Flights ontime ML models

January 24, 2018

1 Data Scource

he U.S. Department of Transportation's (DOT) Bureau of Transportation Statistics (BTS) tracks the on-time performance of domestic flights operated by large air carriers. Summary information about the number of on-time, delayed, canceled, and diverted flights appears in the DOT's monthly Air Travel Consumer Report. It's published about 30 days after the month's end, as well as in summary tables posted on this website.

TS began collecting details on the causes of flight delays in June 2003. Summary statistics and the raw data upon which they are based are made available to the public at the time the Air Travel Consumer Report is released.

The data can be viewed and downloaded using the BTS URL shown below. (https://www.transtats.bts.gov/ot_delay/ot_delaycause1.asp)

2 Data Preparation:

```
In [2]: # load dataset
        flightPerf <- read.csv('D:\\...\\FlightPerformance.csv',head=T)
        head(flightPerf)
        str(flightPerf)
        dim(flightPerf)
        summary(flightPerf)
```

schedtime	carrier	deptime	dest	distance	date	flightnumber	origin	weather	dayweek
1455	OH	1455	JFK	184	1/1/2004	5935	BWI	0	4
1640	DH	1640	JFK	213	1/1/2004	6155	DCA	0	4
1245	DH	1245	LGA	229	1/1/2004	7208	IAD	0	4
1715	DH	1709	LGA	229	1/1/2004	7215	IAD	0	4
1039	DH	1035	LGA	229	1/1/2004	7792	IAD	0	4
840	DH	839	JFK	228	1/1/2004	7800	IAD	0	4

```
'data.frame':        2201 obs. of  13 variables:
 $ schedtime   : int  1455 1640 1245 1715 1039 840 1240 1645 1715 2120 ...
 $ carrier     : Factor w/ 8 levels "CO","DH","DL",..: 5 2 2 2 2 2 2 2 2 2 ...
 $ deptime     : int  1455 1640 1245 1709 1035 839 1243 1644 1710 2129 ...
 $ dest        : Factor w/ 3 levels "EWR","JFK","LGA": 2 2 3 3 3 2 2 2 2 2 ...
 $ distance    : int  184 213 229 229 229 228 228 228 228 228 ...
 $ date        : Factor w/ 31 levels "1/1/2004","1/10/2004",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
$ flightnumber: int  5935 6155 7208 7215 7792 7800 7806 7810 7812 7814 ...
$ origin      : Factor w/ 3 levels "BWI","DCA","IAD": 1 2 3 3 3 3 3 3 3 3 ...
$ weather     : int  0 0 0 0 0 0 0 0 0 0 ...
$ dayweek     : int  4 4 4 4 4 4 4 4 4 4 ...
$ daymonth    : int  1 1 1 1 1 1 1 1 1 1 ...
$ tailnu      : Factor w/ 549 levels "N10323","N10575",..: 526 263 382 350 385 374 241 227 240
$ delay       : Factor w/ 2 levels "delayed","ontime": 2 2 2 2 2 2 2 2 2 2 ...
```

1. 2201 2. 13

```
  schedtime       carrier       deptime      dest         distance
Min.   : 600   DH     :551   Min.   :  10   EWR: 665   Min.   :169.0
1st Qu.:1000   RU     :408   1st Qu.:1004   JFK: 386   1st Qu.:213.0
Median :1455   US     :404   Median :1450   LGA:1150   Median :214.0
Mean   :1372   DL     :388   Mean   :1369              Mean   :211.9
3rd Qu.:1710   MQ     :295   3rd Qu.:1709              3rd Qu.:214.0
Max.   :2130   CO     : 94   Max.   :2330              Max.   :229.0
               (Other): 61
       date        flightnumber  origin        weather           dayweek
1/22/2004:  86   Min.   : 746   BWI: 145   Min.   :0.00000   Min.   :1.000
1/13/2004:  85   1st Qu.:2156   DCA:1370   1st Qu.:0.00000   1st Qu.:2.000
1/20/2004:  85   Median :2385   IAD: 686   Median :0.00000   Median :4.000
1/21/2004:  85   Mean   :3815              Mean   :0.01454   Mean   :3.905
1/6/2004 :  85   3rd Qu.:6155              3rd Qu.:0.00000   3rd Qu.:5.000
1/8/2004 :  85   Max.   :7924              Max.   :1.00000   Max.   :7.000
(Other)  :1690
   daymonth         tailnu         delay
Min.   : 1.00   N225DL :  65   delayed: 428
1st Qu.: 8.00   N242DL :  56   ontime :1773
Median :16.00   N223DZ :  50
Mean   :16.02   N221DL :  45
3rd Qu.:23.00   N241DL :  36
Max.   :31.00   N722UW :  36
                (Other):1913
```

In [3]:
```r
# choose schedtime, carrier, dest, date, origin, and weather as predictors
# delay as response vairable.
flightPerf.data <- flightPerf[,c(1,2,4,6,8,9,13)]
head(flightPerf.data)
str(flightPerf.data)
```

schedtime	carrier	dest	date	origin	weather	delay
1455	OH	JFK	1/1/2004	BWI	0	ontime
1640	DH	JFK	1/1/2004	DCA	0	ontime
1245	DH	LGA	1/1/2004	IAD	0	ontime
1715	DH	LGA	1/1/2004	IAD	0	ontime
1039	DH	LGA	1/1/2004	IAD	0	ontime
840	DH	JFK	1/1/2004	IAD	0	ontime

```
'data.frame':        2201 obs. of  7 variables:
 $ schedtime: int  1455 1640 1245 1715 1039 840 1240 1645 1715 2120 ...
 $ carrier  : Factor w/ 8 levels "CO","DH","DL",..: 5 2 2 2 2 2 2 2 2 2 ...
 $ dest     : Factor w/ 3 levels "EWR","JFK","LGA": 2 2 3 3 3 2 2 2 2 2 ...
 $ date     : Factor w/ 31 levels "1/1/2004","1/10/2004",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ origin   : Factor w/ 3 levels "BWI","DCA","IAD": 1 2 3 3 3 3 3 3 3 3 ...
 $ weather  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ delay    : Factor w/ 2 levels "delayed","ontime": 2 2 2 2 2 2 2 2 2 2 ...
```

2.1 Dummy Variables

```
In [4]: # change some categorical vairables to dummy variables
        flightPerf.data$date <- as.Date(flightPerf.data$date,'%m/%d/%Y')
        flightPerf.data$weather <- as.factor(flightPerf.data$weather)
        flightPerf.X <- model.matrix(delay~.,data=flightPerf.data,is.factor=TRUE)[,-1]
        delay <- flightPerf.data$delay
        flightPerf.data1 <- cbind.data.frame(flightPerf.X,delay)
        flightPerf.data1$date <- flightPerf.data1$date-12418
        head(flightPerf.data1)
        str(flightPerf.data1)
        unique(flightPerf.data1$date)
```

schedtime	carrierDH	carrierDL	carrierMQ	carrierOH	carrierRU	carrierUA	carrierUS	destJFK
1455	0	0	0	1	0	0	0	1
1640	1	0	0	0	0	0	0	1
1245	1	0	0	0	0	0	0	0
1715	1	0	0	0	0	0	0	0
1039	1	0	0	0	0	0	0	0
840	1	0	0	0	0	0	0	1

```
'data.frame':        2201 obs. of  15 variables:
 $ schedtime: num  1455 1640 1245 1715 1039 ...
 $ carrierDH: num  0 1 1 1 1 1 1 1 1 1 ...
 $ carrierDL: num  0 0 0 0 0 0 0 0 0 0 ...
 $ carrierMQ: num  0 0 0 0 0 0 0 0 0 0 ...
 $ carrierOH: num  1 0 0 0 0 0 0 0 0 0 ...
 $ carrierRU: num  0 0 0 0 0 0 0 0 0 0 ...
 $ carrierUA: num  0 0 0 0 0 0 0 0 0 0 ...
 $ carrierUS: num  0 0 0 0 0 0 0 0 0 0 ...
 $ destJFK  : num  1 1 0 0 0 1 1 1 1 1 ...
 $ destLGA  : num  0 0 1 1 1 0 0 0 0 0 ...
 $ date     : num  0 0 0 0 0 0 0 0 0 0 ...
 $ originDCA: num  0 1 0 0 0 0 0 0 0 0 ...
 $ originIAD: num  0 0 1 1 1 1 1 1 1 1 ...
 $ weather1 : num  0 0 0 0 0 0 0 0 0 0 ...
 $ delay    : Factor w/ 2 levels "delayed","ontime": 2 2 2 2 2 2 2 2 2 2 ...
```

1. 0 2. 1 3. 2 4. 3 5. 4 6. 5 7. 6 8. 7 9. 8 10. 9 11. 10 12. 11 13. 12 14. 13 15. 14 16. 15 17. 16 18. 17 19. 18 20. 19 21. 20 22. 21 23. 22 24. 23 25. 24 26. 25 27. 26 28. 27 29. 28 30. 29 31. 30

2.2 Split Data and Normalize

In [5]:
```
# split data to train(70%) and test(30%)
set.seed(99)
n=dim(flightPerf.data1)[1]
trainID <- sample(1:n,round(n*0.7))
train <- flightPerf.data1[trainID,]
test <- flightPerf.data1[-trainID,]
dim(train)
dim(test)
```

1. 1541 2. 15
1. 660 2. 15

In [6]:
```
# Normalize
normalize <- function(x){
  return ((x-min(x))/(max(x)-min(x)))
}

train$delay <- as.numeric(train$delay)
train <- as.data.frame(lapply(train,normalize))
summary(train)

test$delay <- as.numeric(test$delay)
test <- as.data.frame(lapply(test,normalize))
summary(test)
```

```
   schedtime        carrierDH        carrierDL        carrierMQ
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
 1st Qu.:0.2810   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
 Median :0.5425   Median :0.0000   Median :0.0000   Median :0.0000
 Mean   :0.5050   Mean   :0.2537   Mean   :0.1785   Mean   :0.1389
 3rd Qu.:0.7255   3rd Qu.:1.0000   3rd Qu.:0.0000   3rd Qu.:0.0000
 Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
   carrierOH         carrierRU        carrierUA         carrierUS
 Min.   :0.00000   Min.   :0.0000   Min.   :0.00000   Min.   :0.0000
 1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:0.0000
 Median :0.00000   Median :0.0000   Median :0.00000   Median :0.0000
 Mean   :0.01233   Mean   :0.1785   Mean   :0.01298   Mean   :0.1817
 3rd Qu.:0.00000   3rd Qu.:0.0000   3rd Qu.:0.00000   3rd Qu.:0.0000
 Max.   :1.00000   Max.   :1.0000   Max.   :1.00000   Max.   :1.0000
    destJFK          destLGA            date          originDCA
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
 1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.2333   1st Qu.:0.0000
 Median :0.0000   Median :1.0000   Median :0.5000   Median :1.0000
 Mean   :0.1739   Mean   :0.5295   Mean   :0.5005   Mean   :0.6256
```

```
 3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:0.7333   3rd Qu.:1.0000
 Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
   originIAD         weather1           delay
 Min.   :0.0000   Min.   :0.00000   Min.   :0.0000
 1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:1.0000
 Median :0.0000   Median :0.00000   Median :1.0000
 Mean   :0.3089   Mean   :0.01428   Mean   :0.8008
 3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:1.0000
 Max.   :1.0000   Max.   :1.00000   Max.   :1.0000


   schedtime        carrierDH        carrierDL        carrierMQ
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
 1st Qu.:0.2157   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
 Median :0.5588   Median :0.0000   Median :0.0000   Median :0.0000
 Mean   :0.5034   Mean   :0.2424   Mean   :0.1712   Mean   :0.1227
 3rd Qu.:0.7263   3rd Qu.:0.0000   3rd Qu.:0.0000   3rd Qu.:0.0000
 Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
   carrierOH         carrierRU        carrierUA         carrierUS
 Min.   :0.00000   Min.   :0.0000   Min.   :0.00000   Min.   :0.0000
 1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:0.0000
 Median :0.00000   Median :0.0000   Median :0.00000   Median :0.0000
 Mean   :0.01667   Mean   :0.2015   Mean   :0.01667   Mean   :0.1879
 3rd Qu.:0.00000   3rd Qu.:0.0000   3rd Qu.:0.00000   3rd Qu.:0.0000
 Max.   :1.00000   Max.   :1.0000   Max.   :1.00000   Max.   :1.0000
    destJFK           destLGA            date           originDCA
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
 1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.2667   1st Qu.:0.0000
 Median :0.0000   Median :1.0000   Median :0.4833   Median :1.0000
 Mean   :0.1788   Mean   :0.5061   Mean   :0.5017   Mean   :0.6152
 3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:0.7333   3rd Qu.:1.0000
 Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
   originIAD         weather1           delay
 Min.   :0.0000   Min.   :0.00000   Min.   :0.0000
 1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:1.0000
 Median :0.0000   Median :0.00000   Median :1.0000
 Mean   :0.3182   Mean   :0.01515   Mean   :0.8167
 3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:1.0000
 Max.   :1.0000   Max.   :1.00000   Max.   :1.0000
```

3 Build models:

In [8]:
```
library(class)
library(caret)
library(e1071)
library(pROC)
```

```
library(MASS)
library(neuralnet)
library(NeuralNetTools)
library(glmnet)
library(leaps)
```

```
In [ ]: #install.packages("car",repos = 'http://cran.rstudio.com/','C:\\Anaconda3\\R\\library'
        #install.packages("caret",repos = 'http://cran.rstudio.com/','C:\\Anaconda3\\R\\library
```

```
In [10]: # create formula
         (allVars <- colnames(train))
         (predictVars <- allVars[!allVars%in%"delay"])
         (predictVars <- paste(predictVars, collapse=" + "))
         (formula <- as.formula(paste("delay ~", predictVars, collapse=" +")))
```

1. 'schedtime' 2. 'carrierDH' 3. 'carrierDL' 4. 'carrierMQ' 5. 'carrierOH' 6. 'carrierRU' 7. 'carrierUA' 8. 'carrierUS' 9. 'destJFK' 10. 'destLGA' 11. 'date' 12. 'originDCA' 13. 'originIAD' 14. 'weather1' 15. 'delay'

1. 'schedtime' 2. 'carrierDH' 3. 'carrierDL' 4. 'carrierMQ' 5. 'carrierOH' 6. 'carrierRU' 7. 'carrierUA' 8. 'carrierUS' 9. 'destJFK' 10. 'destLGA' 11. 'date' 12. 'originDCA' 13. 'originIAD' 14. 'weather1'

'schedtime + carrierDH + carrierDL + carrierMQ + carrierOH + carrierRU + carrierUA + carrierUS + destJFK + destLGA + date + originDCA + originIAD + weather1'

