# absolute final version

*Chujun Chen*

*5/2/2018*

```
library(rminer)
bank=read.table("/Users/chujunchen/Desktop/COLUMBIA/Statistical_Learning/project/bank-additional/bank-a
bank[bank=="unknown"] <- NA #replace 'unknown' with 'N/A'
print(class(bank)) # show class
```

```
## [1] "data.frame"
```

```
print(names(bank)) # show attributes
```

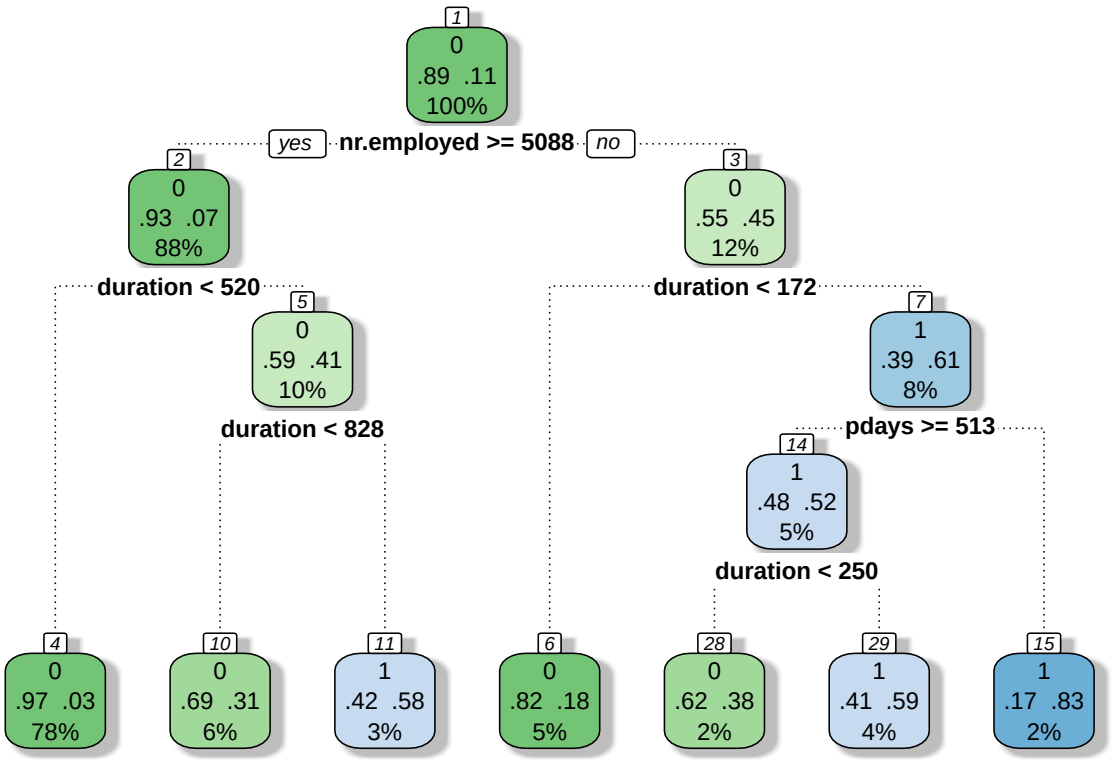
```
##  [1] "age"            "job"            "marital"        "education"
##  [5] "default"        "housing"        "loan"           "contact"
##  [9] "month"          "day_of_week"    "duration"       "campaign"
## [13] "pdays"          "previous"       "poutcome"       "emp.var.rate"
## [17] "cons.price.idx" "cons.conf.idx"  "euribor3m"      "nr.employed"
## [21] "y"
```

```
bank$y<-ifelse(bank$y =='yes', 1,0) #mark y=yes as 1 and y=no as 0
bank$y<-as.factor(bank$y)
## choose 2/3 of the data as training data
smp_size <- floor(0.67 * nrow(bank))
## set the seed to make your partition reproductible
set.seed(123)
##Produce training dataset and testing dataset
train_ind <- sample(seq_len(nrow(bank)), size = smp_size)
train <- bank[train_ind, ]
test <- bank[-train_ind, ]
```

```
library(rpart)
library(rattle)
```

