use the data to construct a new predictor variable based on a passenger's listed title (i.e. Mr., Mrs., Miss., Master). Why might this be an interesting variable to help predict passenger survival?

Use the following custom function to add this predictor to your dataset. ##Title indicates both gender and marital status in common cases. Some times title could also indicate passenger class, e.g. Master.Thus it is an interesting variable to predict survival

```
# Making a feature that includes more titles
titanic_data$title[grep("Mr.", titanic_data$name)] <- "Mr."
titanic_data$title[grep("Mrs.", titanic_data$name)] <- "Mrs."
titanic_data$title[grep("Miss.", titanic_data$name)] <- "Miss."
titanic_data$title[grep("Master", titanic_data$name)] <- "Master"
titanic_data$title <- as.factor(titanic_data$title)
```

14. Fit a second logistic regression model including this new feature. Use the `summary` function to look at the model. Did this new feature improve the model?

```
#split dataset
set.seed(250)
train_index <- sample(seq_len(nrow(titanic_data)), size = train_size)
titanic.train <- titanic_data[train_index, ]
titanic.test <- titanic_data[-train_index, ]
#train logistic regression model
fit.glm.title <- glm(survived ~ title, family = "binomial", data = titanic.train)
```

```r
summary(fit.glm.title)
```

```
##
## Call:
## glm(formula = survived ~ title, family = "binomial", data = titanic.train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7492  -0.6151  -0.6151   0.8716   1.8752
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.3023     0.3198   0.945  0.34462
## titleMiss.    0.4698     0.3523   1.333  0.18241
## titleMr.     -1.8713     0.3372  -5.550 2.86e-08 ***
## titleMrs.     0.9835     0.3740   2.630  0.00854 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1367.5  on 1023  degrees of freedom
## Residual deviance: 1048.4  on 1020  degrees of freedom
##   (23 observations deleted due to missingness)
## AIC: 1056.4
##
## Number of Fisher Scoring iterations: 4
```

15. Comment on the overall fit of this model. For example, you might consider exploring when misclassification occurs.

```r
#Miss - seems to have no association with survival status, thus classification for class "Miss" would b
```

16. Predict the survival of passengers for each observation in your test data using the new model. Save these predictions as `yhat2`.

```r
#use trained logistic regression model to predict test set
prob.glm <- predict(fit.glm.title, titanic.test, type = "response")
yhat2 <- rep(0, 262)
yhat2[prob.glm >.5] <- 1
obs <- titanic.test$survived
table(yhat2, obs)
```

```
##      obs
## yhat2   0   1
##     0 130  21
##     1  38  73
```

```r
mean(yhat2 == obs)
```

```
## [1] 0.7748092
```

Random forests

Another very popular classifier used in data science is called a *random forest*[1].

[1] https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm

17. Use the `randomForest` function to fit a random forest model with passenger class and title as predictors. Make predictions for the test set using the random forest model. Save these predictions as `yhat3`.

```
#train random forest model
titanic.train <- titanic.train[!(is.na(titanic.train$title) | titanic.train$title==""), ]
fit.rf <- randomForest(survived ~ pclass + title, data = titanic.train, mtry = 2, importance =  TRUE)
```

```
## Warning in randomForest.default(m, y, ...): The response has five or fewer
## unique values. Are you sure you want to do regression?
```

```
fit.rf
```

```
##
## Call:
##  randomForest(formula = survived ~ pclass + title, data = titanic.train,      mtry = 2, importance =
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##           Mean of squared residuals: 0.1457736
##                     % Var explained: 38.59
```

```
yhat3 = predict(fit.rf, newdata = titanic.test, type = "response")
```

18. Develop your own random forest model (i.e. add/remove variables at your discretion), attempting to improve the model performance. Make predictions for the test set using your new random forest model. Save these predictions as `yhat4`.

```
#train random forest model
titanic.train <- titanic.train[!(is.na(titanic.train$age) | titanic.train$age==""), ]
titanic.train <- titanic.train[!(is.na(titanic.train$fare) | titanic.train$fare==""), ]
fit.rf <- randomForest(survived ~ pclass + title + sex + age + sibsp + parch + fare + embarked, data =
```

```
## Warning in randomForest.default(m, y, ...): The response has five or fewer
## unique values. Are you sure you want to do regression?
```

```
fit.rf
```

```
##
## Call:
##  randomForest(formula = survived ~ pclass + title + sex + age +      sibsp + parch + fare + embarked
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 8
##
##           Mean of squared residuals: 0.1484597
##                     % Var explained: 38.41
```