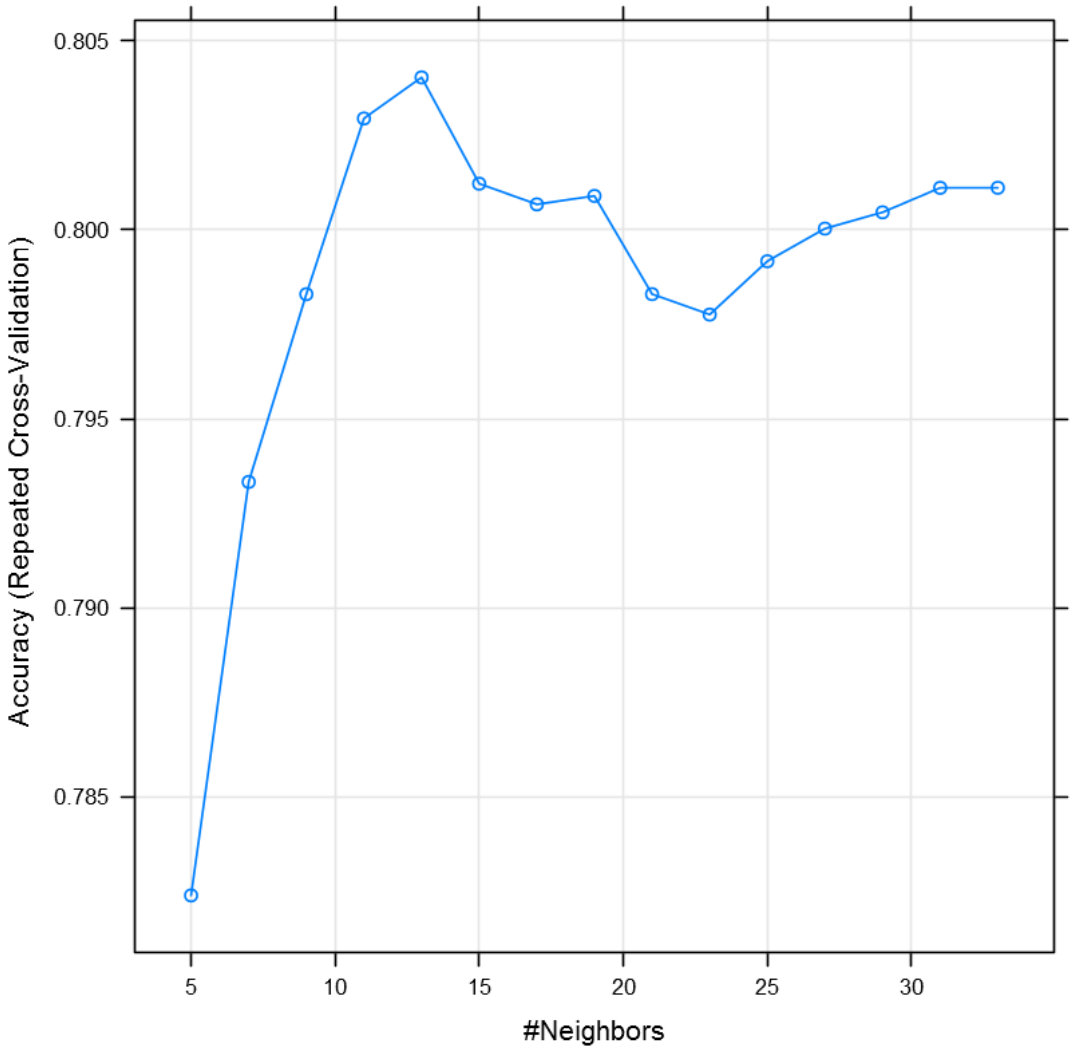
```
delay ~ schedtime + carrierDH + carrierDL + carrierMQ + carrierOH +
    carrierRU + carrierUA + carrierUS + destJFK + destLGA + date +
    originDCA + originIAD + weather1
```

3.1 K Nearest Neighbor

```
In [25]: ctrl <- trainControl(method="repeatedcv",repeats=6)
         knn.fit <- train(as.factor(delay)~., data=train,method="knn",preProcess='center',trCo
         knn.fit  # the highest train accuracy is k=13  0.8040267453  0.136064940338
         plot(knn.fit)
```

```
k-Nearest Neighbors

1541 samples
  14 predictor
   2 classes: '0', '1'

Pre-processing: centered (14)
Resampling: Cross-Validated (10 fold, repeated 6 times)
Summary of sample sizes: 1387, 1387, 1388, 1387, 1388, 1386, ...
Resampling results across tuning parameters:

  k   Accuracy      Kappa
   5  0.7823989694  0.154833472793
```

```
  7  0.7933312295  0.149889716519
  9  0.7982949309  0.138254478657
 11  0.8029402866  0.144799466861
 13  0.8040267453  0.136064940338
 15  0.8012163836  0.111633905121
 17  0.8006695810  0.086293591104
 19  0.8008930773  0.070144692025
 21  0.7982962999  0.038404409951
 23  0.7977559000  0.015015501146
 25  0.7991656650  0.010275877217
 27  0.8000286729  0.006874304938
 29  0.8004594787  0.005782617877
 31  0.8011067346  0.004428307655
 33  0.8011081402  0.002561983168

Accuracy was used to select the optimal model using  the largest value.
The final value used for the model was k = 13.
```

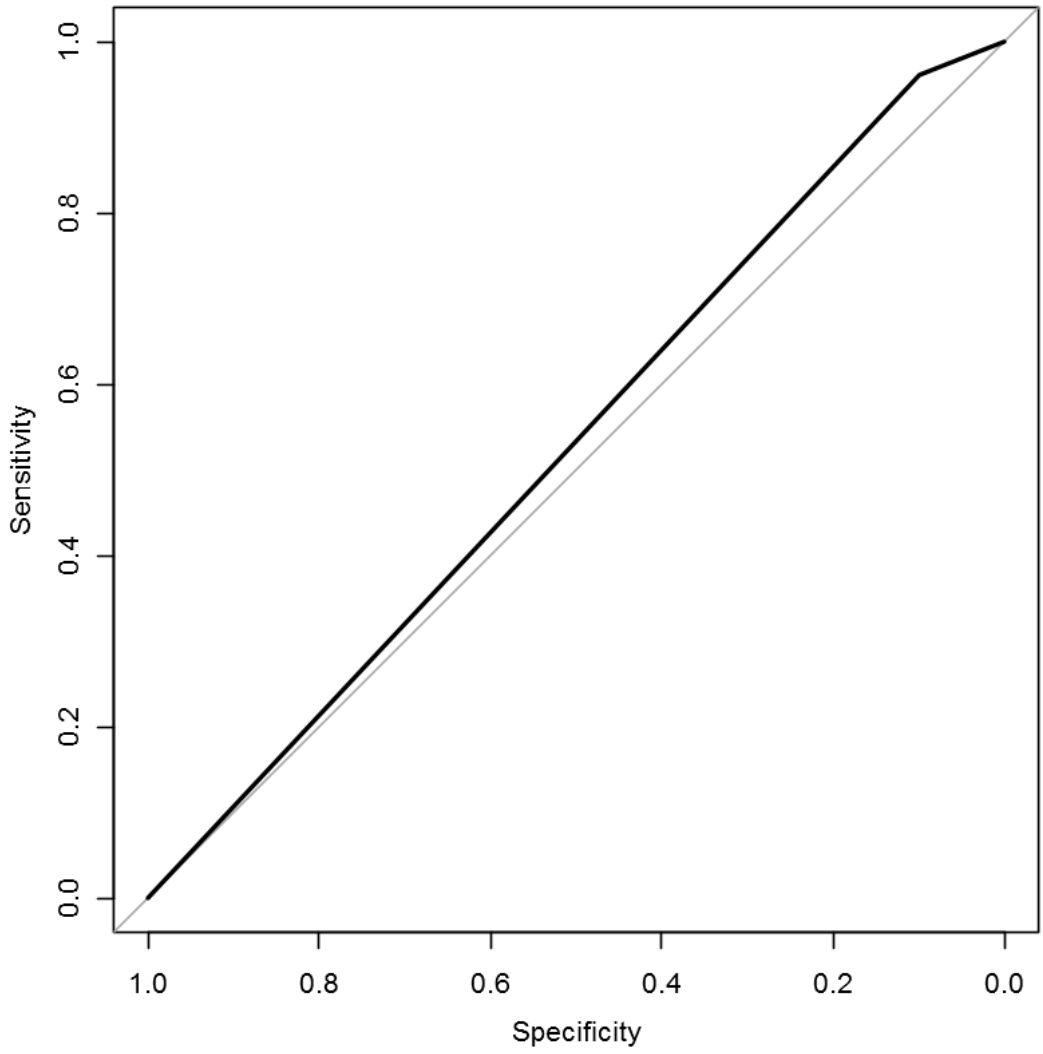



```
In [26]: knn.pred <- knn(train[,-15],test[,-15],train$delay,k=13)
         (knn.confTable <-table(knn.pred, test$delay))
         (knn.acc <- sum(diag(knn.confTable))/sum(knn.confTable))  #0.803030303030303
         roc(test$delay,as.numeric(knn.pred),plot=TRUE)  #0.5301063
```

```
knn.pred   0   1
       0  12  21
       1 109 518
```

0.803030303030303

```
Call:
roc.default(response = test$delay, predictor = as.numeric(knn.pred),     plot = TRUE)

Data: as.numeric(knn.pred) in 121 controls (test$delay 0) < 539 cases (test$delay 1).
Area under the curve: 0.5301063
```