```
## Rattle: A free graphical interface for data mining with R.
## Version 5.0.14 Copyright (c) 2006-2017 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
#train decision tree model with training data
bank.rpart <- rpart(y ~ ., data = train)
#plot decision tree
fancyRpartPlot(bank.rpart)
```




Rattle 2018−May−09 23:45:41 chujunchen

```
#predict basing on testing data, set the type as 'classification'
predictions <- predict(bank.rpart, test, type = "class")
#produce confusion matrix
confusion.matrix <- prop.table(table(predictions, test$y))
#show accuracy of the prediction
accuracy <- confusion.matrix[1,1] + confusion.matrix[2,2]
accuracy
```

```
## [1] 0.9123814
```

```
##Assign code to different kind of character features and set them as numeric
bank$job = c('admin.'=1,'blue-collar'=2,'entrepreneur'=3,'housemaid'=4,'management'=5,'retired'=6,'self
bank$marital=c('single'=1,'married'=2,'divorced'=3)[ as.numeric(bank$marital)]
bank$education=c('basic.4y'=1,'basic.6y'=2,'basic.9y'=3,'high.school'=4,'illiterate'=0,'university.degr
bank$default<-ifelse(bank$default =='yes', 1,0)
bank$housing<-ifelse(bank$housing=="yes",1,0)
bank$loan<-ifelse(bank$loan=='yes',1,0)
bank$contact<-ifelse(bank$contact=='cellular',1,0)
bank$month=c('apr'=4,'may'=5,'aug'=8,'dec'=12,'jul'=7,'jun'=5,'mar'=3,'sep'=9,'oct'=10,'nov'=11)[ as.nu
bank$day_of_week=c('mon'=1,'tue'=2,'wed'=3,'thu'=4,'fri'=5)[as.numeric(bank$day_of_week)]
bank$poutcome=c('nonexistent'=0,'failure'=-1,'success'=1)[as.numeric(bank$poutcome)]
summary(bank)
```

```
##       age             job           marital        education
##  Min.   :17.00   Min.   : 0.00   Min.   :1.000   Min.   :0.000
##  1st Qu.:32.00   1st Qu.: 1.00   1st Qu.:2.000   1st Qu.:3.000
##  Median :38.00   Median : 2.00   Median :2.000   Median :4.000
##  Mean   :40.02   Mean   : 4.49   Mean   :2.169   Mean   :4.161
##  3rd Qu.:47.00   3rd Qu.: 9.00   3rd Qu.:3.000   3rd Qu.:6.000
```

```