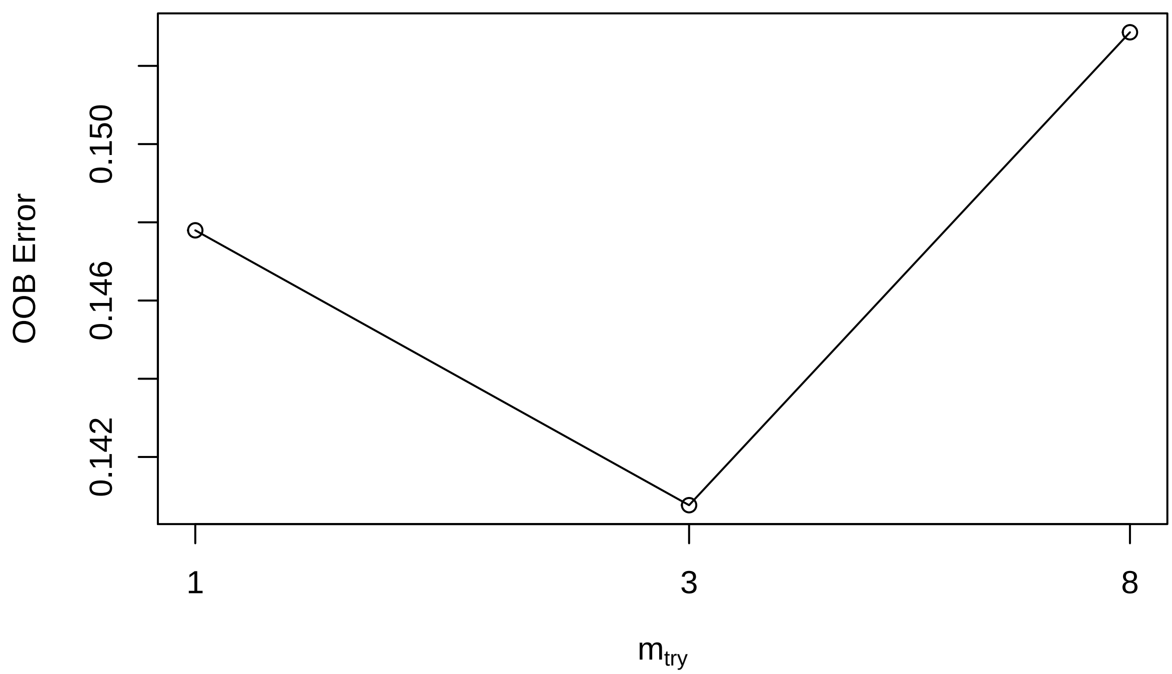
```
yhat4 = predict(fit.rf, newdata = titanic.test, type = "response")
tuneRF(titanic.train[, c(1,4,5,6,7,9,11,15)], titanic.train$survived, mtryStart = 8, ntreeTry=50, stepF
```

```
## Warning in randomForest.default(x, y, mtry = mtryStart, ntree = ntreeTry, :
## The response has five or fewer unique values. Are you sure you want to do
## regression?
```

```
## mtry = 8  OOB error = 0.1528589
## Searching left ...
```

```
## Warning in randomForest.default(x, y, mtry = mtryCur, ntree = ntreeTry, :
## The response has five or fewer unique values. Are you sure you want to do
```

```
## regression?
## mtry = 3     OOB error = 0.1407674
## 0.07910251 0.05
## Warning in randomForest.default(x, y, mtry = mtryCur, ntree = ntreeTry, :
## The response has five or fewer unique values. Are you sure you want to do
## regression?
## mtry = 1     OOB error = 0.1477952
## -0.04992515 0.05
## Searching right ...
```




```
##   mtry  OOBError
## 1    1 0.1477952
## 3    3 0.1407674
## 8    8 0.1528589
```

```
fit.rf <- randomForest(survived ~ pclass + title + sex + age + sibsp + parch + fare + embarked, data = t
```

```
## Warning in randomForest.default(m, y, ...): The response has five or fewer
## unique values. Are you sure you want to do regression?
```

```
fit.rf
```

```
##
## Call:
##  randomForest(formula = survived ~ pclass + title + sex + age +      sibsp + parch + fare + embarked
##                Type of random forest: regression
##                      Number of trees: 500
```

```
## No. of variables tried at each split: 3
##
##           Mean of squared residuals: 0.1384393
##                     % Var explained: 42.57
```

```
yhat4 = predict(fit.rf, newdata = titanic.test, type = "response")
```