3.2 Logistic Regression

3.2.1 StepAIC

```
In [28]: logit.fit <- glm(as.factor(delay)~ ., data=train, family=binomial)
         summary(logit.fit)  #AIC: 1424.7004
```

```
logit.best <- stepAIC(logit.fit, direction="both", trace=TRUE)
summary(logit.best)  #AIC: 1412.4512
```

```
Call:
glm(formula = as.factor(delay) ~ ., family = binomial, data = train)

Deviance Residuals:
       Min          1Q      Median          3Q         Max
-2.4939892   0.3585190   0.5563294   0.7116785   0.9726567

Coefficients:
              Estimate  Std. Error  z value   Pr(>|z|)
(Intercept)   1.0572284   0.4743264  2.22890 0.02582025 *
schedtime    -0.9407772   0.2516697 -3.73814 0.00018538 ***
carrierDH     0.2294532   0.4756323  0.48242 0.62950956
carrierDL     0.7355753   0.4386729  1.67682 0.09357781 .
carrierMQ    -0.2799086   0.4196008 -0.66708 0.50471903
carrierOH     1.0015239   0.8242950  1.21501 0.22436356
carrierRU     0.5055895   0.3645356  1.38694 0.16545969
carrierUA     0.3285310   0.8242043  0.39860 0.69018514
carrierUS     1.3225327   0.4635193  2.85324 0.00432757 **
destJFK       0.2707572   0.2782802  0.97297 0.33057022
destLGA       0.1818752   0.2926009  0.62158 0.53421732
date         -0.2099309   0.2330142 -0.90094 0.36762226
originDCA     0.5491657   0.3479535  1.57827 0.11450283
originIAD     0.3606519   0.3591655  1.00414 0.31531188
weather1    -17.8519548 494.7694467 -0.03608 0.97121748
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1538.9073  on 1540  degrees of freedom
Residual deviance: 1394.7004  on 1526  degrees of freedom
AIC: 1424.7004

Number of Fisher Scoring iterations: 15



Start:  AIC=1424.7
as.factor(delay) ~ schedtime + carrierDH + carrierDL + carrierMQ +
    carrierOH + carrierRU + carrierUA + carrierUS + destJFK +
    destLGA + date + originDCA + originIAD + weather1

            Df  Deviance       AIC
- carrierUA  1 1394.8631 1422.8631
```

```
- carrierDH  1 1394.9328 1422.9328
- destLGA    1 1395.0849 1423.0849
- carrierMQ  1 1395.1474 1423.1474
- date       1 1395.5121 1423.5121
- destJFK    1 1395.6372 1423.6372
- originIAD  1 1395.7052 1423.7052
- carrierOH  1 1396.2781 1424.2781
- carrierRU  1 1396.5999 1424.5999
<none>         1394.7004 1424.7004
- originDCA  1 1397.1743 1425.1743
- carrierDL  1 1397.4807 1425.4807
- carrierUS  1 1402.7639 1430.7639
- schedtime  1 1409.0363 1437.0363
- weather1   1 1455.8601 1483.8601

Step:  AIC=1422.86
as.factor(delay) ~ schedtime + carrierDH + carrierDL + carrierMQ +
    carrierOH + carrierRU + carrierUS + destJFK + destLGA + date +
    originDCA + originIAD + weather1

            Df  Deviance       AIC
- carrierDH  1 1394.9534 1420.9534
- destLGA    1 1395.5674 1421.5674
- date       1 1395.6708 1421.6708
- carrierMQ  1 1395.8429 1421.8429
- originIAD  1 1396.0835 1422.0835
- destJFK    1 1396.1912 1422.1912
- carrierOH  1 1396.2785 1422.2785
- carrierRU  1 1396.6412 1422.6412
<none>         1394.8631 1422.8631
- originDCA  1 1397.2037 1423.2037
- carrierDL  1 1397.6571 1423.6571
+ carrierUA  1 1394.7004 1424.7004
- carrierUS  1 1403.6890 1429.6890
- schedtime  1 1409.9375 1435.9375
- weather1   1 1455.8783 1481.8783

Step:  AIC=1420.95
as.factor(delay) ~ schedtime + carrierDL + carrierMQ + carrierOH +
    carrierRU + carrierUS + destJFK + destLGA + date + originDCA +
    originIAD + weather1

            Df  Deviance       AIC
- date       1 1395.7593 1419.7593
- destLGA    1 1395.7741 1419.7741
- carrierOH  1 1396.3149 1420.3149
- originIAD  1 1396.4307 1420.4307
- destJFK    1 1396.6767 1420.6767
```

```
- carrierMQ  1 1396.7160 1420.7160
<none>         1394.9534 1420.9534
- carrierRU  1 1397.1743 1421.1743
- originDCA  1 1397.2043 1421.2043
- carrierDL  1 1398.0005 1422.0005
+ carrierDH  1 1394.8631 1422.8631
+ carrierUA  1 1394.9328 1422.9328
- carrierUS  1 1405.1973 1429.1973
- schedtime  1 1409.9377 1433.9377
- weather1   1 1456.0920 1480.0920

Step:  AIC=1419.76
as.factor(delay) ~ schedtime + carrierDL + carrierMQ + carrierOH +
    carrierRU + carrierUS + destJFK + destLGA + originDCA + originIAD +
    weather1

            Df  Deviance       AIC
- destLGA    1 1396.6236 1418.6236
- carrierOH  1 1397.0241 1419.0241
- originIAD  1 1397.2106 1419.2106
- destJFK    1 1397.4970 1419.4970
- carrierMQ  1 1397.5077 1419.5077
<none>         1395.7593 1419.7593
- originDCA  1 1397.9625 1419.9625
- carrierRU  1 1398.0351 1420.0351
- carrierDL  1 1398.7241 1420.7241
+ date       1 1394.9534 1420.9534
+ carrierDH  1 1395.6708 1421.6708
+ carrierUA  1 1395.7393 1421.7393
- carrierUS  1 1405.8728 1427.8728
- schedtime  1 1410.7408 1432.7408
- weather1   1 1460.8275 1482.8275

Step:  AIC=1418.62
as.factor(delay) ~ schedtime + carrierDL + carrierMQ + carrierOH +
    carrierRU + carrierUS + destJFK + originDCA + originIAD +
    weather1

            Df  Deviance       AIC
- destJFK    1 1397.5002 1417.5002
- carrierMQ  1 1397.5713 1417.5713
- carrierOH  1 1398.1243 1418.1243
- carrierRU  1 1398.1576 1418.1576
- originIAD  1 1398.3619 1418.3619
- originDCA  1 1398.5973 1418.5973
<none>         1396.6236 1418.6236
+ destLGA    1 1395.7593 1419.7593
+ date       1 1395.7741 1419.7741
```

```
+ carrierDH  1 1396.4155 1420.4155
+ carrierUA  1 1396.5104 1420.5104
- carrierDL  1 1403.9058 1423.9058
- schedtime  1 1411.6425 1431.6425
- carrierUS  1 1415.7975 1435.7975
- weather1   1 1461.5628 1481.5628

Step:  AIC=1417.5
as.factor(delay) ~ schedtime + carrierDL + carrierMQ + carrierOH +
    carrierRU + carrierUS + originDCA + originIAD + weather1

            Df  Deviance       AIC
- carrierMQ  1 1398.4060 1416.4060
- carrierRU  1 1398.5799 1416.5799
- originIAD  1 1399.3034 1417.3034
- originDCA  1 1399.3907 1417.3907
<none>         1397.5002 1417.5002
- carrierOH  1 1399.5521 1417.5521
+ destJFK    1 1396.6236 1418.6236
+ date       1 1396.6776 1418.6776
+ carrierDH  1 1397.0216 1419.0216
+ carrierUA  1 1397.4421 1419.4421
+ destLGA    1 1397.4970 1419.4970
- carrierDL  1 1404.2639 1422.2639
- schedtime  1 1411.8956 1429.8956
- carrierUS  1 1415.8457 1433.8457
- weather1   1 1462.3010 1480.3010

Step:  AIC=1416.41
as.factor(delay) ~ schedtime + carrierDL + carrierOH + carrierRU +
    carrierUS + originDCA + originIAD + weather1

            Df  Deviance       AIC
- originDCA  1 1399.8806 1415.8806
<none>         1398.4060 1416.4060
- carrierOH  1 1400.9900 1416.9900
- originIAD  1 1401.0643 1417.0643
- carrierRU  1 1401.2092 1417.2092
+ carrierDH  1 1397.2606 1417.2606
+ carrierMQ  1 1397.5002 1417.5002
+ destJFK    1 1397.5713 1417.5713
+ date       1 1397.6233 1417.6233
+ destLGA    1 1398.2897 1418.2897
+ carrierUA  1 1398.3302 1418.3302
- schedtime  1 1412.4870 1428.4870
- carrierDL  1 1415.9078 1431.9078
- carrierUS  1 1437.1133 1453.1133
- weather1   1 1463.7527 1479.7527
```

```
Step:  AIC=1415.88
as.factor(delay) ~ schedtime + carrierDL + carrierOH + carrierRU +
    carrierUS + originIAD + weather1

            Df  Deviance       AIC
- carrierOH  1 1401.2352 1415.2352
- originIAD  1 1401.2954 1415.2954
- carrierRU  1 1401.4643 1415.4643
<none>         1399.8806 1415.8806
+ originDCA  1 1398.4060 1416.4060
+ destJFK    1 1399.1088 1417.1088
+ date       1 1399.1330 1417.1330
+ carrierMQ  1 1399.3907 1417.3907
+ carrierDH  1 1399.4044 1417.4044
+ destLGA    1 1399.7935 1417.7935
+ carrierUA  1 1399.8148 1417.8148
- schedtime  1 1413.5342 1427.5342
- carrierDL  1 1418.5592 1432.5592
- carrierUS  1 1440.4440 1454.4440
- weather1   1 1465.3261 1479.3261

Step:  AIC=1415.24
as.factor(delay) ~ schedtime + carrierDL + carrierRU + carrierUS +
    originIAD + weather1

            Df  Deviance       AIC
- originIAD  1 1402.2738 1414.2738
- carrierRU  1 1402.5091 1414.5091
<none>         1401.2352 1415.2352
+ carrierOH  1 1399.8806 1415.8806
+ destJFK    1 1399.9739 1415.9739
+ carrierMQ  1 1400.2554 1416.2554
+ date       1 1400.5945 1416.5945
+ carrierDH  1 1400.8997 1416.8997
+ originDCA  1 1400.9900 1416.9900
+ destLGA    1 1401.0174 1417.0174
+ carrierUA  1 1401.1717 1417.1717
- schedtime  1 1414.7420 1426.7420
- carrierDL  1 1418.8749 1430.8749
- carrierUS  1 1440.5397 1452.5397
- weather1   1 1467.1958 1479.1958

Step:  AIC=1414.27
as.factor(delay) ~ schedtime + carrierDL + carrierRU + carrierUS +
    weather1

            Df  Deviance       AIC
```

```
- carrierRU  1 1403.1354 1413.1354
<none>         1402.2738 1414.2738
+ carrierMQ  1 1400.2805 1414.2805
+ destJFK    1 1400.8746 1414.8746
+ carrierDH  1 1400.9283 1414.9283
+ originIAD  1 1401.2352 1415.2352
+ carrierOH  1 1401.2954 1415.2954
+ date       1 1401.6250 1415.6250
+ originDCA  1 1401.9211 1415.9211
+ destLGA    1 1401.9279 1415.9279
+ carrierUA  1 1402.1285 1416.1285
- schedtime  1 1415.4813 1425.4813
- carrierDL  1 1419.6748 1429.6748
- carrierUS  1 1443.6386 1453.6386
- weather1   1 1468.6882 1478.6882

Step:  AIC=1413.14
as.factor(delay) ~ schedtime + carrierDL + carrierUS + weather1

            Df  Deviance       AIC
+ carrierMQ  1 1400.4513 1412.4513
<none>         1403.1354 1413.1354
+ carrierRU  1 1402.2738 1414.2738
+ carrierOH  1 1402.3110 1414.3110
+ destLGA    1 1402.3313 1414.3313
+ date       1 1402.4581 1414.4581
+ originIAD  1 1402.5091 1414.5091
+ destJFK    1 1402.5110 1414.5110
+ originDCA  1 1402.7689 1414.7689
+ carrierDH  1 1402.8425 1414.8425
+ carrierUA  1 1403.0463 1415.0463
- schedtime  1 1416.7496 1424.7496
- carrierDL  1 1419.7043 1427.7043
- carrierUS  1 1444.1561 1452.1561
- weather1   1 1470.4197 1478.4197

Step:  AIC=1412.45
as.factor(delay) ~ schedtime + carrierDL + carrierUS + weather1 +
    carrierMQ

            Df  Deviance       AIC
<none>         1400.4513 1412.4513
- carrierMQ  1 1403.1354 1413.1354
+ destJFK    1 1399.7006 1413.7006
+ date       1 1399.7289 1413.7289
+ carrierOH  1 1399.8248 1413.8248
+ originDCA  1 1400.1504 1414.1504
+ carrierRU  1 1400.2805 1414.2805
```

```
+ carrierDH  1 1400.4166 1414.4166
+ carrierUA  1 1400.4201 1414.4201
+ destLGA    1 1400.4373 1414.4373
+ originIAD  1 1400.4484 1414.4484
- carrierDL  1 1413.3166 1423.3166
- schedtime  1 1414.5044 1424.5044
- carrierUS  1 1435.0940 1445.0940
- weather1   1 1466.0088 1476.0088



Call:
glm(formula = as.factor(delay) ~ schedtime + carrierDL + carrierUS +
    weather1 + carrierMQ, family = binomial, data = train)

Deviance Residuals:
       Min          1Q      Median          3Q         Max
-2.4564217   0.3629235   0.5799034   0.7234797   0.9186127

Coefficients:
               Estimate  Std. Error  z value               Pr(>|z|)
(Intercept)   1.7155353   0.1612931 10.63614 < 0.000000000000000222 ***
schedtime    -0.9103598   0.2457723 -3.70408             0.00021216 ***
carrierDL     0.6879229   0.2014440  3.41496             0.00063792 ***
carrierUS     1.2691326   0.2435912  5.21009          0.00000018875 ***
weather1    -16.9259270 300.9432363 -0.05624             0.95514829
carrierMQ    -0.2974672   0.1794500 -1.65766             0.09738592 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1538.9073  on 1540  degrees of freedom
Residual deviance: 1400.4512  on 1535  degrees of freedom
AIC: 1412.4512

Number of Fisher Scoring iterations: 14



In [133]: summary(logit.best)


Call:
glm(formula = as.factor(delay) ~ schedtime + carrierDL + carrierUS +
    weather1 + carrierMQ, family = binomial, data = train)

Deviance Residuals:
```

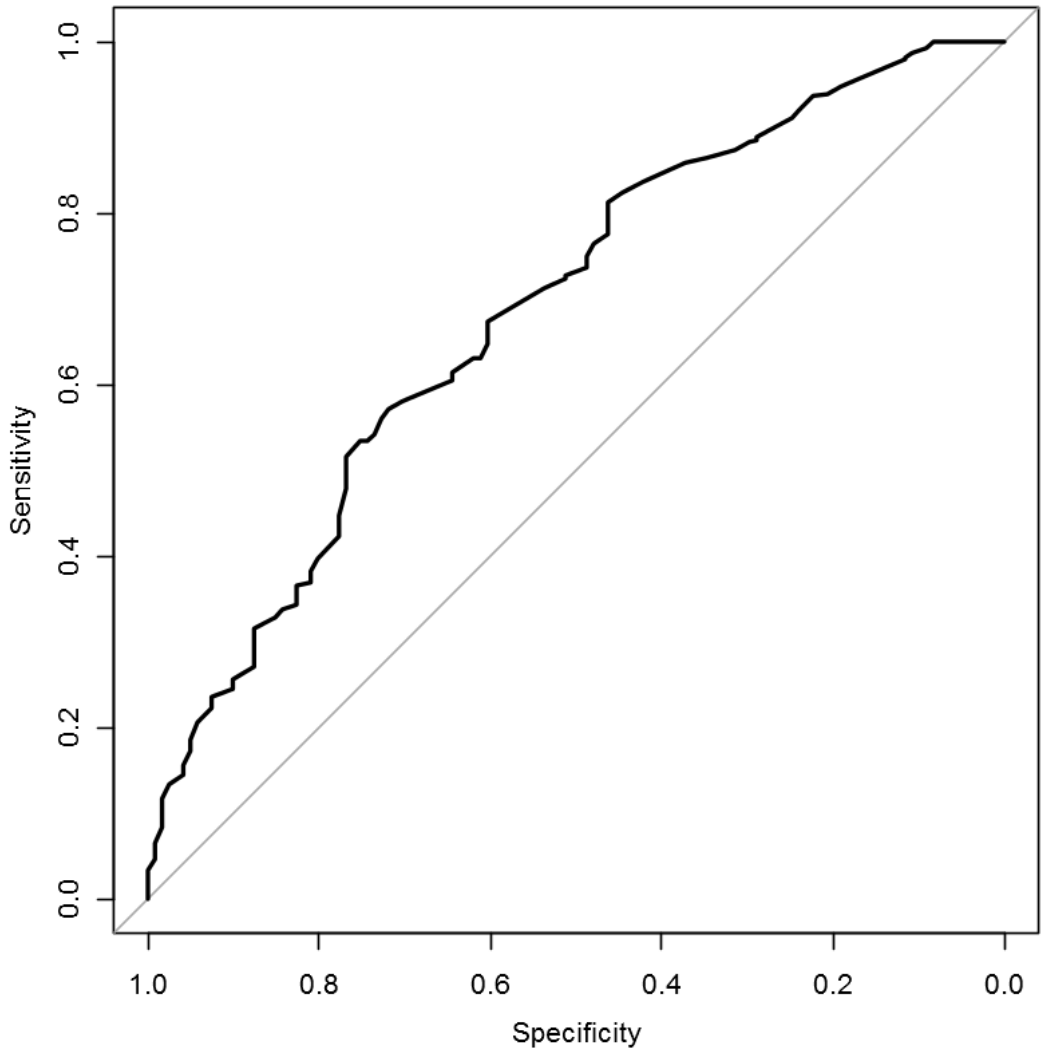
```
       Min          1Q      Median          3Q         Max
-2.4564217   0.3629235   0.5799034   0.7234797   0.9186127

Coefficients:
               Estimate  Std. Error  z value               Pr(>|z|)
(Intercept)   1.7155353   0.1612931 10.63614 < 0.000000000000000222 ***
schedtime    -0.9103598   0.2457723 -3.70408             0.00021216 ***
carrierDL     0.6879229   0.2014440  3.41496             0.00063792 ***
carrierUS     1.2691326   0.2435912  5.21009          0.00000018875 ***
weather1    -16.9259270 300.9432363 -0.05624             0.95514829
carrierMQ    -0.2974672   0.1794500 -1.65766             0.09738592 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1538.9073  on 1540  degrees of freedom
Residual deviance: 1400.4512  on 1535  degrees of freedom
AIC: 1412.4512

Number of Fisher Scoring iterations: 14
```

```
In [38]: logit.pred <- predict(logit.best, test, type='response')
         (logit.roc <- roc(test$delay,logit.pred,plot=TRUE)) #0.6865177
         logit.pred <- ifelse(logit.pred > 0.5, 1, 0)
         (logit.confTable <- table(logit.pred, test$delay))
         (logit.acc <- sum(diag(logit.confTable))/sum(logit.confTable)) #0.831818181818182
```

```
Call:
roc.default(response = test$delay, predictor = logit.pred, plot = TRUE)

Data: logit.pred in 121 controls (test$delay 0) < 539 cases (test$delay 1).
Area under the curve: 0.6865177
```

```
logit.pred   0   1
         0  10   0
         1 111 539
```

0.831818181818182



3.2.2 Lasso regression

```
In [40]: # Lasso regression
         train.xmat <- model.matrix(delay~., data=train)[,-1]
         test.xmat <- model.matrix(delay~., data=test)[,-1]

         lasso.fit <- cv.glmnet(train.xmat, train$delay, alpha=1,family="binomial")
         (lambda <- lasso.fit$lambda.min)  # optimal lambda 0.006489747543855525
         lasso.pred <- predict(lasso.fit, s=lambda, newx=test.xmat)
         lasso.pred <- as.numeric(lasso.pred)
         (lasso.roc <- roc(test$delay,lasso.pred,plot=TRUE)) #0.6891627
         lasso.pred <- ifelse(lasso.pred > 0.5, 1, 0)
```

