Load Data Set

We are doing a 70-30 split

```
test = read.csv("test.csv")
train = read.csv("train.csv")
test_completer = read.csv("test_completer.csv")
train_completer = read.csv("train_completer.csv")
```

```
## 'data.frame':    779 obs. of  18 variables:
##  $ converter           : int  1 0 0 0 1 0 0 0 0 0 ...
##  $ PTGENDER            : chr  "Male" "Female" "Male" "Female" ...
##  $ AGE                 : num  81.1 68 71.2 63.5 82.2 71.6 72.1 75 83.7 77.1 ...
##  $ PTEDUCAT            : int  16 16 18 15 20 20 14 16 18 17 ...
##  $ PTMARRY             : chr  "Married" "Married" "Married" "Married" ...
##  $ DX.bl               : chr  "LMCI" "EMCI" "LMCI" "EMCI" ...
##  $ CDRSB               : num  4 0.5 1.5 0.5 1.5 2.5 1.5 1 1 0 ...
##  $ FAQ                 : int  0 0 9 0 4 8 1 0 0 0 ...
##  $ MMSE                : int  26 28 29 29 24 26 27 29 29 28 ...
##  $ ADAS11              : num  18 5 6 9 12.3 ...
##  $ ADAS13              : num  32 8 11 12 20.3 ...
##  $ ADASQ4              : int  9 2 4 3 5 6 4 5 5 5 ...
##  $ RAVLT.immediate     : int  18 46 32 45 29 28 44 37 30 33 ...
##  $ RAVLT.learning      : int  4 6 6 8 0 3 8 8 0 4 ...
##  $ RAVLT.forgetting    : int  6 6 6 4 5 6 7 2 7 4 ...
##  $ RAVLT.perc.forgetting: num  100 50 66.7 33.3 83.3 ...
##  $ LDELTOTAL           : int  4 10 5 8 2 2 9 20 6 8 ...
##  $ TRABSCOR            : int  217 60 79 54 155 66 159 53 44 180 ...

## 'data.frame':    779 obs. of  18 variables:
##  $ converter           : Factor w/ 2 levels "0","1": 2 1 1 1 2 1 1 1 1 1 ...
##  $ PTGENDER            : Factor w/ 2 levels "Female","Male": 2 1 2 1 2 2 2 1 1 2 ...
##  $ AGE                 : num  81.1 68 71.2 63.5 82.2 71.6 72.1 75 83.7 77.1 ...
##  $ PTEDUCAT            : int  16 16 18 15 20 20 14 16 18 17 ...
##  $ PTMARRY             : Factor w/ 4 levels "Divorced","Married",..: 2 2 2 2 2 2 2 2 1 2 ...
##  $ DX.bl               : Factor w/ 2 levels "EMCI","LMCI": 2 1 2 1 2 2 1 1 2 2 ...
##  $ CDRSB               : num  4 0.5 1.5 0.5 1.5 2.5 1.5 1 1 0 ...
##  $ FAQ                 : int  0 0 9 0 4 8 1 0 0 0 ...
##  $ MMSE                : int  26 28 29 29 24 26 27 29 29 28 ...
##  $ ADAS11              : num  18 5 6 9 12.3 ...
##  $ ADAS13              : num  32 8 11 12 20.3 ...
##  $ ADASQ4              : int  9 2 4 3 5 6 4 5 5 5 ...
##  $ RAVLT.immediate     : int  18 46 32 45 29 28 44 37 30 33 ...
##  $ RAVLT.learning      : int  4 6 6 8 0 3 8 8 0 4 ...
##  $ RAVLT.forgetting    : int  6 6 6 4 5 6 7 2 7 4 ...
##  $ RAVLT.perc.forgetting: num  100 50 66.7 33.3 83.3 ...
##  $ LDELTOTAL           : int  4 10 5 8 2 2 9 20 6 8 ...
##  $ TRABSCOR            : int  217 60 79 54 155 66 159 53 44 180 ...

## 'data.frame':    189 obs. of  18 variables:
##  $ converter           : Factor w/ 2 levels "0","1": 2 2 1 1 2 1 1 2 1 1 ...
##  $ PTGENDER            : Factor w/ 2 levels "Female","Male": 1 2 2 2 2 2 2 2 1 2 ...
##  $ AGE                 : num  72.8 78.8 73.5 74.8 74.8 72.4 57.8 85.6 81.4 79.8 ...
```