19. Compare the accuracy of each of the models from this problem set using ROC curves. Comment on which statistical learning method works best for predicting survival of the Titanic passengers.

```
#compare accuracies
obs <- titanic.test$survived
par(mfrow=c(2,2))
roc1 <- roc(obs, yhat)
```

```
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases
```

```
plot.roc(roc1)
roc2 <- roc(obs, yhat2)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

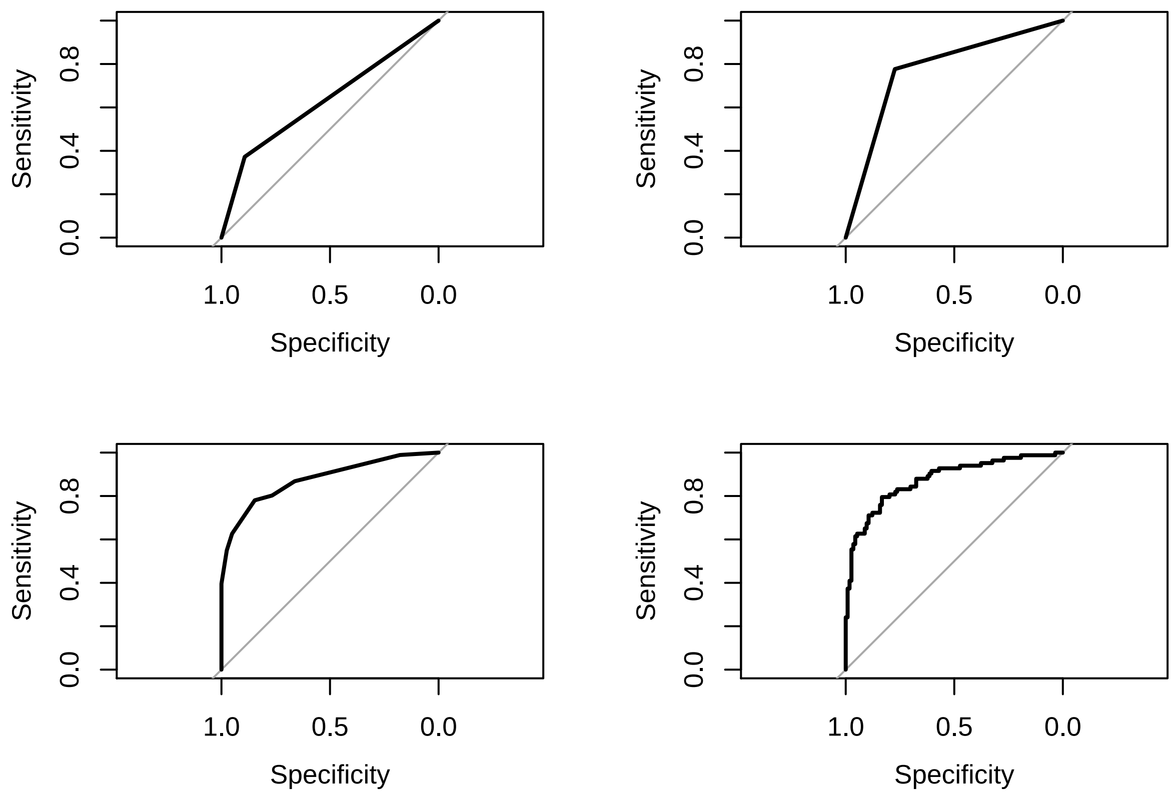
```
plot.roc(roc2)
yhat3 <- as.numeric(yhat3)
roc3 <- roc(obs, yhat3, na.rm = TRUE)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```
plot.roc(roc3)
yhat4 <- as.numeric(yhat4)
roc4 <- roc(obs, yhat4, na.rm = TRUE)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```
plot.roc(roc4)
```



#Random forest modeling method works best for predicting survival of the Titanic passengers based on the (both random forest models have the highest prediction accuracies) ROC curves.