##  Max.   :98.00   Max.   :10.00   Max.   :3.000   Max.   :6.000
##                  NA's   :330     NA's   :80      NA's   :1731
##     default        housing            loan           contact
##  Min.   :0      Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0      1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0      Median :1.0000   Median :0.0000   Median :1.0000
##  Mean   :0      Mean   :0.5367   Mean   :0.1554   Mean   :0.6347
##  3rd Qu.:0      3rd Qu.:1.0000   3rd Qu.:0.0000   3rd Qu.:1.0000
##  Max.   :1      Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##  NA's   :8597   NA's   :990      NA's   :990
##      month         day_of_week       duration         campaign
##  Min.   : 3.000   Min.   :1.000   Min.   :   0.0   Min.   : 1.000
##  1st Qu.: 3.000   1st Qu.:2.000   1st Qu.: 102.0   1st Qu.: 1.000
##  Median : 5.000   Median :3.000   Median : 180.0   Median : 2.000
##  Mean   : 6.327   Mean   :3.005   Mean   : 258.3   Mean   : 2.568
##  3rd Qu.: 9.000   3rd Qu.:4.000   3rd Qu.: 319.0   3rd Qu.: 3.000
##  Max.   :12.000   Max.   :5.000   Max.   :4918.0   Max.   :56.000
##
##      pdays          previous        poutcome        emp.var.rate
##  Min.   :  0.0   Min.   :0.000   Min.   :-1.0000   Min.   :-3.40000
##  1st Qu.:999.0   1st Qu.:0.000   1st Qu.:-1.0000   1st Qu.:-1.80000
##  Median :999.0   Median :0.000   Median :-1.0000   Median : 1.10000
##  Mean   :962.5   Mean   :0.173   Mean   :-0.8301   Mean   : 0.08189
##  3rd Qu.:999.0   3rd Qu.:0.000   3rd Qu.:-1.0000   3rd Qu.: 1.40000
##  Max.   :999.0   Max.   :7.000   Max.   : 1.0000   Max.   : 1.40000
##
##  cons.price.idx  cons.conf.idx     euribor3m      nr.employed   y
##  Min.   :92.20   Min.   :-50.8   Min.   :0.634   Min.   :4964   0:36548
##  1st Qu.:93.08   1st Qu.:-42.7   1st Qu.:1.344   1st Qu.:5099   1: 4640
##  Median :93.75   Median :-41.8   Median :4.857   Median :5191
##  Mean   :93.58   Mean   :-40.5   Mean   :3.621   Mean   :5167
##  3rd Qu.:93.99   3rd Qu.:-36.4   3rd Qu.:4.961   3rd Qu.:5228
##  Max.   :94.77   Max.   :-26.9   Max.   :5.045   Max.   :5228
##
```

```
print(summary(bank$education))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   0.000   3.000   4.000   4.161   6.000   6.000    1731
```

```
meanage=mean(bank$education,na.rm=TRUE)
# subsitute NA by the average value
bank2=imputation("value",bank,"education",Value=meanage)
print("mean imputation age summary:")
```

```
## [1] "mean imputation age summary:"
```

```