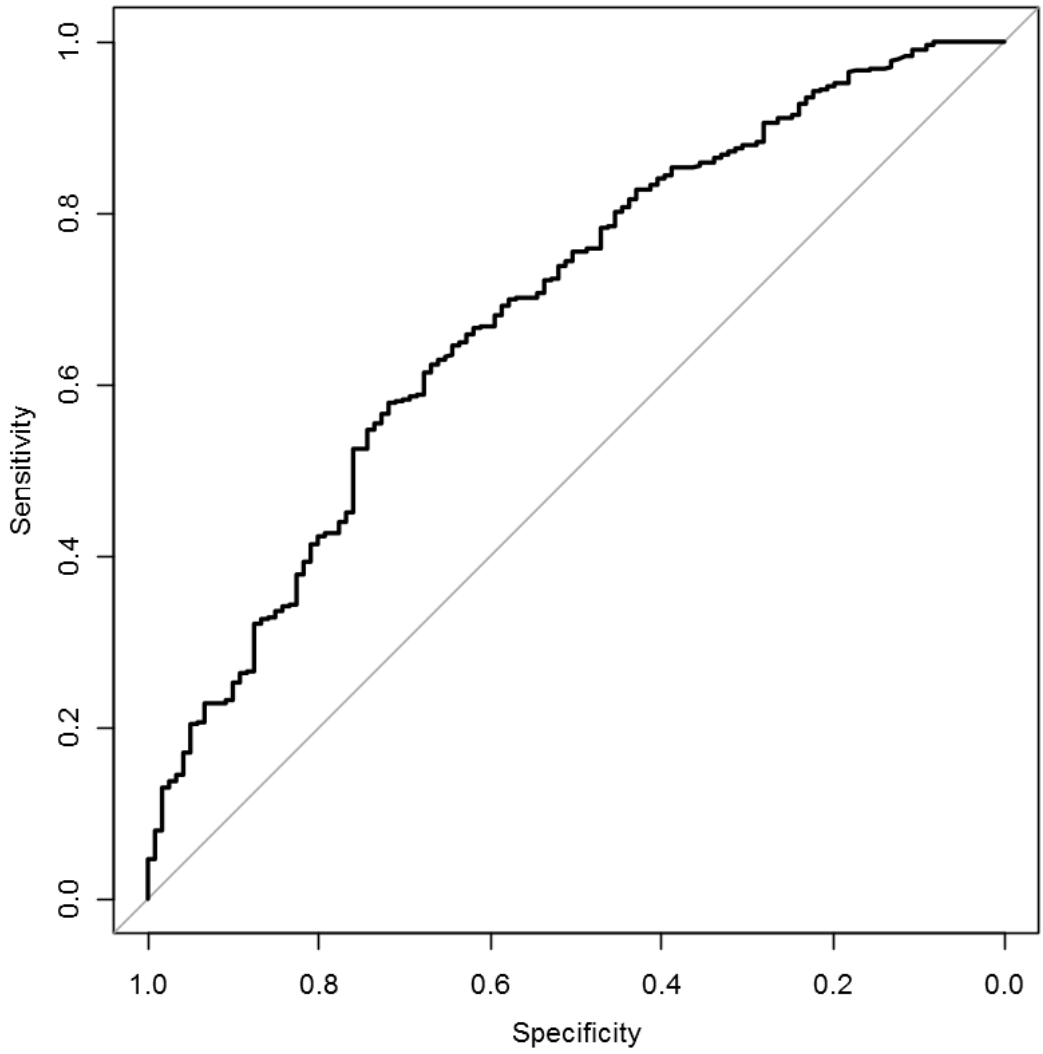
```
(lasso.confTable <- table(lasso.pred, test$delay))
(lasso.acc <- sum(diag(lasso.confTable))/sum(lasso.confTable)) #0.831818181818182
```

0.00648974754385525

```
Call:
roc.default(response = test$delay, predictor = lasso.pred, plot = TRUE)

Data: lasso.pred in 121 controls (test$delay 0) < 539 cases (test$delay 1).
Area under the curve: 0.6891627
```

```
lasso.pred   0   1
         0  10   0
         1 111 539
```

0.831818181818182



3.2.3 Ridge regression

```
In [41]: # create matrix
         train.xmat <- model.matrix(delay~., data=train)[,-1]
         test.xmat <- model.matrix(delay~., data=test)[,-1]

         ridge.fit <- cv.glmnet(train.xmat, train$delay, alpha=0,family="binomial")
         (lambda <- ridge.fit$lambda.min)  # optimal lambda 0.022262879625133
         ridge.pred <- predict(ridge.fit, s=lambda, newx=test.xmat)
         ridge.pred <- as.numeric(ridge.pred)
         (ridge.roc <- roc(test$delay,ridge.pred,plot=TRUE)) #0.6965762

         ridge.pred <- ifelse(ridge.pred > 0.5, 1, 0)
```

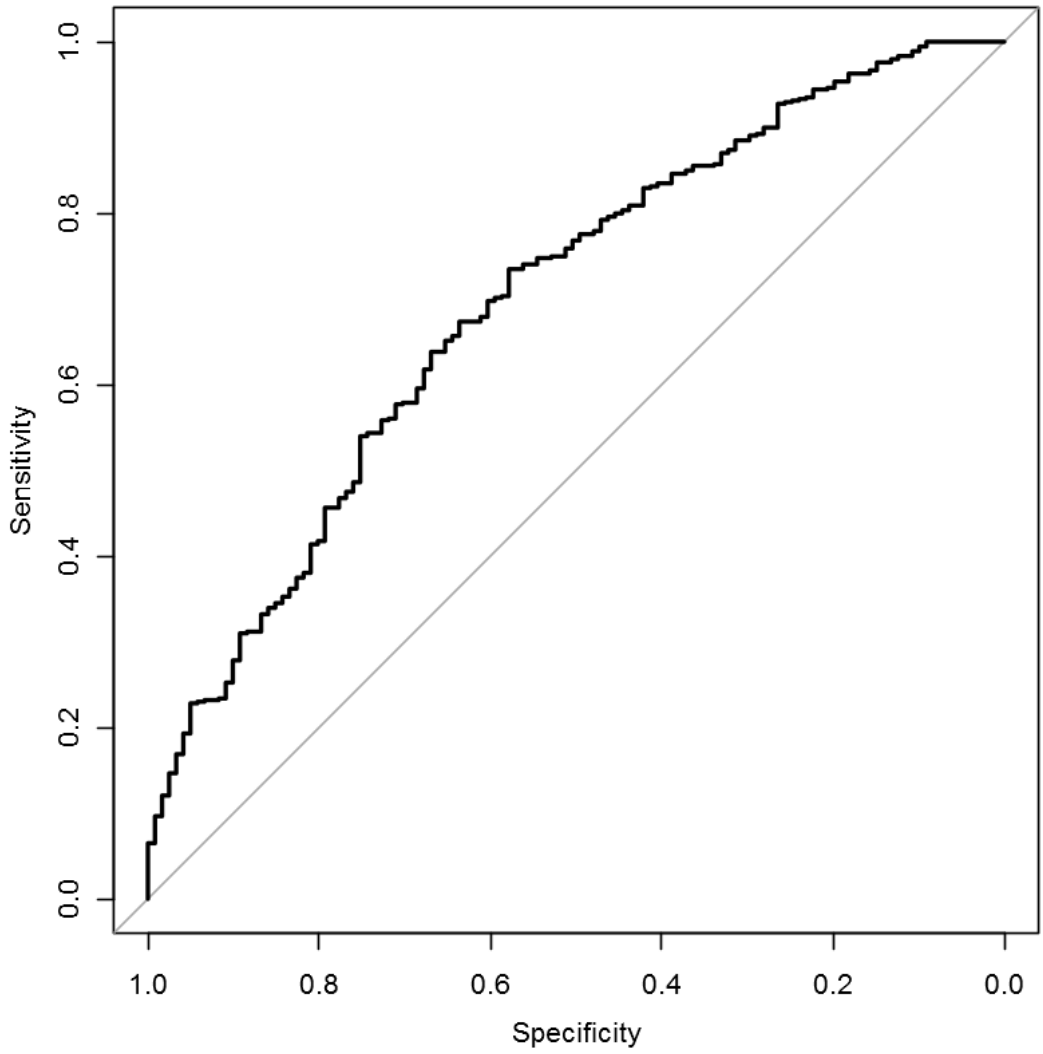
```
(ridge.confTable <- table(ridge.pred, test$delay))
(ridge.acc <- sum(diag(ridge.confTable))/sum(ridge.confTable)) #0.831818181818182
```

0.022262879625133

```
Call:
roc.default(response = test$delay, predictor = ridge.pred, plot = TRUE)

Data: ridge.pred in 121 controls (test$delay 0) < 539 cases (test$delay 1).
Area under the curve: 0.6965762
```

```
ridge.pred   0   1
         0  10   0
         1 111 539
```

0.831818181818182



3.3 Discriminant Analysis

3.3.1 LDA

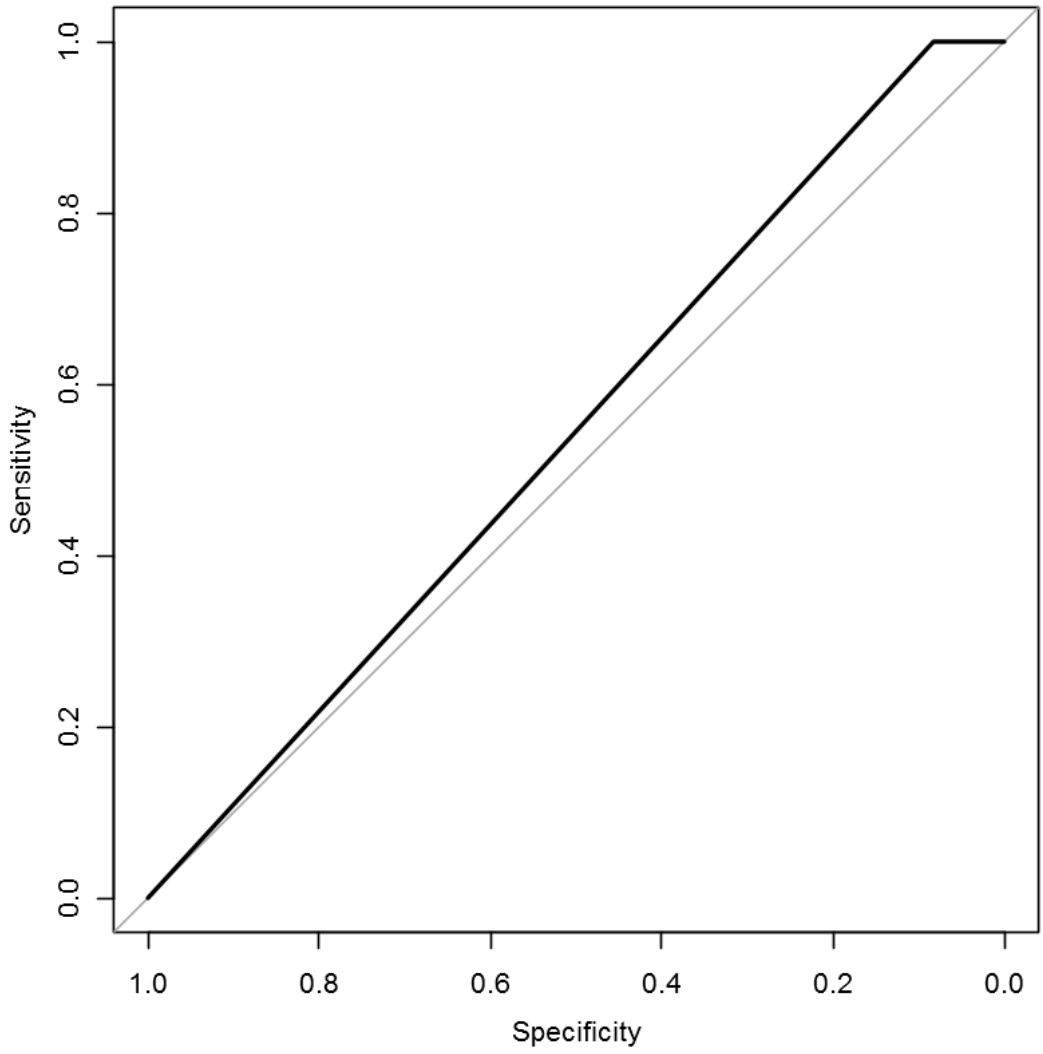
```
In [55]: lda.fit <- lda(formula, data=train)
         lda.pred <- predict(lda.fit, test)$class
         (lda.roc <- roc(test$delay,as.numeric(lda.pred),plot=TRUE)) #0.5413223
         (lda.confTable <- table(lda.pred, test$delay))
         (lda.acc <- sum(diag(lda.confTable))/sum(lda.confTable)) #0.831818181818182
```

```
Call:
roc.default(response = test$delay, predictor = as.numeric(lda.pred),     plot = TRUE)
```

```
Data: as.numeric(lda.pred) in 121 controls (test$delay 0) < 539 cases (test$delay 1).
Area under the curve: 0.5413223
```

```
lda.pred   0   1
       0  10   0
       1 111 539
```

0.831818181818182

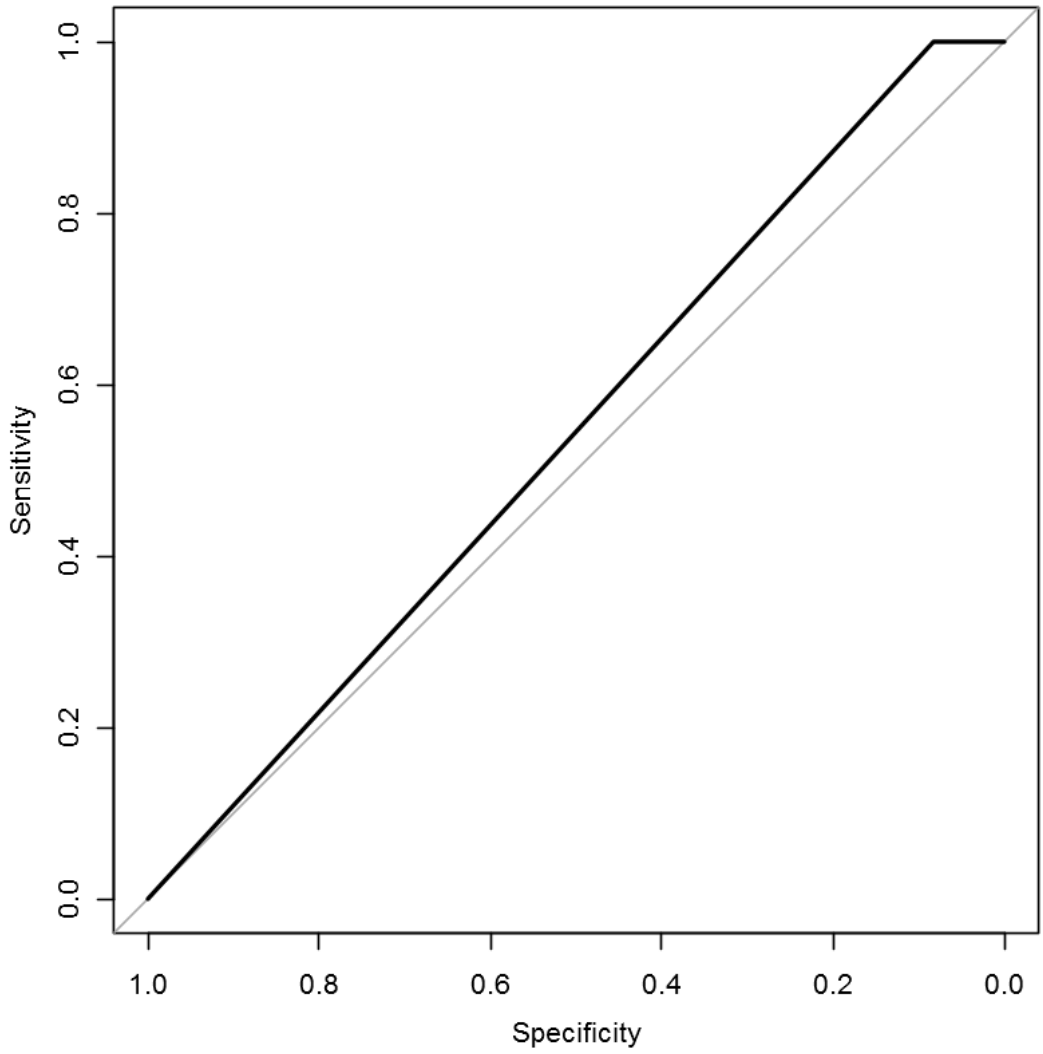


```
In [59]: # cross validation
         ctrl <- trainControl(method="repeatedcv",repeats=5)
         lda.fit1 <- train(as.factor(delay)~., data=train,method="lda",preProcess='center',trC
         summary(lda.fit1)
         print(lda.fit1)
         lda.pred1 <- predict(lda.fit1, test)
         (lda.roc1 <- roc(test$delay,as.numeric(lda.pred1),plot=TRUE)) #0.5413223
         (lda.confTable1 <- table(lda.pred1, test$delay))
         (lda.acc1 <- sum(diag(lda.confTable1))/sum(lda.confTable1)) #0.831818181818182
```

```
            Length Class      Mode
prior        2     -none-     numeric
counts       2     -none-     numeric
means       28     -none-     numeric
scaling     14     -none-     numeric
lev          2     -none-     character
svd          1     -none-     numeric
N            1     -none-     numeric
call         3     -none-     call
xNames      14     -none-     character
problemType  1     -none-     character
tuneValue    1     data.frame list
obsLevels    2     -none-     character
param        0     -none-     list


Linear Discriminant Analysis

1541 samples
  14 predictor
   2 classes: '0', '1'

Pre-processing: centered (14)
Resampling: Cross-Validated (10 fold, repeated 5 times)
Summary of sample sizes: 1387, 1386, 1387, 1387, 1387, 1387, ...
Resampling results:

  Accuracy      Kappa
  0.8150630348  0.1076056185




Call:
roc.default(response = test$delay, predictor = lda.pred1, plot = TRUE)

Data: lda.pred1 in 121 controls (test$delay 0) < 539 cases (test$delay 1).
Area under the curve: 0.5413223
```

```
lda.pred1   0   1
        1  10   0
        2 111 539
```

