Lab15 - Random Forest, Bagging e Boosting

Machine Learning usando o R - Análise Macro

Thalles Quinaglia Liduares

2022-09-08

Upload pacotes

```
library(randomForest)
library(MASS)
```

Upload database

```
data<-Boston
```

Divisão da amostra entre treino e teste

```
set.seed(0809)

train = sample(1:nrow(Boston), nrow(Boston)/2)
```

Modelo RandomForest

```
set.seed(080922)

bag.boston = randomForest(medv~.,data,
             subset=train,mtry=13,importance=TRUE)

bag.boston
```

```
##
## Call:
##  randomForest(formula = medv ~ ., data = data, mtry = 13, importance = TRUE,      subset =
train)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 13
##
##           Mean of squared residuals: 14.00253
##                     % Var explained: 81.89
```

Analise preditiva do modelo

```
yhat.bag = predict(bag.boston,newdata=Boston[-train,])
boston.test=Boston[-train,"medv"]
```

EQM

```
mean((yhat.bag-boston.test)^2)
```

```
## [1] 10.76047
```

Alteração do modelo para `ntree` =30

```
set.seed(30)

bag.boston=randomForest(medv~.,data=Boston,subset=train,mtry=13,ntree=30)
yhat.bag = predict(bag.boston,newdata=Boston[-train,])
mean((yhat.bag-boston.test)^2)
```

```
## [1] 11.53845
```

Com `ntree` =30 o EQM cai de 22.99 para 11.53

Reestimando modelo com numero menor de preditores

```
set.seed(30)

rf.boston = randomForest(medv~.,data=Boston,
          subset=train,mtry=7,importance=TRUE)

yhat.rf = predict(rf.boston,newdata=Boston[-train,])

mean((yhat.rf-boston.test)^2)
```

```
## [1] 10.16317
```

O EQM cai de 19.62 para 10.16.

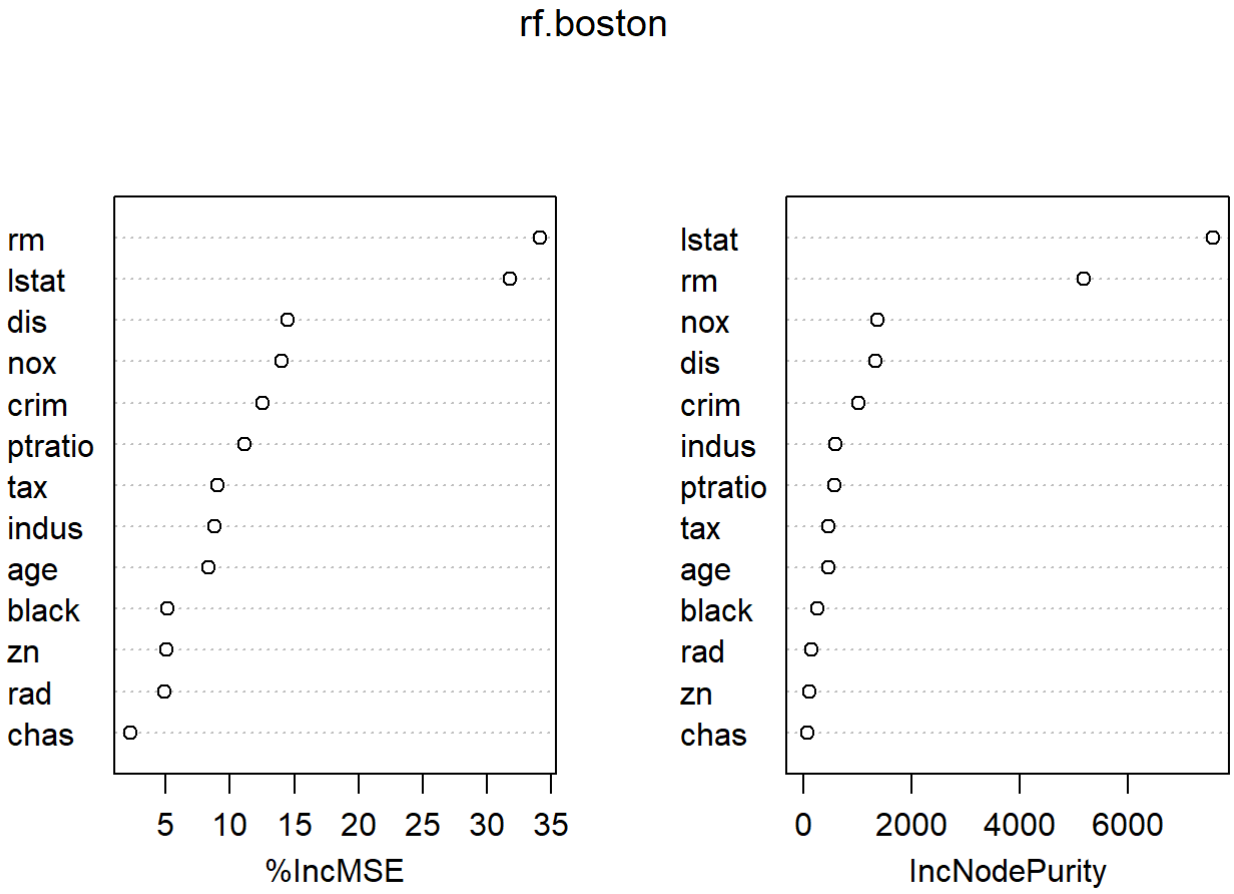
Calculo da importancia de cada umas das variaveis

```
importance(rf.boston)
```

```
##           %IncMSE IncNodePurity
## crim    12.552352    1019.99185
## zn       5.080236     116.73781
## indus    8.786244     592.85668
## chas     2.234996      81.27452
## nox     14.040565    1366.84897
## rm      34.118830    5187.14610
## age      8.319937     460.72895
## dis     14.474688    1329.44027
## rad      4.886991     152.86806
## tax      9.069219     474.31526
## ptratio 11.140126     584.53621
## black    5.110307     253.03133
## lstat   31.782551    7571.48218
```