print(summary(bank2$education))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   3.000   4.000   4.161   6.000   6.000
```

```
# substitute NA values by the most common value of bank$job
print("original job summary:")
```

```
## [1] "original job summary:"
```

```
print(summary(bank$job))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    0.00    1.00    2.00    4.49    9.00   10.00     330
```

```
bank2=imputation("value",bank2,"job",Value=as.numeric(names(which.max(table(bank$job)))))
print("mode imputation job summary:")
```

```
## [1] "mode imputation job summary:"
```

```
print(summary(bank2$job))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   1.000   2.000   4.462   9.000  10.000
```

```
print("original marital summary:")
```

```
## [1] "original marital summary:"
```

```
print(summary(bank$marital))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   1.000   2.000   2.000   2.169   3.000   3.000      80
```

```
bank2=imputation("value",bank2,"marital",Value=as.numeric(names(which.max(table(bank$marital)))))

bank2=imputation("value",bank2,"default",Value=as.numeric(names(which.max(table(bank$default)))
))

bank2=imputation("value",bank2,"housing",Value=as.numeric(names(which.max(table(bank$housing)))
))

bank2=imputation("value",bank2,"loan",Value=as.numeric(names(which.max(table(bank$loan)))
))

summary(bank2)
```

```
##       age             job            marital        education
##  Min.   :17.00   Min.   : 0.000   Min.   :1.000   Min.   :0.000
##  1st Qu.:32.00   1st Qu.: 1.000   1st Qu.:2.000   1st Qu.:3.000
##  Median :38.00   Median : 2.000   Median :2.000   Median :4.000
##  Mean   :40.02   Mean   : 4.462   Mean   :2.169   Mean   :4.161
##  3rd Qu.:47.00   3rd Qu.: 9.000   3rd Qu.:3.000   3rd Qu.:6.000
##  Max.   :98.00   Max.   :10.000   Max.   :3.000   Max.   :6.000
##     default            housing            loan           contact
##  Min.   :0.00e+00   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.00e+00   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.00e+00   Median :1.0000   Median :0.0000   Median :1.0000
##  Mean   :7.28e-05   Mean   :0.5479   Mean   :0.1517   Mean   :0.6347
##  3rd Qu.:0.00e+00   3rd Qu.:1.0000   3rd Qu.:0.0000   3rd Qu.:1.0000
##  Max.   :1.00e+00   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##      month         day_of_week       duration         campaign
##  Min.   : 3.000   Min.   :1.000   Min.   :   0.0   Min.   : 1.000
##  1st Qu.: 3.000   1st Qu.:2.000   1st Qu.: 102.0   1st Qu.: 1.000
##  Median : 5.000   Median :3.000   Median : 180.0   Median : 2.000
##  Mean   : 6.327   Mean   :3.005   Mean   : 258.3   Mean   : 2.568
##  3rd Qu.: 9.000   3rd Qu.:4.000   3rd Qu.: 319.0   3rd Qu.: 3.000
```

```
##  Max.   :12.000   Max.   :5.000   Max.   :4918.0   Max.   :56.000
##      pdays          previous        poutcome        emp.var.rate
##  Min.   :  0.0   Min.   :0.000   Min.   :-1.0000   Min.   :-3.40000
##  1st Qu.:999.0   1st Qu.:0.000   1st Qu.:-1.0000   1st Qu.:-1.80000
##  Median :999.0   Median :0.000   Median :-1.0000   Median : 1.10000
##  Mean   :962.5   Mean   :0.173   Mean   :-0.8301   Mean   : 0.08189
##  3rd Qu.:999.0   3rd Qu.:0.000   3rd Qu.:-1.0000   3rd Qu.: 1.40000
##  Max.   :999.0   Max.   :7.000   Max.   : 1.0000   Max.   : 1.40000
##  cons.price.idx  cons.conf.idx     euribor3m      nr.employed   y
##  Min.   :92.20   Min.   :-50.8   Min.   :0.634   Min.   :4964   0:36548
##  1st Qu.:93.08   1st Qu.:-42.7   1st Qu.:1.344   1st Qu.:5099   1: 4640
##  Median :93.75   Median :-41.8   Median :4.857   Median :5191
##  Mean   :93.58   Mean   :-40.5   Mean   :3.621   Mean   :5167
##  3rd Qu.:93.99   3rd Qu.:-36.4   3rd Qu.:4.961   3rd Qu.:5228
##  Max.   :94.77   Max.   :-26.9   Max.   :5.045   Max.   :5228
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
#choose data whose y=1
one<-filter(bank2,y==1)
```

```
## Warning: package 'bindrcpp' was built under R version 3.4.4
```

```
#choose data whose y=0
zero<-filter(bank2,y==0)
#Because data whose y=1 is much less than data whose y=0, to make both class of data have the same prop
smp_size <- floor(0.67 * nrow(one))
test_size <- floor(0.33 * nrow(one))
set.seed(100)
train_ind1 <- sample(seq_len(nrow(one)), size = smp_size)
train_ind0 <- sample(seq_len(nrow(zero)),size = smp_size)
#produce testing data whose labbel is y=0
test_ind <- sample(seq_len(nrow(zero[-train_ind0,])), size = test_size)
#combine data from different classes to generate training dataset and testing dataset