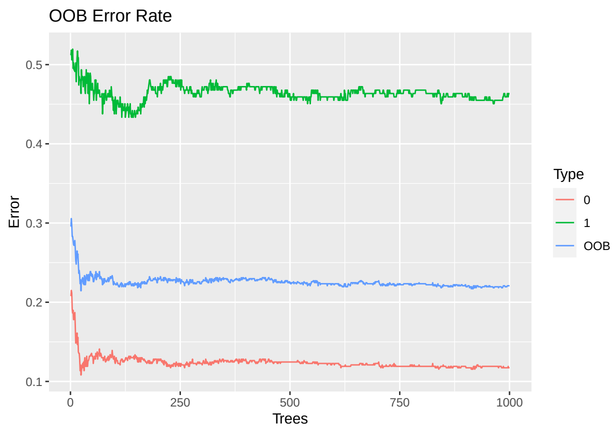
```
##  $ PTEDUCAT             : int  12 20 12 15 12 14 20 20 18 14 ...
##  $ PTMARRY              : Factor w/ 4 levels "Divorced","Married",..: 1 2 2 1 2 2 2 2 2 2 ...
##  $ DX.bl                : Factor w/ 2 levels "EMCI","LMCI": 2 2 2 1 2 2 2 2 1 2 ...
##  $ CDRSB                : num  2 1.5 1.5 0.5 2.5 2 1.5 4 0.5 1.5 ...
##  $ FAQ                  : int  1 3 0 0 6 0 1 19 0 0 ...
##  $ MMSE                 : int  28 29 27 29 25 29 28 27 29 29 ...
##  $ ADAS11               : num  7.33 15.67 8.67 6 12 ...
##  $ ADAS13               : num  10.3 22.7 17.7 9 23 ...
##  $ ADASQ4               : int  3 6 7 2 10 6 7 6 3 3 ...
##  $ RAVLT.immediate      : int  44 31 34 48 22 30 33 24 51 37 ...
##  $ RAVLT.learning       : int  9 3 4 7 4 5 5 1 10 3 ...
##  $ RAVLT.forgetting     : int  8 5 7 6 6 6 6 6 9 6 ...
##  $ RAVLT.perc.forgetting: num  61.5 71.4 77.8 54.5 100 ...
##  $ LDELTOTAL            : int  3 6 4 11 0 4 5 3 11 6 ...
##  $ TRABSCOR             : int  112 134 107 108 125 300 70 143 75 69 ...
```

Fit Random Forest

```
##
## Call:
##  randomForest(formula = converter ~ ., data = train, mtry = mtry.value,      importance = TRUE, prox
##                Type of random forest: classification
##                      Number of trees: 1000
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 22.08%
## Confusion matrix:
##     0   1 class.error
## 0 482  64   0.1172161
## 1 108 125   0.4635193
```

- Out of Bag Error rate here is 100% - 22.08% = 77.92%. This means that 77.92% of the OOB samples were correctly classified by the random forest. Just as a reminder, OOB is used to make predictions on the data. It is the "testing" data.

Optimality of Trees



- As we can observe from the graph above. The default number of trees in R is 500 and after 500 trees, the error seems to stabilize. We will decide to stick to 500 trees.

MTRY Value Search

```
## [1] 4
```

Table 1: OOB Error Rate for Different MTRY Values

MTRY	OOB Error
1	0.222
2	0.221
3	0.218
4	0.214
5	0.221
6	0.220
7	0.218
8	0.231
9	0.221
10	0.234

- Here we are finding different values of mtry which give the lowest OOB-error. We set values of **mtry** from 1 to 10 and noticed that an **mtry** of 1 has the lowest OOB-error.

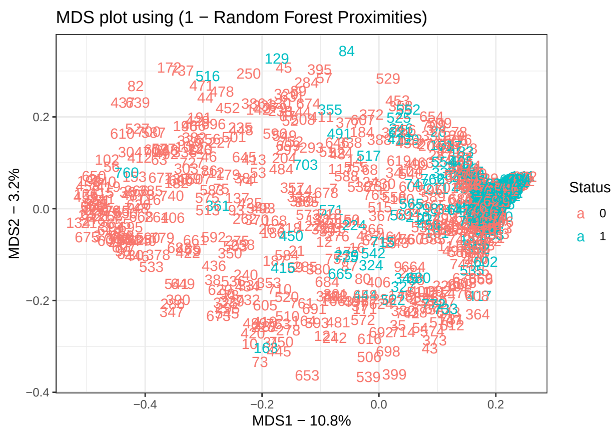
Final Random Forest Model

```
##
## Call:
##  randomForest(formula = converter ~ ., data = train, mtry = 4,      importance = TRUE, proximity = T
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 22.34%
## Confusion matrix:
##     0   1 class.error
## 0 478  68   0.1245421
## 1 106 127   0.4549356
```

	0	1	class.error
0	478	68	0.1245421
1	106	127	0.4549356

- The final model on the training set appears to be struggling with correctly classifying people who are not going to develop Alzheimer's Disease in the next 5 years. Basically, the type 2 error is at 53%.

MDS Plot



- Just as a reminder, MDS aims to represent the pairwise dissimilarities or distances between data points in a lower-dimensional space while preserving the original distances as much as possible.
- In the above MDS plot, we can see that there is a clear distinction between groups but there appears to be overlaps in some areas. There are converters in the non-converter groups and vice versa.

Predict on Test Data

```
## [1] 0.8
```

- The accuracy of our random forest model on the test set is 79%.

Sensitivity Analysis

MTRY Value Search

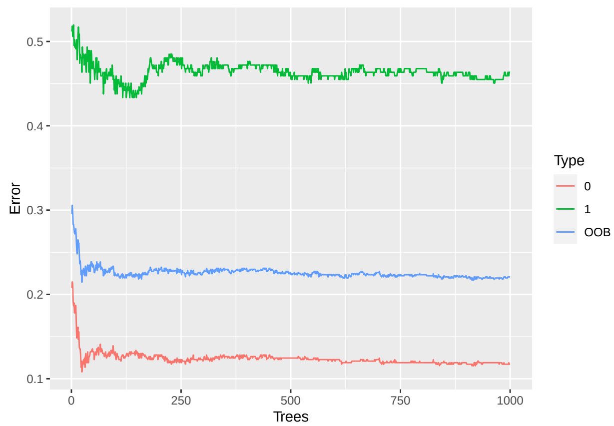
```
##  [1] 0.2116402 0.2116402 0.2222222 0.2116402 0.2275132 0.2222222 0.2222222
##  [8] 0.2063492 0.2169312 0.1957672
```

```
## [1] 0.1957672
```

```
## [1] 10
```

Random Forest Model

Optimality of Trees



Final Random Forest Model

Predict on Test Data

```
## [1] 0.7654321
```