Impureza do nó, medido pelo RSS

```
varImpPlot(rf.boston)
```

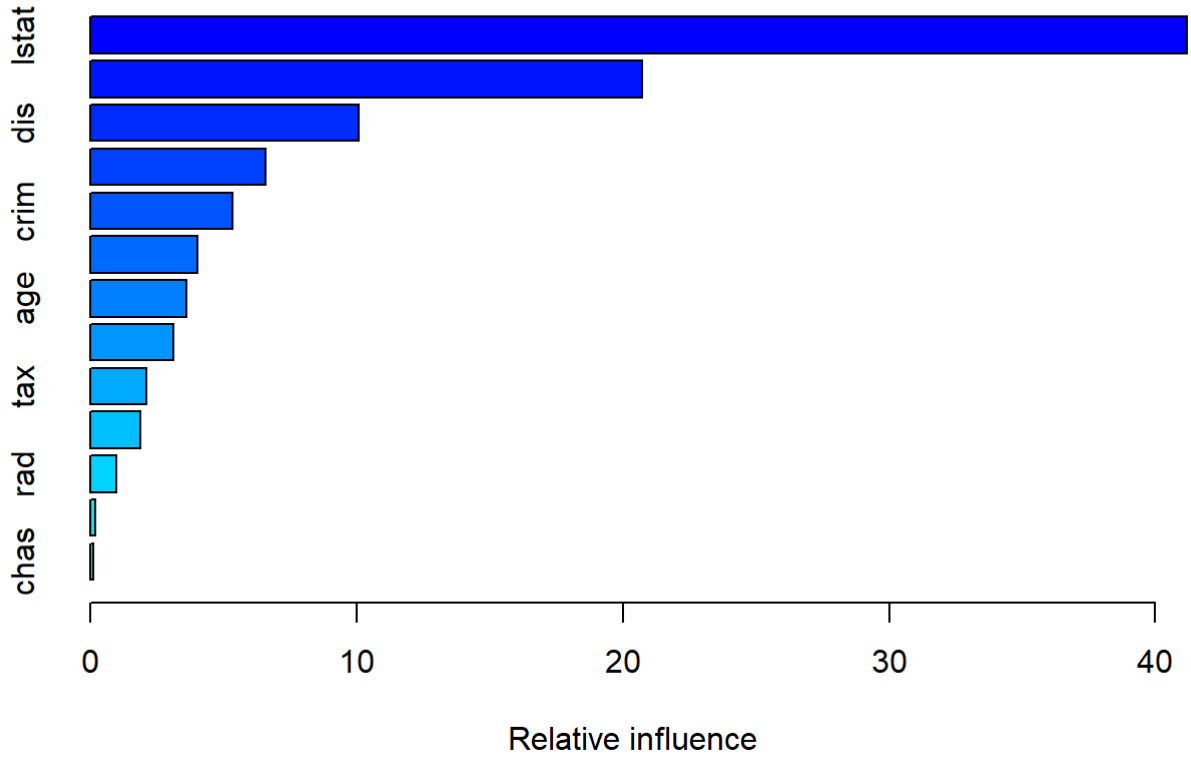


Boosting

```
library(gbm)
```

```
## Loaded gbm 2.1.8.1
```

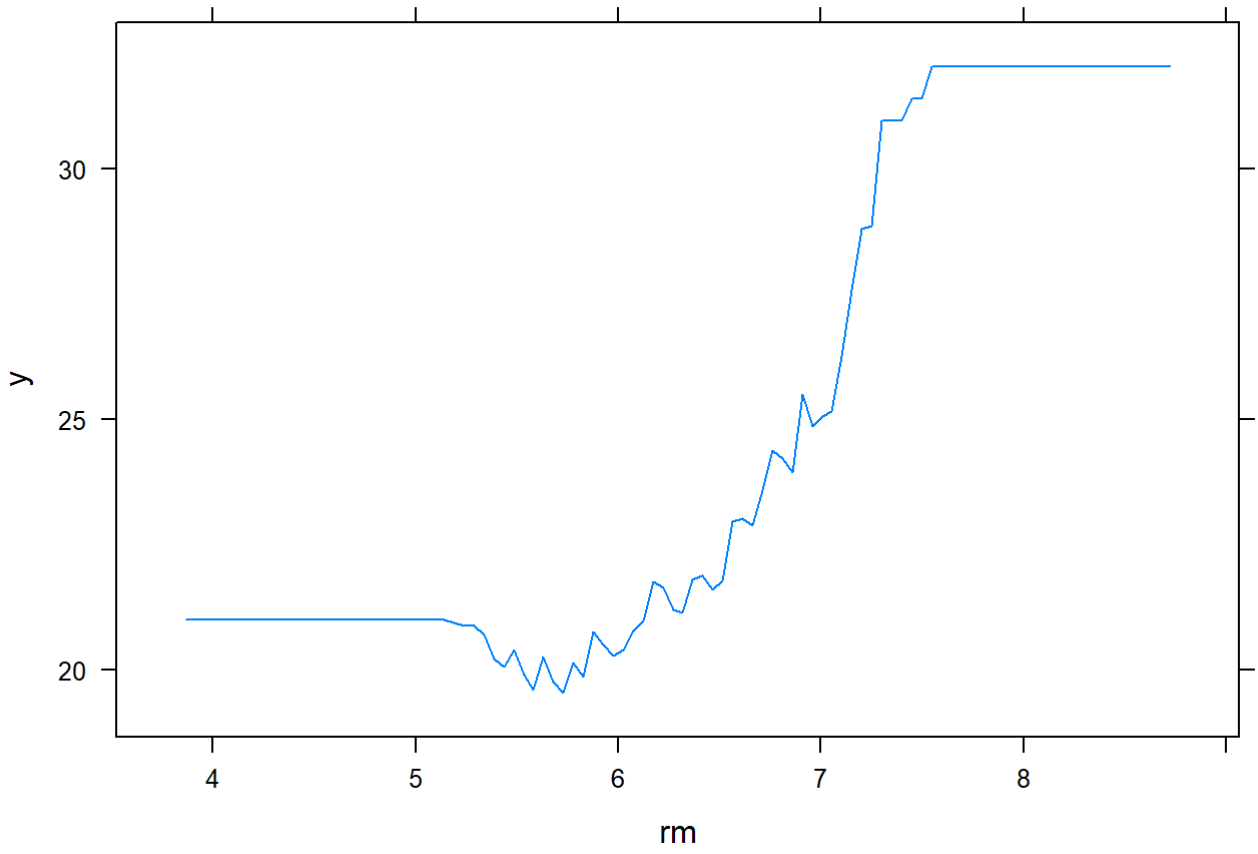
```
set.seed(111)

boost.boston = gbm(medv~.,data=Boston[train,],
             distribution="gaussian",n.trees=5200,interaction.depth=4) # 5200 arvores

summary(boost.boston)
```



