Model Evaluation - Resampling

source("~/MCD/MCD/m.R")  
  
library(mlbench)  
data(PimaIndiansDiabetes)  
  
print("Loaded")

## [1] "Loaded"

## k-fold Cross-Validation

The k-fold cross-validation method involves splitting the dataset into k-subsets. Each subset is held-out while the model is trained on all other subsets. This process is repeated until accuracy is determined for each instance in the dataset, and an overall accuracy estimate is provided.

A variant of this method is to select k partitions in a way that makes the folds balanced with respect to the outcome.

It is a robust method for estimating accuracy.

Typically the values for k are 5 or 10. As k gets larger, the difference in size between the training set and the resampling subsets gets smaller. As this difference decreases, the bias of the technique gets smaller. In this context, the bias is the difference betwen the estimated and true values of performance.

K-fold cross validation has a high variance compared to other methods. However this is less of a problem for large data sets.

Also larger values of k are more computationally expensive.

Its been found that leave-out-one and k=10 cv yielded similar results.

Small values of k (2 ,3) have high bias but are very computationally efficient. The bias in this case is similar to bootstrap but with much larger variance.

Repeating k-fold cv can be used to effectively increase the precision while still maintaining a small bias.

The following example uses 10-fold cross-validation to estimate the accuracy of the Naive Bayes algorithm on the iris dataset.

m <- m\_raw(iris) %>% m\_response(~ Species) %>% m\_cross\_validation(number = 10) %>% m\_naive\_bayes() %>% m\_accuracy\_metric() %>% m\_train() %>% m\_run()

## [1] "iris"

print(m[[1]]$train$result)

## Naive Bayes   
##   
## 150 samples  
## 4 predictor  
## 3 classes: 'setosa', 'versicolor', 'virginica'   
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 135, 135, 135, 135, 135, 135, ...   
## Resampling results across tuning parameters:  
##   
## usekernel Accuracy Kappa  
## FALSE 0.9533333 0.93   
## TRUE 0.9600000 0.94   
##   
## Tuning parameter 'fL' was held constant at a value of 0  
## Tuning  
## parameter 'adjust' was held constant at a value of 1  
## Accuracy was used to select the optimal model using the largest value.  
## The final values used for the model were fL = 0, usekernel = TRUE  
## and adjust = 1.

Example of Repeated k-fold Cross-Validation

m <- m\_raw(iris) %>% m\_response(~ Species) %>% m\_repeated\_cross\_validation(number = 10, repeats = 4) %>% m\_naive\_bayes() %>% m\_accuracy\_metric() %>% m\_train() %>% m\_run()

## [1] "iris"

print(m[[1]]$train$result)

## Naive Bayes   
##   
## 150 samples  
## 4 predictor  
## 3 classes: 'setosa', 'versicolor', 'virginica'   
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold, repeated 3 times)   
## Summary of sample sizes: 135, 135, 135, 135, 135, 135, ...   
## Resampling results across tuning parameters:  
##   
## usekernel Accuracy Kappa  
## FALSE 0.9533333 0.93   
## TRUE 0.9600000 0.94   
##   
## Tuning parameter 'fL' was held constant at a value of 0  
## Tuning  
## parameter 'adjust' was held constant at a value of 1  
## Accuracy was used to select the optimal model using the largest value.  
## The final values used for the model were fL = 0, usekernel = TRUE  
## and adjust = 1.

## Generalized Cross Validation

For linear regression models, there is a formula for approximating the leave-out-one error rate. The generalized cross-valdation (GCV) statistic. (APM - Section 4.4)

## Repeated Training/Test Splits

Repeated training/set splits is also known as "leave-group-out cross validation" or "Monto Carlo cross-validation".

The modeller controls how much data goes into each split along with the number of repetitions.

The bias decreases as the amount of data in the test set approaches the amount in the modelling set.

A good rule of thumb is 75-80%.

Higher proportions are a good idea if the number of repetitions is lage.

The number of repetions is important. incresing the number of subsets has the effect of decreasing the uncertainty of the performance estimates.

For a gross model estimate 25 repetitions will be adequete, but their will be instability in the estimate. To get a stable estimate then 50-200 repeats is suggested.

The higher the proportion of data allocated to the training set the higher the number of repetitions that are needed.

## Bootstrap

Bootstrap resampling involves taking random samples from the dataset (with re-selection) against which to evaluate the model. In aggregate, the results provide an indication of the variance of the model’s performance. Typically, large number of resampling iterations are performed (thousands or tens of thousands). The following example uses a bootstrap with 100 resamples to estimate the accuracy of a Naive Bayes model.

In general, bootstrap error rates tend to have less uncertainty than k-fold cross-validation. However this technique has similar bias to k-fold cv k~2. But will decrease as the training set sample size becomes larger.

The 632 mthod attempts to correct for this bias. However it can be unstable with small sample sizes. (APM - Section 4.4)

m <- m\_raw(iris) %>% m\_response(~ Species) %>% m\_bootstrap\_validation(number = 100) %>% m\_naive\_bayes() %>% m\_accuracy\_metric() %>% m\_train() %>% m\_run()

## [1] "iris"

print(m[[1]]$train$result)

## Naive Bayes   
##   
## 150 samples  
## 4 predictor  
## 3 classes: 'setosa', 'versicolor', 'virginica'   
##   
## No pre-processing  
## Resampling: Bootstrapped (100 reps)   
## Summary of sample sizes: 150, 150, 150, 150, 150, 150, ...   
## Resampling results across tuning parameters:  
##   
## usekernel Accuracy Kappa   
## FALSE 0.9533094 0.9293741  
## TRUE 0.9543506 0.9309620  
##   
## Tuning parameter 'fL' was held constant at a value of 0  
## Tuning  
## parameter 'adjust' was held constant at a value of 1  
## Accuracy was used to select the optimal model using the largest value.  
## The final values used for the model were fL = 0, usekernel = TRUE  
## and adjust = 1.

Running this example, you will see the estimated accuracy of the Naive Bayes model with two diﬀerent values for the usekernel model parameter.

## Leave One Out Cross-Validation

In Leave One Out Cross-Validation (LOOCV), a data instance is left out and a model constructed on all other data instances in the training set. This is repeated for all data instances. The following example demonstrates LOOCV to estimate the accuracy of the Naive Bayes algorithm on the iris dataset.

m <- m\_raw(iris) %>% m\_response(~ Species) %>% m\_loocv\_validation() %>% m\_naive\_bayes() %>% m\_accuracy\_metric() %>% m\_train() %>% m\_run()

## [1] "iris"

## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info =  
## trainInfo, : There were missing values in resampled performance measures.

print(m[[1]]$train$result)

## Naive Bayes   
##   
## 150 samples  
## 4 predictor  
## 3 classes: 'setosa', 'versicolor', 'virginica'   
##   
## No pre-processing  
## Resampling: Leave-One-Out Cross-Validation   
## Summary of sample sizes: 149, 149, 149, 149, 149, 149, ...   
## Resampling results across tuning parameters:  
##   
## usekernel Accuracy Kappa  
## FALSE 0.9533333 0   
## TRUE 0.9600000 0   
##   
## Tuning parameter 'fL' was held constant at a value of 0  
## Tuning  
## parameter 'adjust' was held constant at a value of 1  
## Accuracy was used to select the optimal model using the largest value.  
## The final values used for the model were fL = 0, usekernel = TRUE  
## and adjust = 1.