0.831818181818182



3.3.2 QDA

```
In [60]: table(flightPerf$weather) # weahter=1 only 32
```

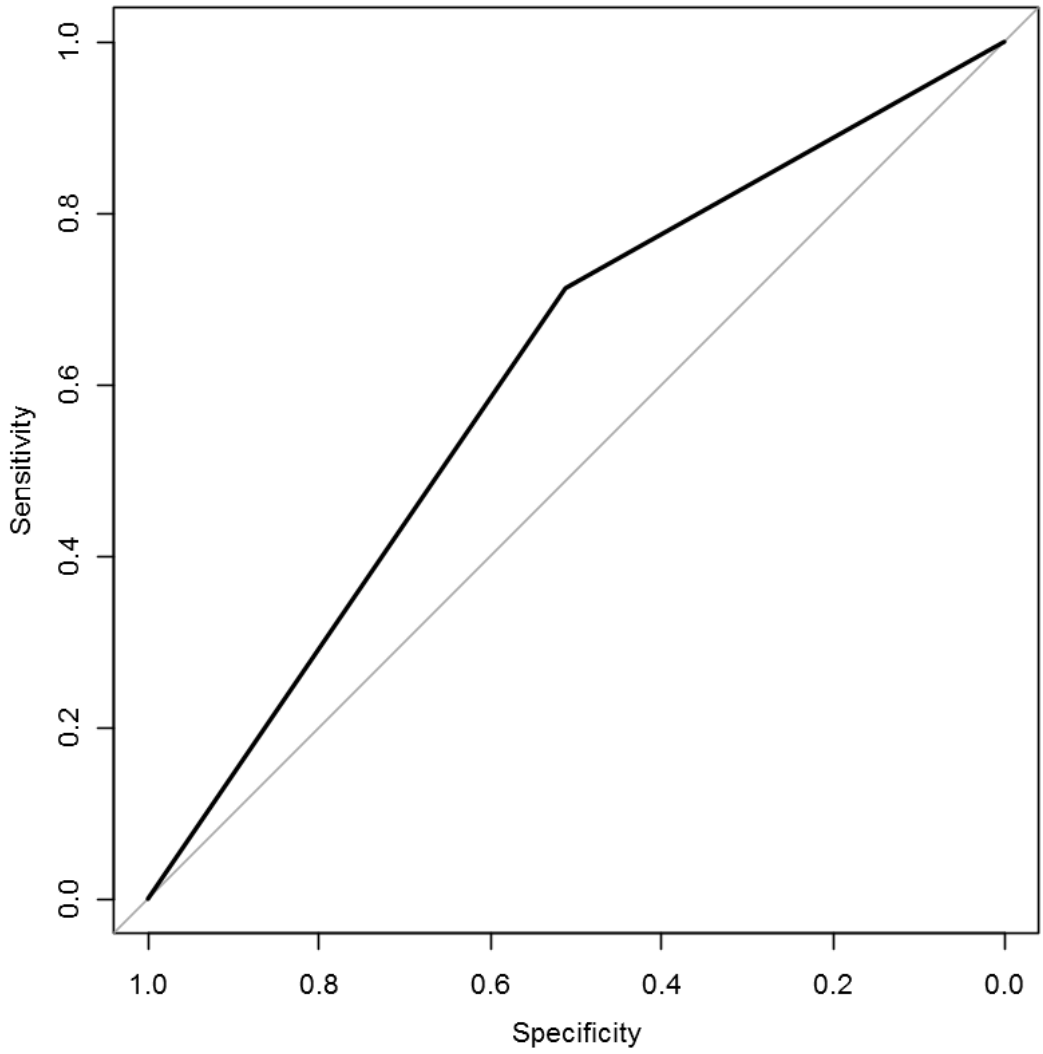
```
   0    1
2169   32
```

```
In [62]: qda.fit <- qda(as.factor(delay)~schedtime+carrierDH+carrierDL+carrierMQ+carrierOH+carr
                        +destJFK+destLGA+date+originDCA+originIAD, data=train) # remove weathe
         qda.pred <- predict(qda.fit, test)$class
         (qda.roc <- roc(test$delay,as.numeric(qda.pred),plot=TRUE)) #0.6124136
         (qda.confTable <- table(qda.pred, test$delay))
         (qda.acc <- sum(diag(qda.confTable))/sum(qda.confTable)) #0.6757575757575576
```

```
Call:
roc.default(response = test$delay, predictor = as.numeric(qda.pred),     plot = TRUE)

Data: as.numeric(qda.pred) in 121 controls (test$delay 0) < 539 cases (test$delay 1).
Area under the curve: 0.6124136
```

```
qda.pred   0   1
       0  62 155
       1  59 384
```

0.675757575757576



In [66]:
```
# cross validation
ctrl <- trainControl(method="repeatedcv",repeats=5)
qda.fit1 <- train(as.factor(delay)~schedtime+carrierDH+carrierDL+carrierMQ+carrierOH+
                  +destJFK+destLGA+date+originDCA+originIAD, data=train,method="qda",prel
summary(qda.fit1)

qda.pred1 <- predict(qda.fit1, test)
(qda.roc1 <- roc(test$delay,as.numeric(qda.pred1),plot=TRUE)) #0.6124136

(qda.confTable1 <- table(qda.pred1, test$delay))
(qda.acc1 <- sum(diag(qda.confTable1))/sum(qda.confTable1)) #0.67575757575757576
```

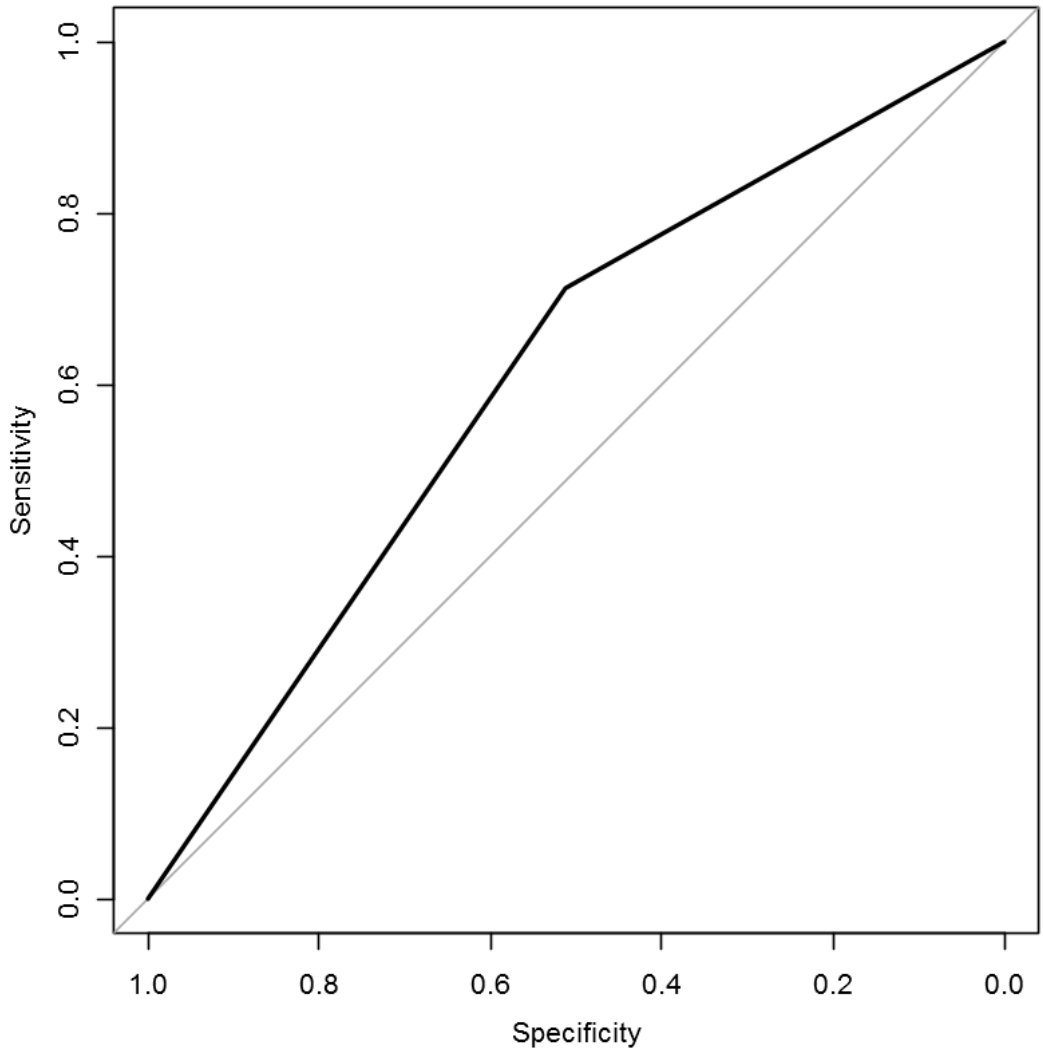
```
   Length Class      Mode
```

```
prior         2    -none-     numeric
counts        2    -none-     numeric
means        26    -none-     numeric
scaling     338    -none-     numeric
ldet          2    -none-     numeric
lev           2    -none-     character
N             1    -none-     numeric
call          3    -none-     call
xNames       13    -none-     character
problemType   1    -none-     character
tuneValue     1    data.frame list
obsLevels     2    -none-     character
param         0    -none-     list
```

```
Call:
roc.default(response = test$delay, predictor = as.numeric(qda.pred1),     plot = TRUE)

Data: as.numeric(qda.pred1) in 121 controls (test$delay 0) < 539 cases (test$delay 1).
Area under the curve: 0.6124136
```

```
qda.pred1   0   1
        0  62 155
        1  59 384
```

0.675757575757576



3.4 Support Vector Machine

3.4.1 Linear

```
In [67]: tune.out1 <- tune(svm, as.factor(delay)~., data=train,kernel='linear',ranges=list(cost
         summary(tune.out1) # best performance: cost 0.01
```

```
Parameter tuning of 'svm':

- sampling method: 10-fold cross validation

- best parameters:
```

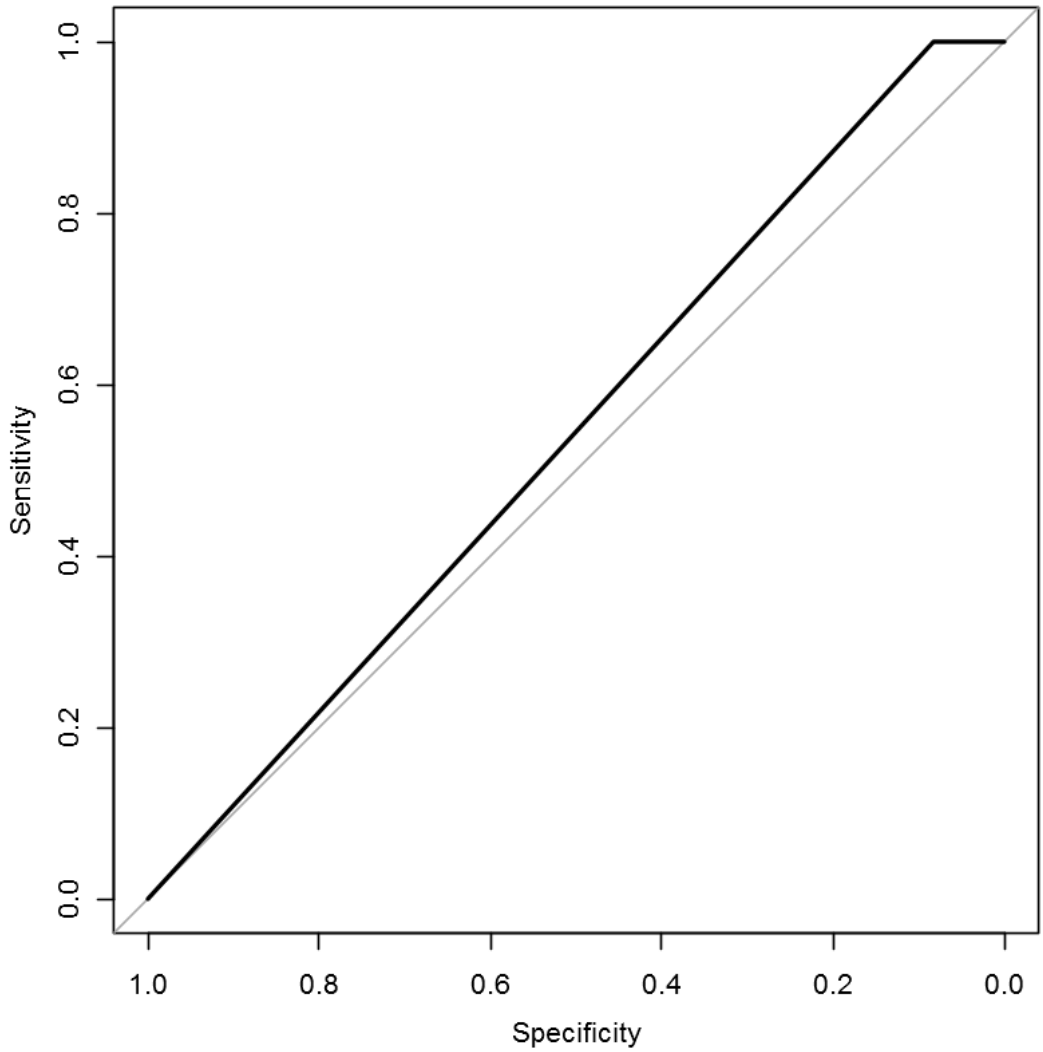
```
 cost
 0.01

- best performance: 0.1849602011

- Detailed performance results:
     cost        error    dispersion
1    0.01 0.1849602011 0.02594266192
2    0.10 0.1849602011 0.02594266192
3    1.00 0.1849602011 0.02594266192
4    5.00 0.1849602011 0.02594266192
5   10.00 0.1849602011 0.02594266192
6  100.00 0.1849602011 0.02594266192
7 1000.00 0.1849602011 0.02594266192
```

```
In [69]: svm.fit1 <- svm(as.factor(delay)~.,data=train,kernel='linear',cost=0.01)
         svm.pred1 <- predict(svm.fit1,test)
         svm.pred1 <- as.numeric(svm.pred1)
         (svm.roc1 <- roc(test$delay,svm.pred1,plot=TRUE)) #0.5413223

         (svm.confTable1 <- table(svm.pred1, test$delay))
         (svm.acc1 <- sum(diag(svm.confTable1))/sum(svm.confTable1)) #0.831818181818182
```

```
Call:
roc.default(response = test$delay, predictor = svm.pred1, plot = TRUE)

Data: svm.pred1 in 121 controls (test$delay 0) < 539 cases (test$delay 1).
Area under the curve: 0.5413223
```

```
svm.pred1   0   1
        1  10   0
        2 111 539
```