train<-rbind(one[train_ind1,],zero[train_ind0,])
test<-rbind(one[-train_ind1,],zero[test_ind,])
```

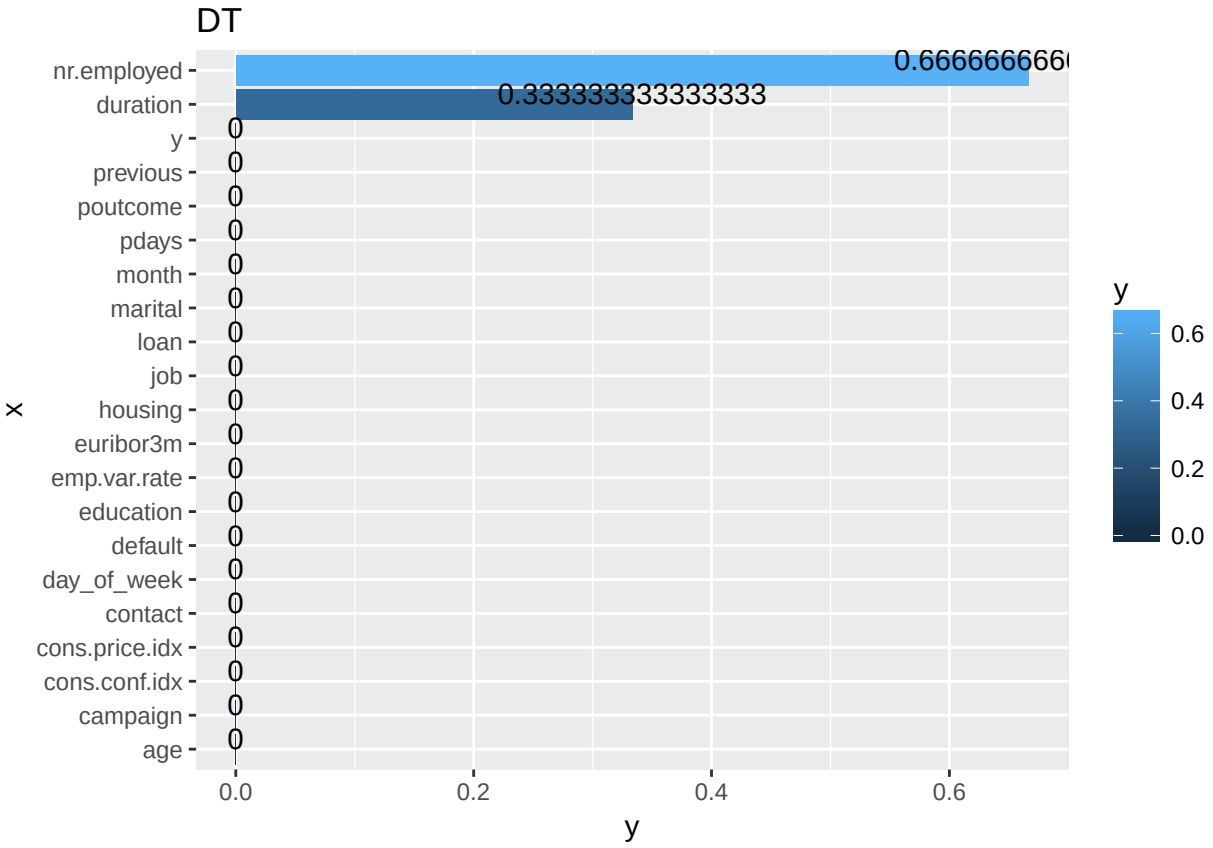
```
library(rminer)
#train 4 different models with normalized inputs and set the type as classification
M1=fit(y~.,train,model="dt",scale="inputs",task="c")
M2=fit(y~.,train,model="ksvm",scale="inputs",task="c")
```

```
## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.
```

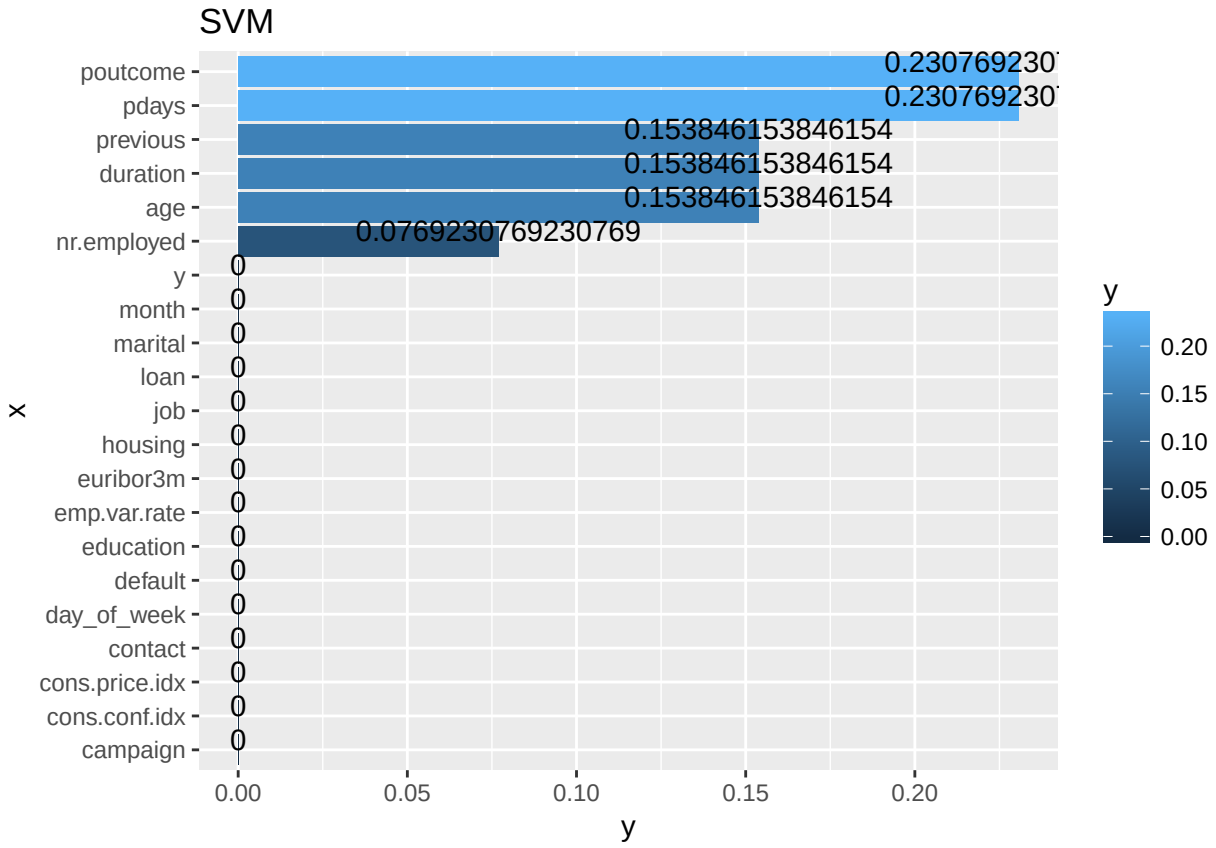
```
M3=fit(y~.,train,model="naiveBayes",scale="inputs",task="c")
M4=fit(y~.,train,model="lr",scale="inputs",task="c")
#output the importance of each attribute basing on each model
```

```
library(ggplot2)
nb.imp <- Importance(M1,train,task='c')
dt.imp <- Importance(M2,train,task='c')
svm.imp <- Importance(M3,train,task='c')
lr.imp <- Importance(M4,train,task='c')
```

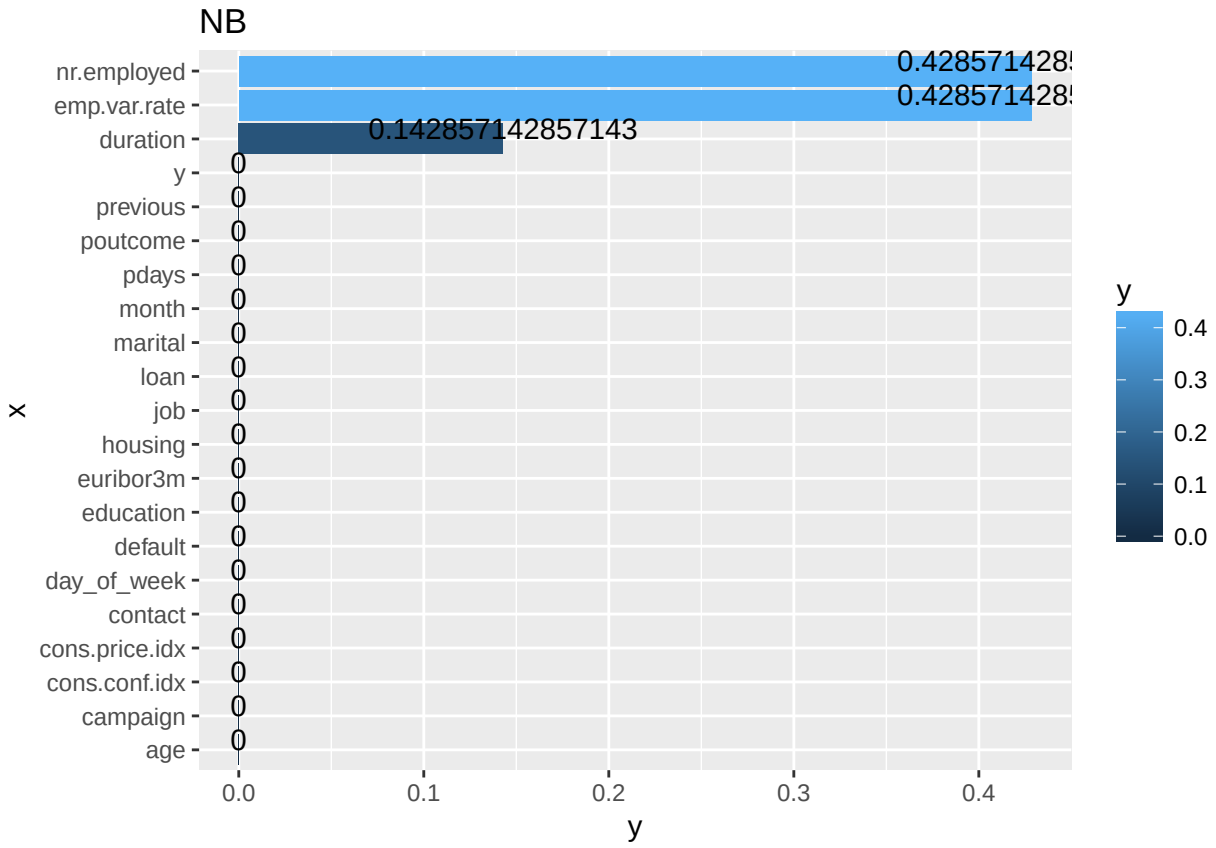
```
x<-names(train)

#attribute importance on DT
y<-dt.imp$imp
df <- data.frame(x= x, y = y)
ggplot(data = df, mapping = aes(x = reorder(x, y), y =  y,fill= y),main='') + labs(title="DT") +
        geom_bar(stat= 'identity')+
        geom_text(label=y,colour = "black", vjust=00)+coord_flip()+
        labs(x="x",y="y")
```



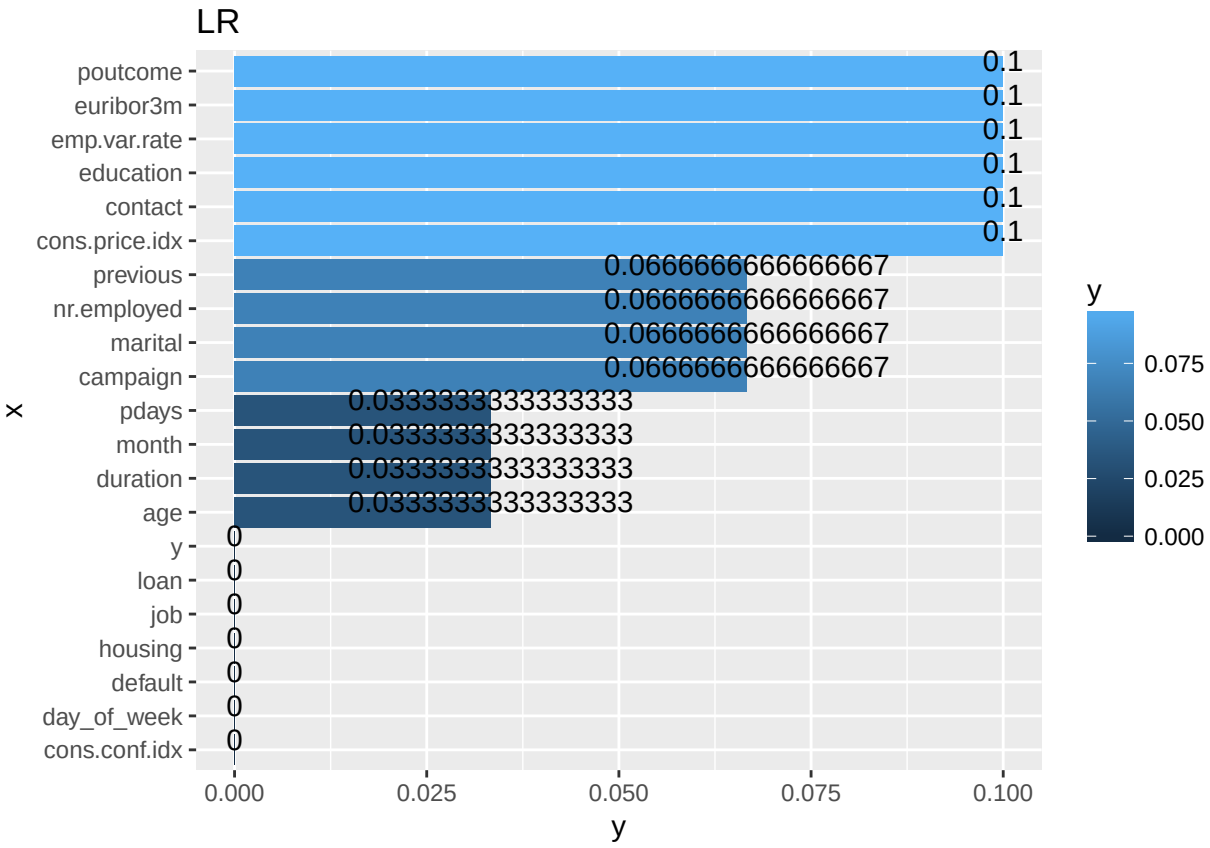

```
#attribute importance on SVM
y<-svm.imp$imp
df <- data.frame(x= x, y = y)
ggplot(data = df, mapping = aes(x = reorder(x, y), y =  y,fill= y),main='') + labs(title="SVM")+
        geom_bar(stat= 'identity')+
        geom_text(label=y,colour = "black", vjust=00)+coord_flip()+
        labs(x="x",y="y")
```




```
#attribute importance on NB
y<-nb.imp$imp
df <- data.frame(x= x, y = y)
ggplot(data = df, mapping = aes(x = reorder(x, y), y =  y,fill= y),main='') +labs(title="NB")+
        geom_bar(stat= 'identity')+
        geom_text(label=y,colour = "black", vjust=00)+coord_flip()+
        labs(x="x",y="y")
```




```
#attribute importance on LR
y<-lr.imp$imp
df <- data.frame(x= x, y = y)
ggplot(data = df, mapping = aes(x = reorder(x, y), y =  y,fill= y),main='') + labs(title="LR")+
        geom_bar(stat= 'identity')+
        geom_text(label=y,colour = "black", vjust=00)+coord_flip()+
        labs(x="x",y="y")
```




```