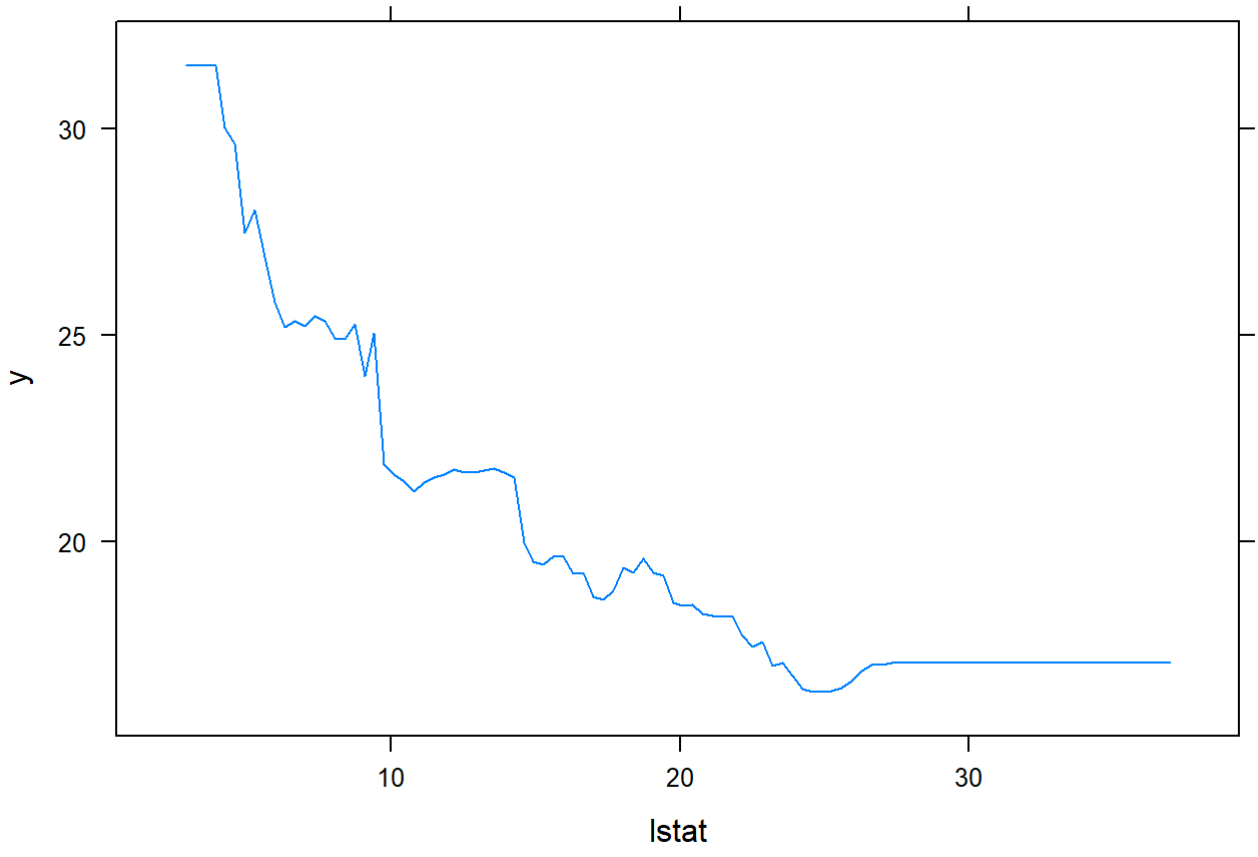

```
##             var    rel.inf
## lstat     lstat 41.1814878
## rm           rm 20.7370059
## dis         dis 10.0847910
## nox         nox  6.5905225
## crim       crim  5.3405561
## black     black  4.0139706
## age         age  3.6221691
## ptratio ptratio  3.1407748
## tax         tax  2.1102752
## indus     indus  1.8949965
## rad         rad  0.9803291
## zn           zn  0.1737322
## chas       chas  0.1293892
```

Efeito marginal das variaveis do modelo

```
par(mfrow=c(1,2))
plot(boost.boston,i="rm")
```




```
plot(boost.boston,i="lstat")
```



Analise preditiva do modelo e EQM

```
yhat.boost=predict(boost.boston,newdata=Boston[-train,],n.trees=5200)
mean((yhat.boost-boston.test)^2)
```

```
## [1] 13.22411
```

Com `ntrees` =5200 o EQM cai de 18.84 para 13.22.

Parâmetro de encolhimento *shrinkage*

```
boost.boston = gbm(medv~.,data=Boston[train,],
          distribution="gaussian",n.trees=5200,
          interaction.depth=4,shrinkage=0.3,verbose=F)

yhat.boost = predict(boost.boston,newdata=Boston[-train,],n.trees=5200)

mean((yhat.boost-boston.test)^2)
```

```
## [1] 14.86439
```