0.831818181818182



3.4.2 Polynomial

```
In [70]: tune.out2 <- tune(svm, as.factor(delay)~., data=train,kernel='polynomial',ranges=list
         summary(tune.out2) # best performance: cost 0.01, degree 2
```

```
Parameter tuning of 'svm':

- sampling method: 10-fold cross validation

- best parameters:
 cost degree
    5      3
```

```
- best performance: 0.1830121491

- Detailed performance results:
      cost degree        error    dispersion
1     0.01      2 0.1849518224 0.02588819600
2     0.10      2 0.1849518224 0.02588819600
3     1.00      2 0.1875492250 0.02501749210
4     5.00      2 0.1849518224 0.02533946082
5    10.00      2 0.1849518224 0.02533946082
6   100.00      2 0.1888395475 0.02533197475
7  1000.00      2 0.1888395475 0.02533197475
8     0.01      3 0.1849518224 0.02588819600
9     0.10      3 0.1849518224 0.02588819600
10    1.00      3 0.1868998743 0.02412758881
11    5.00      3 0.1830121491 0.02691788600
12   10.00      3 0.1843108504 0.02574940640
13  100.00      3 0.1895014663 0.02474536956
14 1000.00      3 0.1869040637 0.02638249157
15    0.01      4 0.1849518224 0.02588819600
16    0.10      4 0.1849518224 0.02588819600
17    1.00      4 0.1875492250 0.02347155294
18    5.00      4 0.1862589024 0.02536413856
19   10.00      4 0.1881985756 0.02603924741
20  100.00      4 0.1933766234 0.02625711532
21 1000.00      4 0.1907792208 0.02408304131
```

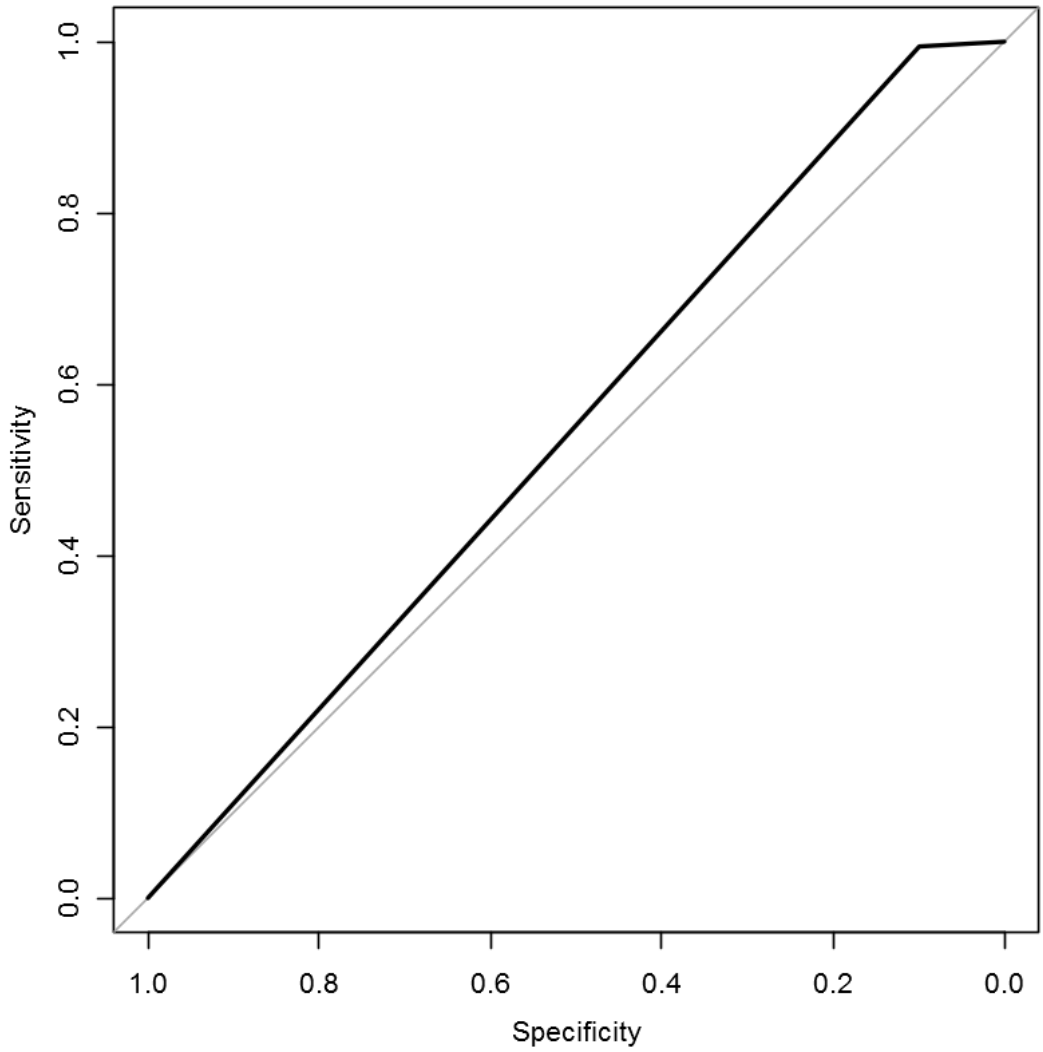
```
In [71]: svm.fit2 <- svm(as.factor(delay)~.,data=train,kernel='polynomial',cost=5, degree=3)
         svm.pred2 <- predict(svm.fit2,test)
         svm.pred2 <- as.numeric(svm.pred2)
         (svm.roc2 <- roc(test$delay,svm.pred2,plot=TRUE)) #0.5468038

         (svm.confTable2 <- table(svm.pred2, test$delay))
         (svm.acc2 <- sum(diag(svm.confTable2))/sum(svm.confTable2)) #0.83030303030303
```

```
Call:
roc.default(response = test$delay, predictor = svm.pred2, plot = TRUE)

Data: svm.pred2 in 121 controls (test$delay 0) < 539 cases (test$delay 1).
Area under the curve: 0.5468038
```

```
svm.pred2   0   1
        1  12   3
```

```
2 109 536
```

0.83030303030303



3.4.3 Radial

```
In [72]: tune.out3 <- tune(svm, as.factor(delay)~., data=train,kernel='radial',ranges=list(cost
         summary(tune.out3)  #best performance: cost 1, gamma 0.5
```

```
Parameter tuning of 'svm':
```

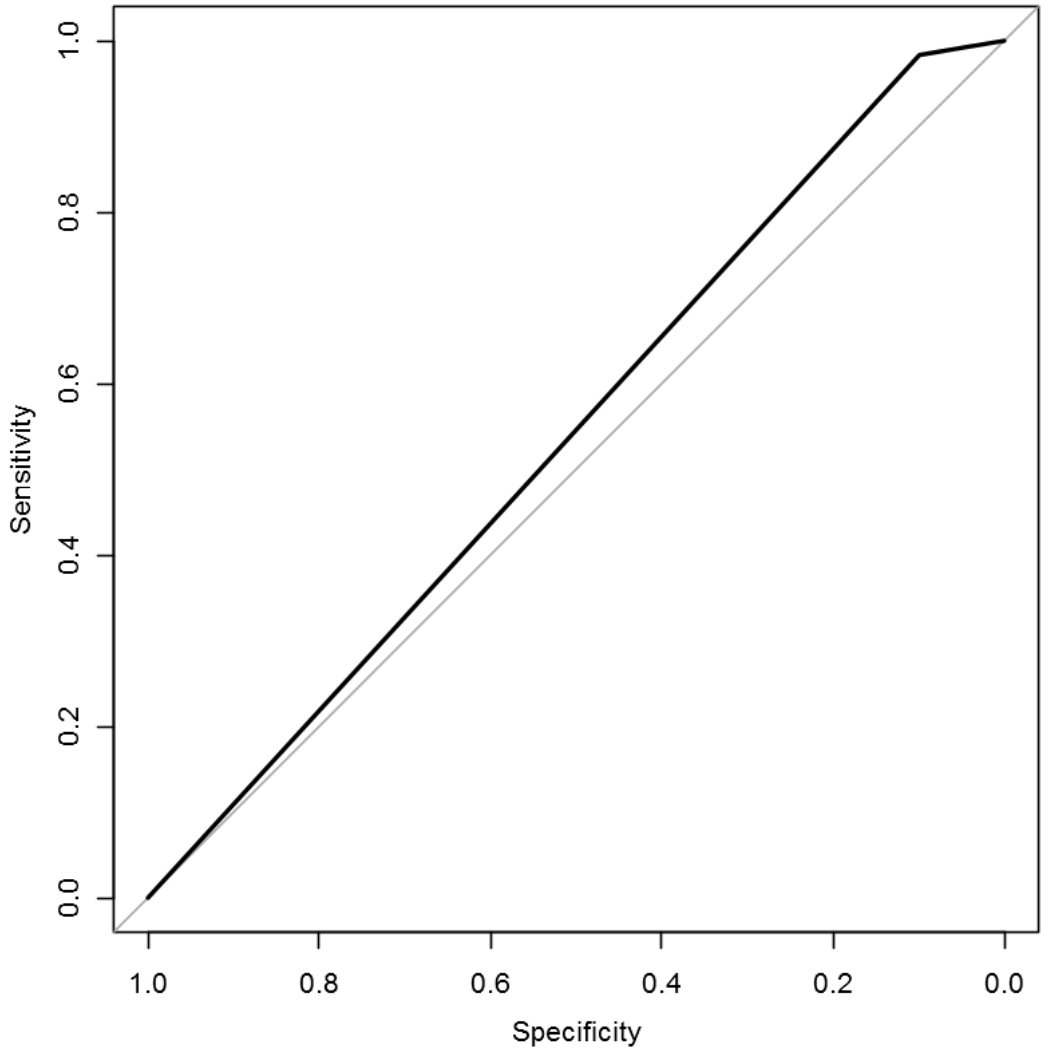
```
- sampling method: 10-fold cross validation

- best parameters:
 cost gamma
    1   0.5

- best performance: 0.1927230834

- Detailed performance results:
      cost gamma        error    dispersion
1     0.01   0.0 0.1992124005 0.02534977113
2     0.10   0.0 0.1992124005 0.02534977113
3     1.00   0.0 0.1992124005 0.02534977113
4     5.00   0.0 0.1992124005 0.02534977113
5    10.00   0.0 0.1992124005 0.02534977113
6   100.00   0.0 0.1992124005 0.02534977113
7  1000.00   0.0 0.1992124005 0.02534977113
8     0.01   0.5 0.1992124005 0.02534977113
9     0.10   0.5 0.1992124005 0.02534977113
10    1.00   0.5 0.1927230834 0.02881217302
11    5.00   0.5 0.1992291579 0.03613925527
12   10.00   0.5 0.2018223712 0.03539206085
13  100.00   0.5 0.2200083787 0.04076884806
14 1000.00   0.5 0.2413992459 0.03348071620
15    0.01   1.0 0.1992124005 0.02534977113
16    0.10   1.0 0.1992124005 0.02534977113
17    1.00   1.0 0.2011646418 0.03042842460
18    5.00   1.0 0.2115542522 0.03454304733
19   10.00   1.0 0.2226015920 0.03906900811
20  100.00   1.0 0.2355509007 0.02989821351
21 1000.00   1.0 0.2478843737 0.02828225906
22    0.01   2.0 0.1992124005 0.02534977113
23    0.10   2.0 0.1992124005 0.02534977113
24    1.00   2.0 0.2031085044 0.03486239162
25    5.00   2.0 0.2167448680 0.03243946244
26   10.00   2.0 0.2258357771 0.03213445651
27  100.00   2.0 0.2517804776 0.03053717386
28 1000.00   2.0 0.2731755341 0.03819178761
```

```
In [73]: svm.fit3 <- svm(as.factor(delay)~.,data=train,kernel='radial',cost=1, gamma=0.5)
         svm.pred3 <- predict(svm.fit3,test)
         svm.pred3 <- as.numeric(svm.pred3)
         (svm.roc3 <- roc(test$delay,svm.pred3,plot=TRUE)) #0.541238

         (svm.confTable3 <- table(svm.pred3, test$delay))
         (svm.acc3 <- sum(diag(svm.confTable3))/sum(svm.confTable3)) #0.821212121212121
```

```
Call:
roc.default(response = test$delay, predictor = svm.pred3, plot = TRUE)

Data: svm.pred3 in 121 controls (test$delay 0) < 539 cases (test$delay 1).
Area under the curve: 0.541238
```

```
svm.pred3   0   1
        1  12   9
        2 109 530
```

0.821212121212121



3.5 Neural Network

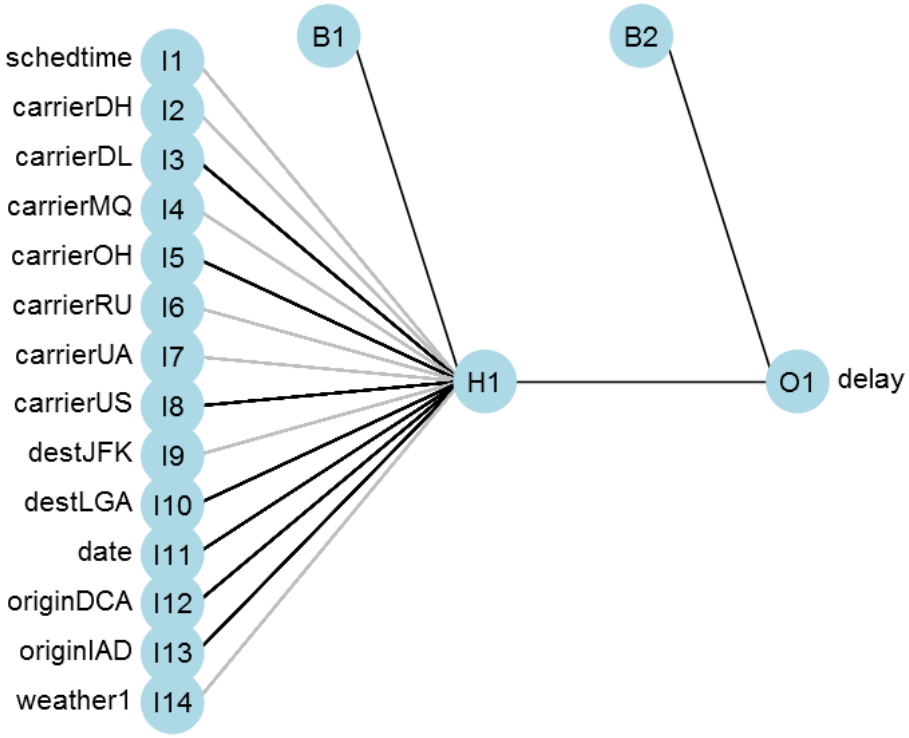
3.5.1 Model1: 1 layer, 1 neuron

In [74]:
```
nn1.fit <- neuralnet(formula=formula,data=train, hidden=1, linear.output=FALSE, stepma
# show the weights
nn1.fit$weights
```