library(rminer)
print("DT")
```

```
## [1] "DT"
```

```
#Do prediction given testing data
P1=predict(M1,test)
#show accuracy of DT
print(mmetric(test$y,P1,"ACC"))
```

```
## [1] 89.16095
```

```
#Show confusion matrix of DT
print(mmetric(test$y,P1,"CONF"))
```

```
## $res
## NULL
##
## $conf
##        pred
## target    0    1
##      0 1366  165
##      1  167 1365
##
## $roc
## NULL
##
## $lift
## NULL
```

```r
print("SVM")
```

```
## [1] "SVM"
```

```r
P2=predict(M2,test)
print(mmetric(test$y,P2,"ACC"))
```

```
## [1] 87.8224
```

```r
print(mmetric(test$y,P2,"CONF"))
```

```
## $res
## NULL
##
## $conf
##       pred
## target    0    1
##      0 1319  212
##      1  161 1371
##
## $roc
## NULL
##
## $lift
## NULL
```

```r
print("NB")
```

```
## [1] "NB"
```

```r
P3=predict(M3,test)
print(mmetric(test$y,P3,"ACC"))
```

```
## [1] 79.98694
```

```r
print(mmetric(test$y,P3,"CONF"))
```

```
## $res
## NULL
##
## $conf
##       pred
## target    0    1
##      0 1379  152
##      1  461 1071
##
## $roc
## NULL
##
## $lift
## NULL
```

```r
print("LR")
```

```
## [1] "LR"
```

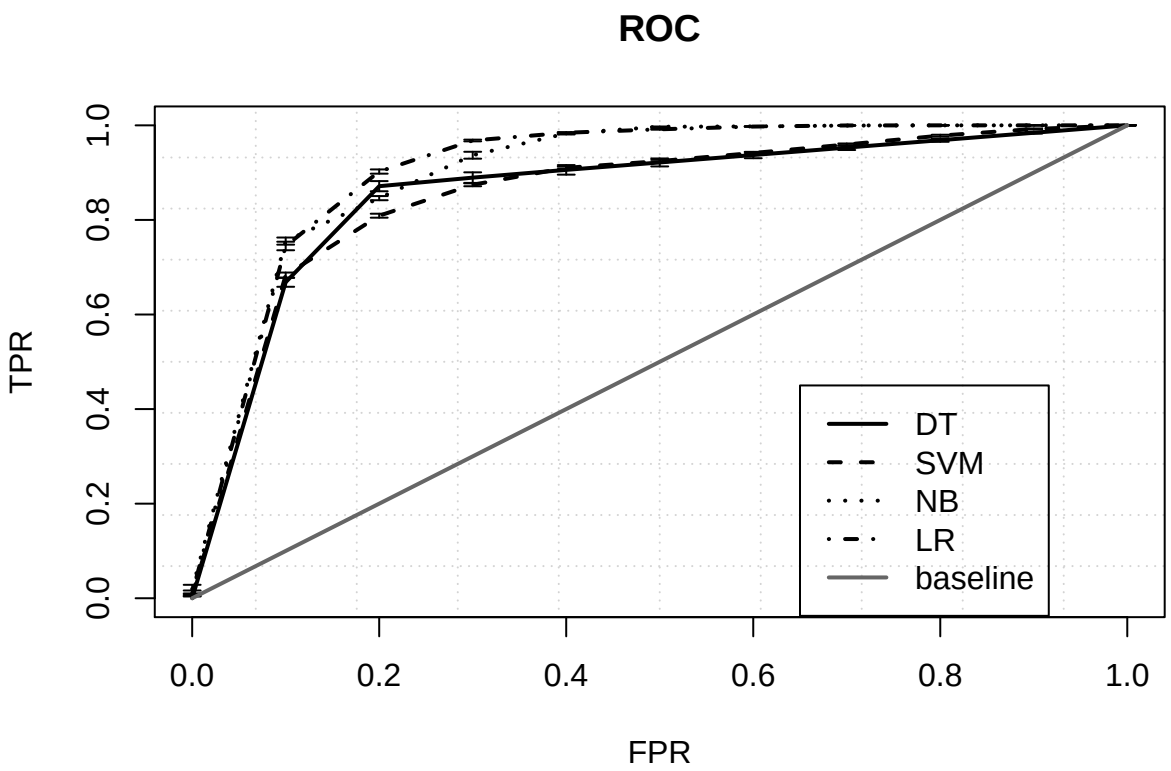
```r
P4=predict(M4,test)
print(mmetric(test$y,P4,"ACC"))
```

```
## [1] 88.08358
```

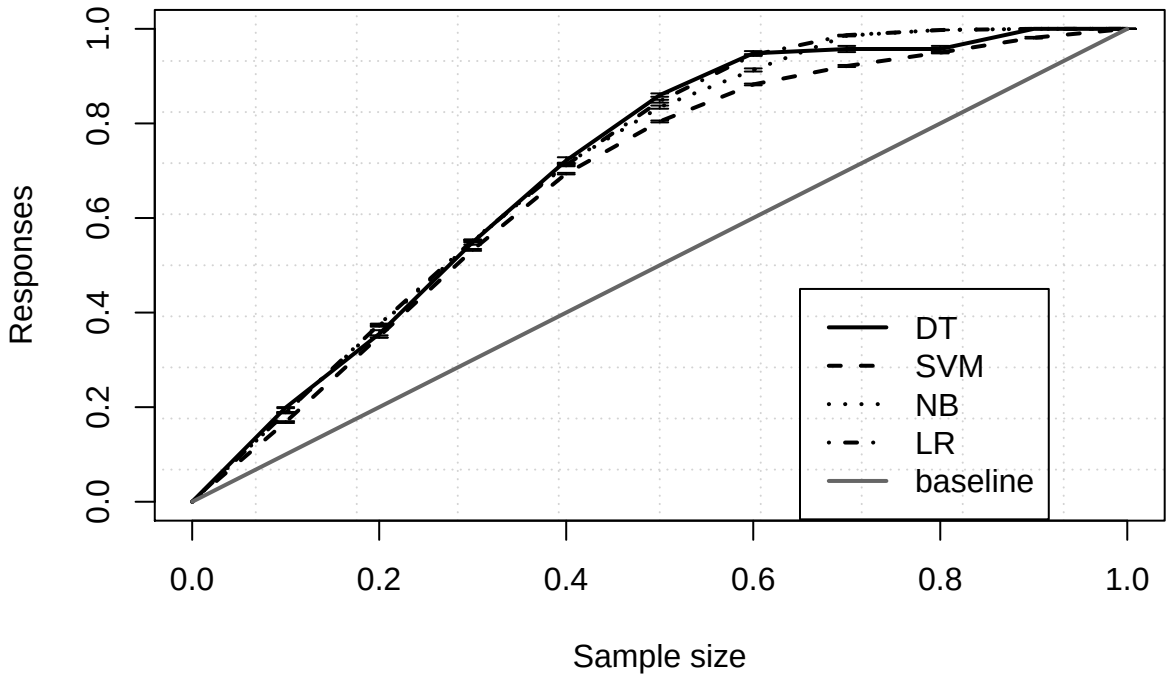
```
print(mmetric(test$y,P4,"CONF"))
```

```
## $res
## NULL
##
## $conf
##       pred
## target    0    1
##      0 1386  145
##      1  220 1312
##
## $roc
## NULL
##
## $lift
## NULL
```

```
#combine training data and testing data together to get a larger dataset
bank3=rbind(train,test)
#select attributes with largest importance
col_dt=c('duration','nr.employed','y')
#filter data to get the attributes we care
bank_dt=bank3[,col_dt]
#training and testing model with 2/3 of data holdout for testing, the same process will be repeated 20
M1_=mining(y~.,bank_dt,method=c("holdout",2/3),model="dt",Runs=20)

col_svm=c('poutcome','pdays','previous','duration','age','y')
bank_svm=bank3[,col_svm]
M2_=mining(y~.,bank_svm,method=c("holdout",2/3),model="ksvm",Runs=20)

col_nb=c('nr.employed','emp.var.rate','duration','y')
bank_nb=bank3[,col_nb]
M3_=mining(y~.,bank_nb,method=c("holdout",2/3),model="naiveBayes",Runs=20)

