

Aprendizado de Máquinas

Métodos de Re-amostragem

Douglas Rodrigues

Universidade Federal Fluminense

- Vamos apresentar métodos de re-amostragem que são utilizados em três circunstâncias distintas:
 - 1 Reduzir a variância do preditor no treinamento.
 - 2 Realizar aprendizado de máquinas com amostras pequenas, onde não é possível fazer a separação das amostras treino/teste.
 - 3 Realizar comparações entre modelos (entre métodos distintos ou diferentes hiperparâmetros), tentando reduzir a dependência da amostra.
- Há diversos métodos, mas vamos nos concentrar em 3 métodos: bootstrap, k-fold e repeated k-fold.

Separando amostras Treino/Teste

```
> library(caret)
> library(kernlab)

> data(spam)
> set.seed(100)
> inTrain <- createDataPartition(y=spam$type,p=0.80,list=F)

#Separamos linhas para amostras treino e teste.
> training <- spam[inTrain,]
> testing <- spam[-inTrain,]
```

- É o método *default* do comando `train()`. Consiste em realizar amostragem dos dados de treino, com repetições.
- O padrão do comando `train()` é realizar 25 re-amostragens por bootstrap. Mas podemos alterar, através do comando `trainControl()`

```
> set.seed(100)
> ctrl <- trainControl(method = "boot", number = 10)
> model_boot <- train(type~. , data=training, method="glm",
trControl=ctrl)
> model_boot
```
- Para obter a precisão em cada uma das amostragens,

```
> model_boot$resample
```

- Para realizar re-amostragens por *bootstrap* fora da função `train()`, realizamos o seguinte comando.

```
> set.seed(32323)  
> folds <- createResample(y=training$type, times=10, list=F)
```
- Podemos obter as re-amostragens como uma lista, trocando o argumento final para `list=T`.

k-fold

- O método *k-fold* consiste em fatiar os dados em k -pedaços, separando um para teste e os demais para treino. Realizamos essa iteração k -vezes, até que todos pedaços tenham sido utilizados.



- Vamos aplicar o método *k-fold*, para $k = 10$.

```
> ctrl <- trainControl(method = "cv", number = 10)
> model_kfold <- train(type~. , data=training, method="glm",
trControl=ctrl)
```

- Para obter a precisão em cada uma das amostragens,
> model_kfold\$resample
- Observação: quando k é igual ao tamanho amostral, o método é também conhecido com *leave-one-out*.

- Para realizar re-amostragens por *k-fold* fora da função `train()`, realizamos o seguinte comando.

```
set.seed(100)
> folds_training<-createFolds(y=training$type,k=10,list=T,
returnTrain = T)
> summary(folds_training)
> training_Fold01<-training[folds_training$Fold01,]
> testing_Fold01<-training[-folds_training$Fold01,]
```


- *Repeated k-fold* nada mais é que repetir o método *k-fold* diversas vezes.
- Por exemplo, se queremos aplicar um método de treino 3 vezes em 10-folds,

```
> ctrl <- trainControl(method= "repeatedcv", number=10,  
  repeats=3)  
> model <- train(type ~., data=training, method="glm",  
  trControl=ctrl)
```

- Vamos realizar a avaliação dos classificadores criados.
 - > `pred_boot<-predict(model_boot,testing)`
 - > `confusionMatrix(pred_boot,testing$type)`

- Vamos realizar a avaliação dos classificadores criados.
 - > `pred_boot<-predict(model_boot,testing)`
 - > `confusionMatrix(pred_boot,testing$type)`
- Faça o mesmo para `model_kfold` e `model_rkfold`