1. (a)
   9.196607429
   -34.591641906
   -12.393005960
   3.659076478
   -77.495456263
   18.220798757
   -2.343907388
   -22.471124439
   34.911585500
   -9.291372130
   7.873098178
   2.096965799
   7.921734503
   10.006225412
   -82.618491148

   (b)
   0.7996747728
   1.6215450255

In [75]:
```
#Plot the neural network
plotnet(nn1.fit,rel_rsc=TRUE)
```

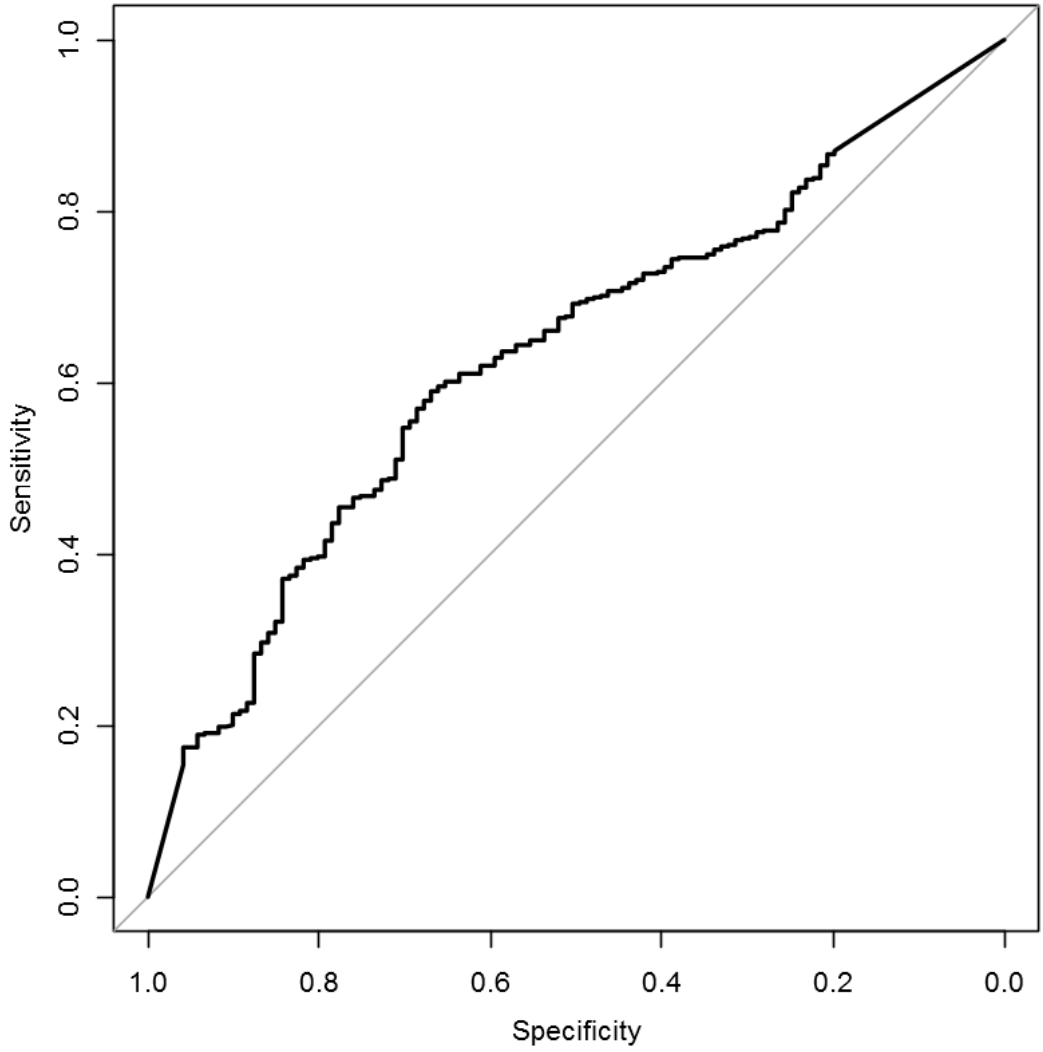



```
In [190]: nn1.pred <- compute(nn1.fit,test[,-15])$net.result
          # un-scale the predict values of delay
          #nn1.pred <- nn1.pred$net.result * (max(test$delay)-min(test$delay)) + min(test$dela
          # un-scale the actual vlaues of delay
          #delay <- test$delay * (max(test$delay)-min(test$delay)) + min(test$delay)
          # evaluation
          (nn1.roc <- roc(test$delay,as.numeric(nn1.pred),plot=TRUE)) #0.5206446
          #(nn1.bestThreh <- coords(nn1.roc, "best", ret = "threshold")) # best thrshold:0.699
          nn1.pred1 <- ifelse(nn1.pred > 0.69,1,0)
          (nn1.confTable <- table(nn1.pred1, test$delay))
          (nn1.acc <- sum(diag(nn1.confTable))/sum(nn1.confTable))  #0.604545454545454
```

```
Call:
roc.default(response = test$delay, predictor = as.numeric(nn1.pred),     plot = TRUE)

Data: as.numeric(nn1.pred) in 121 controls (test$delay 0) < 539 cases (test$delay 1).
Area under the curve: 0.6305678
```

```
nn1.pred1   0   1
        0  61 174
        1  60 365
```

0.645454545454546



3.5.2 Model2: 1 layer, 2 neurons

In [134]:
```
nn2.fit <- neuralnet(formula=formula,data=train, hidden=2, linear.output=FALSE, stepr
nn2.fit$weights
```

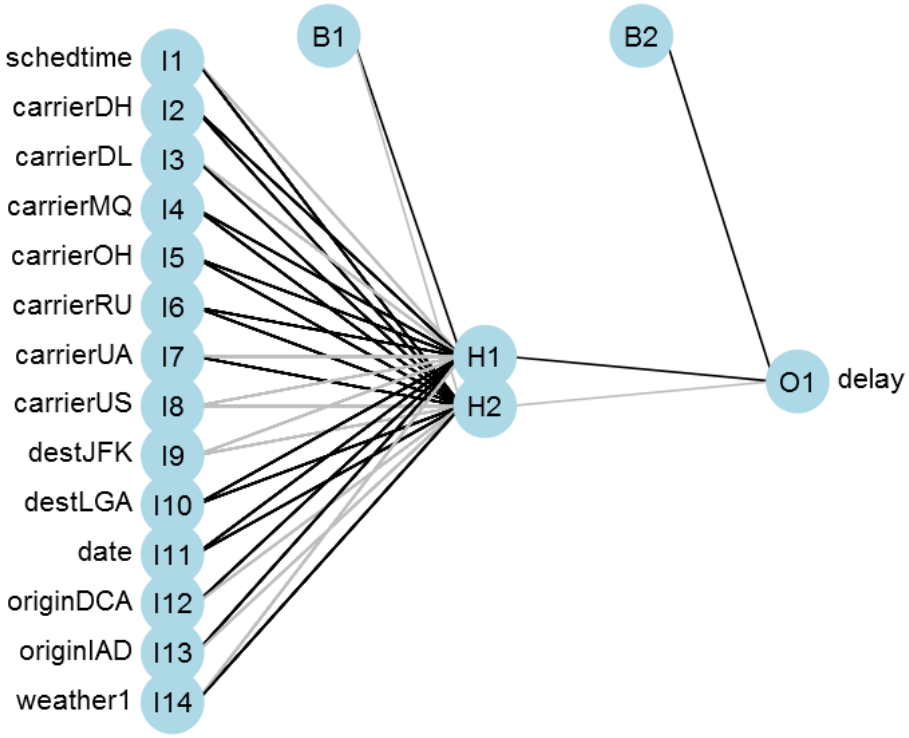
1. (a)

22.168120455	-1.67633327511
-51.405275919	0.85125917938
7.795655775	1.18401772448
-47.907042005	0.34624505116
18.927675650	1.86174444816
6.101743092	0.48018739377
25.194951699	1.01660480945
-107.791694799	0.80330171837
-45.528692075	-0.39723285927
-3.598434249	-0.64376891763
108.833992678	0.03457689134
19.862879302	0.37267242164
1.664911941	-0.70129417250
21.928087253	-0.26865253139
-413.624358105	40.07665975430

(b)

0.3225495615
2.7335355482
-4.2695867641

In [135]:
```
#Plot the neural network
plotnet(nn2.fit,rel_rsc=TRUE)
```

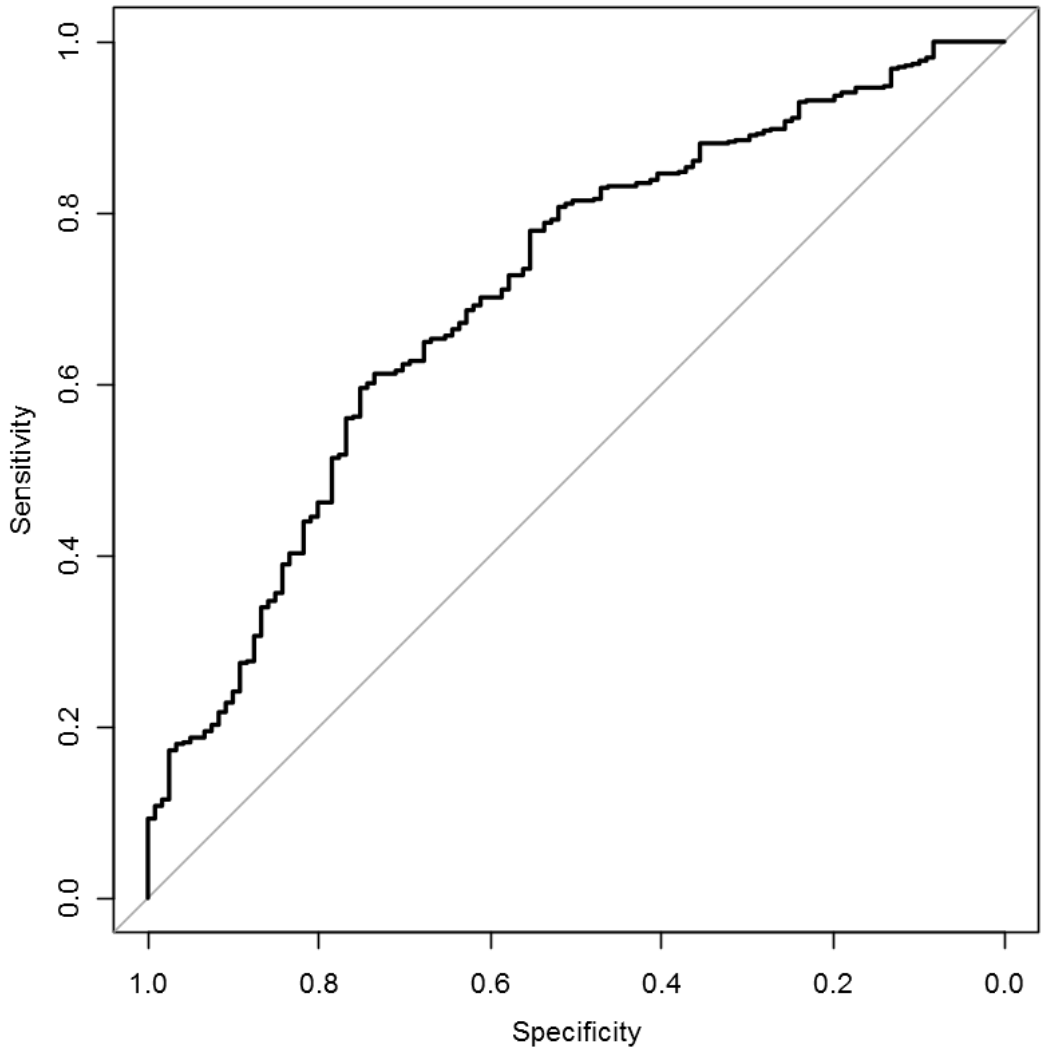



```
In [173]: nn2.pred <- compute(nn2.fit,test[,-15])$net.result
          # evaluation
          #(nn2.bestThreh <- coords(nn2.roc, "best", ret = "threshold")) # best thrshold:0.856
          nn2.pred1 <- ifelse(nn2.pred > 0.2,1,0)
          (nn2.confTable <- table(nn2.pred1, test$delay))
          (nn2.acc <- sum(diag(nn2.confTable))/sum(nn2.confTable))  #0.806060606060606
          (nn2.roc <- roc(test$delay,as.numeric(nn2.pred),plot=TRUE)) #0.7068032
```

```
nn2.pred1   0   1
        0  10   0
        1 111 539
```

0.831818181818182

```
Call:
roc.default(response = test$delay, predictor = as.numeric(nn2.pred),     plot = TRUE)

Data: as.numeric(nn2.pred) in 121 controls (test$delay 0) < 539 cases (test$delay 1).
Area under the curve: 0.7068032
```



3.5.3 Model3: 1 layer, 3 neurons

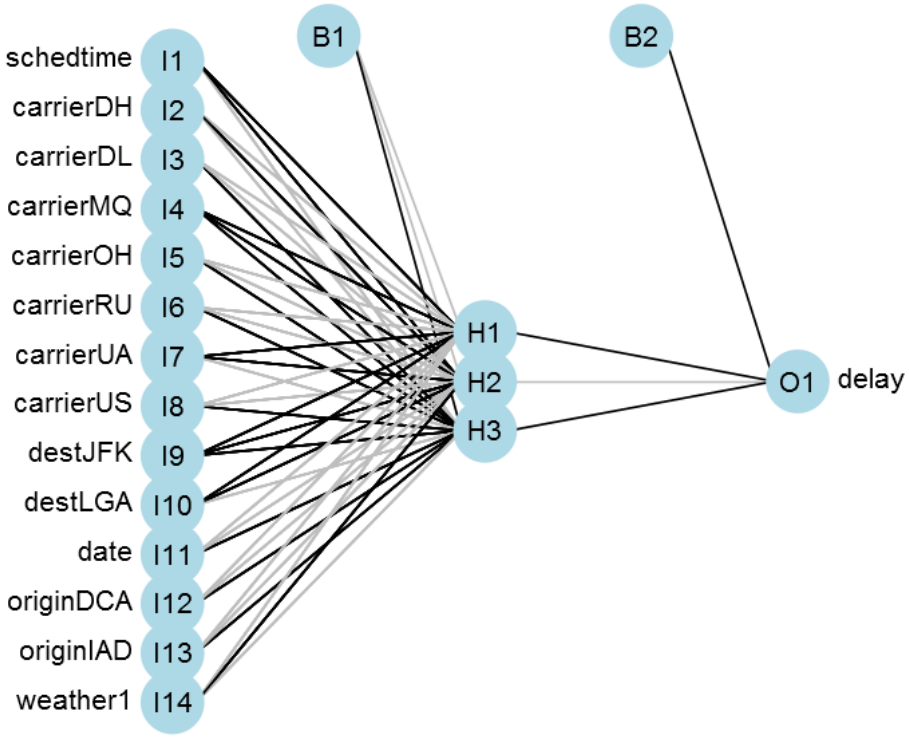
```
In [90]: nn3.fit <- neuralnet(formula=formula,data=train, hidden=3, linear.output=FALSE, stepma
         nn3.fit$weights
```