col_lr=c('poutcome','euribor3m','emp.var.rate','education','contact','cons.price.idx','previous','nr.em
bank_lr=bank3[,col_lr]
M4_=mining(y~.,bank_dt,method=c("holdout",2/3),model="lr",Runs=20)

#compine results together and plot ROC curve and LIFT curve
L=vector("list",4); L[[1]]=M1_; L[[2]]=M2_;L[[3]]=M3_; L[[4]]=M4_;
mgraph(L,graph="ROC",leg=c("DT","SVM","NB","LR"),baseline=TRUE,Grid=10,
       main="ROC")
```




```
mgraph(L,graph="LIFT",leg=c("DT","SVM","NB","LR"),
       baseline=TRUE,Grid=10,main="LIFT")
```




```
#return the total time consumed on the trainng and testing for 20 runs
print('DT')
```

```
## [1] "DT"
```

```
print (sum(M1_$time))
```

```
## [1] 1.786
```

```
print('SVM')
```

```
## [1] "SVM"
```

```
print (sum(M2_$time))
```

```
## [1] 70.374
```

```
print('NB')
```

```
## [1] "NB"
```

```
print (sum(M3_$time))
```

```
## [1] 8.078
```

```
print('LR')
```

```
## [1] "LR"
```

```
print (sum(M4_$time))
```

```
## [1] 1.831
```

```
#show AUC and ALIFT values for each model, the results are the average value of 20 runs
print('DT')
```

```
## [1] "DT"
```

```
mmetric(M1_,metric="AUC",aggregate="mean")
```

```
## [1] 0.8716057
```

```
mmetric(M1_,metric="ALIFT",aggregate="mean")
```

```
## [1] 0.7083969
```

```
print('SVM')
```

```
## [1] "SVM"
```

```
mmetric(M2_,metric="AUC",aggregate="mean")
```

```
## [1] 0.8601067
```

```
mmetric(M2_,metric="ALIFT",aggregate="mean")
```

```
## [1] 0.6799777
```

```
print('NB')
```

```
## [1] "NB"
```

```
mmetric(M3_,metric="AUC",aggregate="mean")
```

```
## [1] 0.9149248
```

```
mmetric(M3_,metric="ALIFT",aggregate="mean")
```

```
## [1] 0.7073801
```

```
print('LR')
```

```
## [1] "LR"
```

```
mmetric(M4_,metric="AUC",aggregate="mean")
```

```
## [1] 0.9237938
```

```
mmetric(M4_,metric="ALIFT",aggregate="mean")
```

```
## [1] 0.7118326
```

```
names(bank)
```

```
##  [1] "age"            "job"            "marital"        "education"
##  [5] "default"        "housing"        "loan"           "contact"
##  [9] "month"          "day_of_week"    "duration"       "campaign"
## [13] "pdays"          "previous"       "poutcome"       "emp.var.rate"
## [17] "cons.price.idx" "cons.conf.idx"  "euribor3m"      "nr.employed"
## [21] "y"
```