		-0.2801182249	-1.0346716498	28.1400432820
		8.2645089906	3.1038821014	-82.4987315731
		-3.2980798511	0.4592065094	-2.9904881265
		-6.5446495132	-2.1952734940	71.9269113793
		5.6953068469	3.3921392051	0.1219045951
		-438.1382927130	-7.5220097893	6.2381739994
		-5.6245169278	-0.3844306350	71.0700160422
1.	(a)	1.0908142315	1.3803957237	-71.4360083496
		-1.5385013259	-75.8223608411	75.6910698953
		10.1112423280	5.8182910872	7.3363198629
		0.6703171193	0.1347209519	-1.3907483080
		-5.6090104236	-1.6692742946	24.0760448816
		-6.3376387983	-1.8618595219	15.1767198260
		-2.7167295736	-0.9142114861	94.9263308127
		-440.5377665258	116.1237320319	-443.2636516205

(b)

0.2683866164
3.7733403372
-4.2493248394
2.0285953124

In [91]:
```
#Plot the neural network
plotnet(nn3.fit,rel_rsc=TRUE)
```

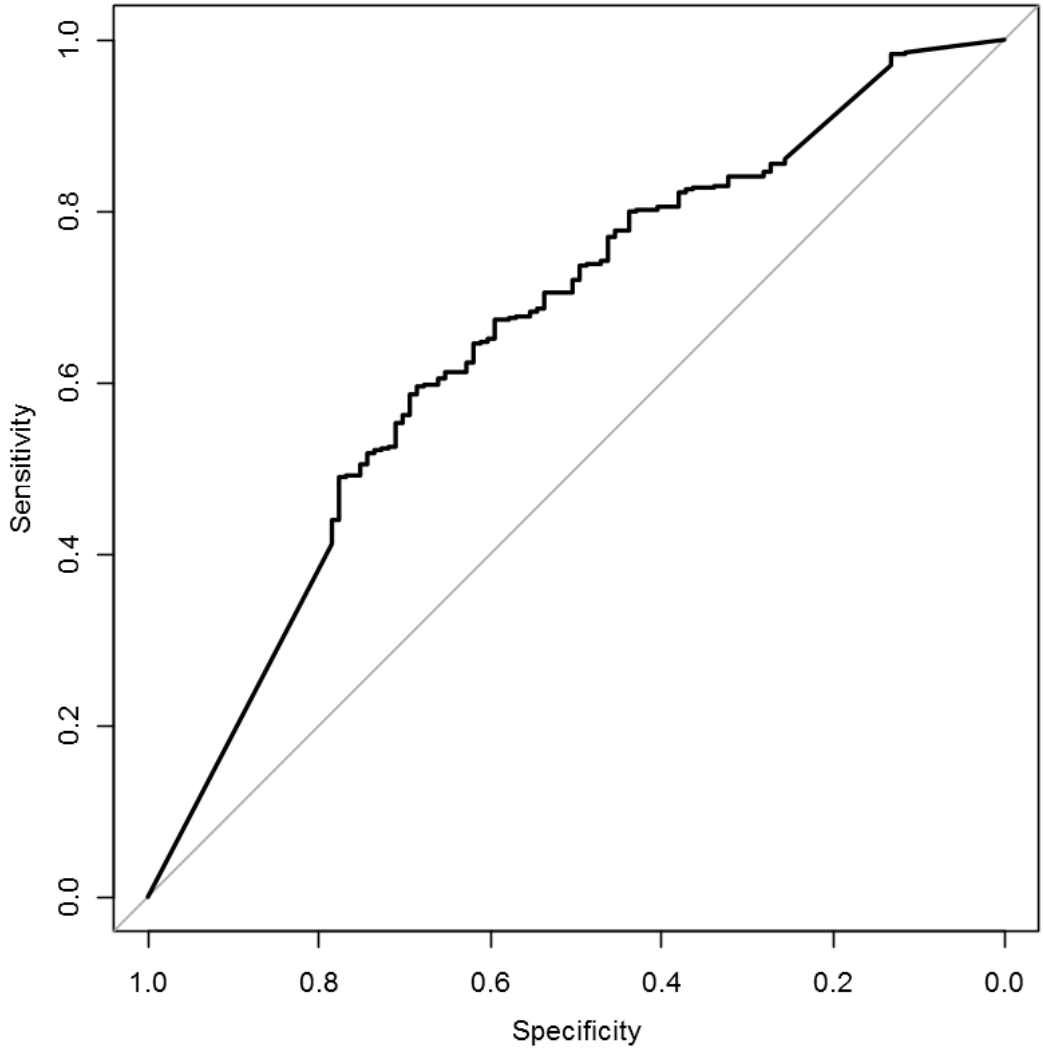


In [207]:
```
nn3.pred <- compute(nn3.fit,test[,-15])$net.result
# un-scale the predict values of delay
#nn2.pred <- nn2.pred$net.result * (max(test$delay)-min(test$delay)) + min(test$dela
# un-scale the actual values of delay
#delay <- test$delay * (max(test$delay)-min(test$delay)) + min(test$delay)
# evaluation
(nn3.roc <- roc(test$delay,as.numeric(nn3.pred),plot=TRUE)) #0.6672135
#(nn3.bestThreh <- coords(nn3.roc, "best", ret = "threshold")) # best thrshold:0.856
nn3.pred1 <- ifelse(nn3.pred > 0.04,1,0)
(nn3.confTable <- table(nn3.pred1, test$delay))
(nn3.acc <- sum(diag(nn3.confTable))/sum(nn3.confTable))  #0.812121212121212
```

```
Call:
roc.default(response = test$delay, predictor = as.numeric(nn3.pred),     plot = TRUE)

Data: as.numeric(nn3.pred) in 121 controls (test$delay 0) < 539 cases (test$delay 1).
Area under the curve: 0.6585888
```

```
nn3.pred1   0   1
        0  16   9
        1 105 530
```

0.827272727272727



3.5.4 Model4: 2 layers, 2 + 2 neurons

In [138]:
```
nn4.fit <- neuralnet(formula=formula,data=train, hidden=c(2,2), linear.output=FALSE,
#Plot the neural network
nn4.fit$weights
```

1. (a)

-3.5175790179	-11.466101354
-14.3418017528	-8.037585523
-5.7628897818	-17.927480696
-0.3067885398	-25.599067777
-1.8497872641	-12.225749397
-1.3351194890	-53.890740002
-1.2359891855	-42.310526439
-4.2442380835	12.147714362
-0.7514964732	-50.698775608
6.5058399019	17.670928451
2.6895446966	13.784557170
10.3678258640	54.027666748
12.4047203749	12.454525307
7.1706219801	-22.453362348
-728.5622128338	62.174435678

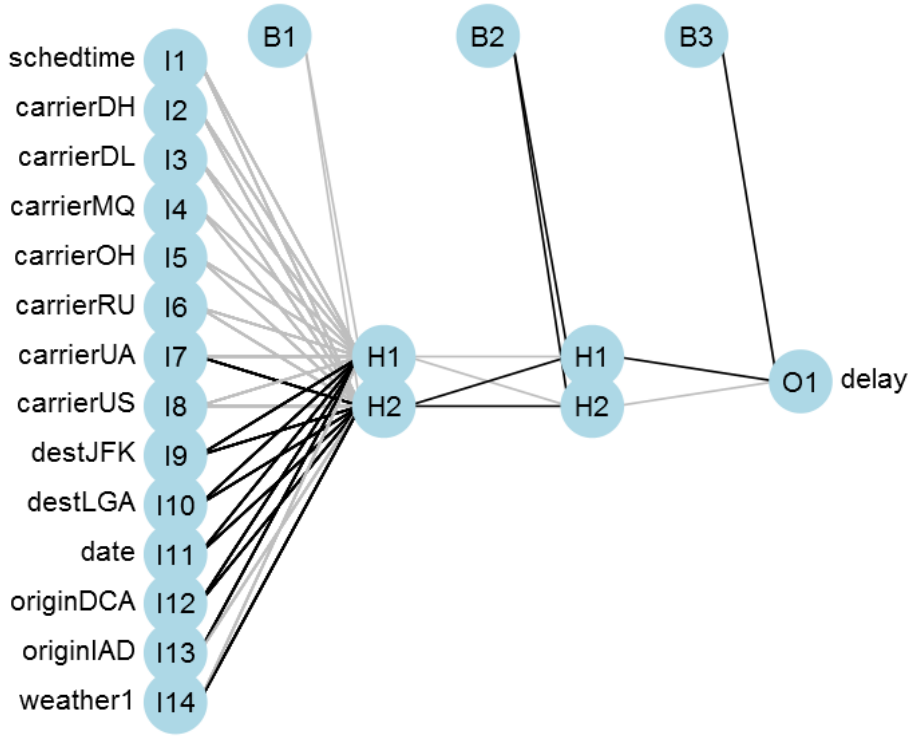
(b)

1.65990621	0.3826130386
-14.90764527	-27.6377530002
3391.51513793	27.3443890081

(c)

1.883580622
2.832256661
-6.181024256

In [140]:
```
#Plot the neural network
plotnet(nn4.fit,rel_rsc=TRUE)
```

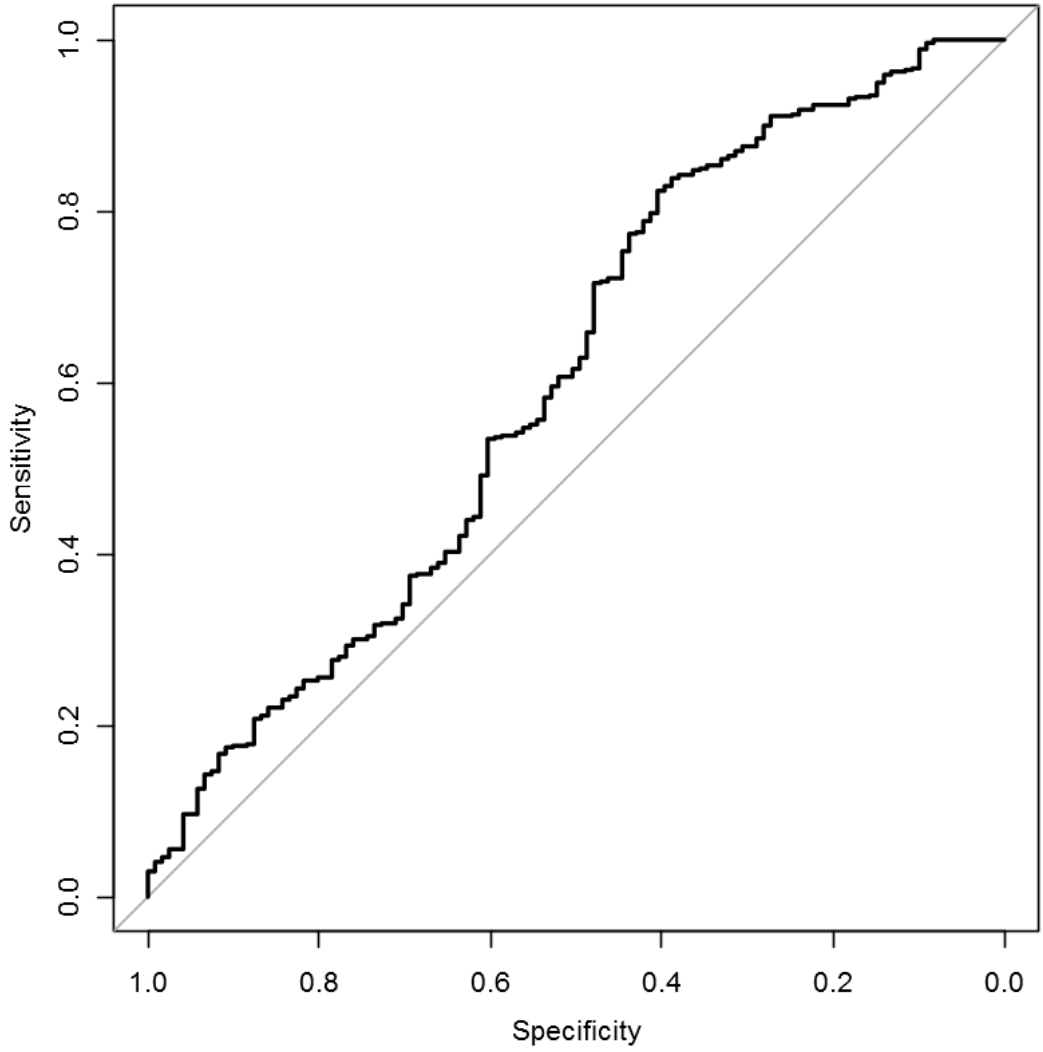



```
In [216]: nn4.pred <- compute(nn4.fit,test[,-15])$net.result
          # un-scale the predict values of delay
          #nn3.pred <- nn3.pred$net.result * (max(test$delay)-min(test$delay)) + min(test$dela
          # un-scale the actual values of delay
          #delay <- test$delay * (max(test$delay)-min(test$delay)) + min(test$delay)
          # evaluation
          (nn4.roc <- roc(test$delay,as.numeric(nn4.pred),plot=TRUE)) #0.6585888
          #(nn4.bestThreh <- coords(nn4.roc, "best", ret = "threshold")) # best thrshold:0.777
          nn4.pred1 <- ifelse(nn4.pred > 0.2,1,0)
          (nn4.confTable <- table(nn4.pred1, test$delay))
          (nn4.acc <- sum(diag(nn4.confTable))/sum(nn4.confTable))  #0.81969696969697
```

```
Call:
roc.default(response = test$delay, predictor = as.numeric(nn4.pred),     plot = TRUE)

Data: as.numeric(nn4.pred) in 121 controls (test$delay 0) < 539 cases (test$delay 1).
Area under the curve: 0.6022785
```

```
nn4.pred1   0   1
        0  12  13
        1 109 526
```

0.815151515151515



4 Conclusion

Modelling Method	Methods	Model Specification	Accuracy(Confusion Matrix)	Roc Plot
kNN		k=13	0.8030	0.5301
Logistic Regression	StepAIC	AIC:1412.45 log(p(delay=1)/(1-p(delay=1)))=1.7-0.91*schedtime*+*0.69*carrierDL+1.27*carrierUS*-*16.9*weather1-0.30*carrierMQ	0.8318	0.6865
Logistic Regression	Lasso Regression	optimal lambda:0.00649	0.8318	0.6891
Logistic Regression	Radge Regression	optimal lambda:0.02226	0.8318	0.6965
Discriminant Analysis	LDA		0.8318	0.5413
Discriminant Analysis	QDA		0.6758	0.6124
Support Vector Machine	Linear	cost:0.01	0.8318	0.5413
Support Vector Machine	Polynomial	cost:5,degree:3	0.8303	0.5468
Support Vector Machine	Radial	cost:1,gamma:0.5	0.8212	0.5412
Neural Network	1 layer, 1 neuron	threshold: 0.69	0.6454	0.6305
Neural Network	1 layer, 2 neurons	threshold: 0.2	0.8318	0.7068
Neural Network	1 layer, 3 neurons	threshold: 0.4	0.8273	0.6585
Neural Network	2 layers, 2+2 neurons	threshold: 0.2	0.8151	0.6022

From the result table, we can tell: 1. The highest accuracy is 0.8313, which we can get from Logistic Regression, Linear Discrimiant Analysis(LDA), Support Vector Machine(Linear with cost:0.01), and Neural Network(with 1 layer,2 neurons). 2. In these four best models, the AUC value from high to low is Neural Network(0.7068) > Logistic Regression(0.68) > LDA = SVM(linear)(0.5412) In this way, neural network is the best model. In real world application, if the dataset has small predictors and observations, I would choose Logistic regression model. if the dataset has lots of predictors and observations, I would choose Neural Network model.