CARET_Lab

2023-12-06

Package loading

```
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

Load Data

```
# attach the iris dataset to the environment.
data(iris)
# rename the dataset
dataset <- iris
```

Task1: Create a Validation/Training Dataset You need to split the loaded dataset into two, 80% of which we will use to train our models and 20% that we will hold back as a validation dataset. Hint: use createDataPartition function

```
trainRowNumbers <- createDataPartition(dataset$Species, p=0.8, list=FALSE)
trainData <- dataset[trainRowNumbers,]
testData <- dataset[-trainRowNumbers,]
```

Task2: Summarize Dataset Use skimr library to summarize the dataset

```
library(skimr)
skimmed <- skim_to_wide(trainData)
```

```
## Warning: 'skim_to_wide' is deprecated.
## Use 'skim()' instead.
## See help("Deprecated")
```

```
skimmed
```

Data summary

Name	Piped data
Number of rows	120
Number of columns	5
_____	
Column type frequency:	
factor	1
numeric	4
_____	
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
Species	0	1	FALSE	3	set: 40, ver: 40, vir: 40

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
Sepal.Length	0	1	5.82	0.82	4.3	5.1	5.70	6.4	7.7	▆▇▅▂▁
Sepal.Width	0	1	3.06	0.41	2.2	2.8	3.00	3.3	4.2	▂▆▇▂▁
Petal.Length	0	1	3.77	1.76	1.0	1.6	4.35	5.1	6.9	▇▁▃▆▃
Petal.Width	0	1	1.21	0.78	0.1	0.3	1.30	1.8	2.5	▇▁▂▅▂

Task3: split input and output It is the time to seperate the input attributes and the output attributes. call the inputs attributes x and the output attribute (or class) y.

```
x= trainData[ ,1:4]
y= trainData[ ,5]
```

Task4: Train Control for Validation Test

We will use 10-fold crossvalidation to estimate accuracy.

```
# Run algorithms using 10-fold cross validation
control <- trainControl(method="cv", number=10)
metric <- "Accuracy"
```

Task5: Model Training Train 5 different algorithms using 'train' function:

- Linear Discriminant Analysis (LDA)
- Classification and Regression Trees (CART).
- k-Nearest Neighbors (kNN).
- Support Vector Machines (SVM) with a linear kernel.
- Random Forest (RF)

```
#train Model using LDA
set.seed(100)
#1
model_LDA = train (Species~., data = trainData, method = 'lda', trControl = control, metric=metric)
model_LDA
```

```
## Linear Discriminant Analysis
```

```
## Linear Discriminant Analysis
##
## 120 samples
##   4 predictor
##   3 classes: 'setosa', 'versicolor', 'virginica'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 108, 108, 108, 108, 108, 108, ...
## Resampling results:
##
##   Accuracy  Kappa
##   0.975     0.9625
```

```
#2
model_CART = train (Species~., data = trainData, method = 'rpart', trControl = control, metric=metric)
model_CART
```

```
## CART
##
## 120 samples
##   4 predictor
##   3 classes: 'setosa', 'versicolor', 'virginica'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 108, 108, 108, 108, 108, 108, ...
## Resampling results across tuning parameters:
##
##   cp    Accuracy   Kappa
##   0.00  0.9250000  0.8875
##   0.45  0.8333333  0.7500
##   0.50  0.3333333  0.0000
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.
```

```
#3
model_KNN = train (Species~., data = trainData, method = 'knn', trControl = control, metric=metric)
model_KNN
```

```
## k-Nearest Neighbors
##
## 120 samples
##   4 predictor
##   3 classes: 'setosa', 'versicolor', 'virginica'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 108, 108, 108, 108, 108, 108, ...
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
##   5  0.9500000  0.925
##   7  0.9666667  0.950
##   9  0.9666667  0.950
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 9.
```

```
#4
model_SVM = train (Species~., data = trainData, method = 'svmRadial', trControl = control, metric=metric)
model_SVM
```

```
## Support Vector Machines with Radial Basis Function Kernel
##
## 120 samples
##   4 predictor
##   3 classes: 'setosa', 'versicolor', 'virginica'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 108, 108, 108, 108, 108, 108, ...
## Resampling results across tuning parameters:
##
##   C     Accuracy   Kappa
##   0.25  0.9333333  0.9000
##   0.50  0.9333333  0.9000
##   1.00  0.9250000  0.8875
##
## Tuning parameter 'sigma' was held constant at a value of 1.31386
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were sigma = 1.31386 and C = 0.25.
```

```
#5
model_RF = train (Species~., data = trainData, method = 'rf', trControl = control, metric=metric)
model_RF
```

```
## Random Forest
##
## 120 samples
##   4 predictor
##   3 classes: 'setosa', 'versicolor', 'virginica'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 108, 108, 108, 108, 108, 108, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##   2     0.9333333  0.9000
##   3     0.9416667  0.9125
```

```
##   4     0.9416667  0.9125
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 3.
```

Task6: Select the Best Model We now have 5 models and accuracy estimations for each. We need to compare the models to each other and select the most accurate. Use resamples function to complete this task

```
model_compare <- resamples(list(LDA=model_LDA, CART= model_CART, KNN=model_CART, SVM=model_SVM, RF=model_RF))
summary(model_compare)
```

```
##
## Call:
## summary.resamples(object = model_compare)
##
## Models: LDA, CART, KNN, SVM, RF
## Number of resamples: 10
##
## Accuracy
##          Min.   1st Qu.    Median      Mean   3rd Qu. Max. NA's
## LDA  0.9166667 0.9375000 1.0000000 0.9750000 1.0000000    1    0
## CART 0.8333333 0.9166667 0.9166667 0.9250000 0.9166667    1    0
## KNN  0.8333333 0.9166667 0.9166667 0.9250000 0.9166667    1    0
## SVM  0.8333333 0.9166667 0.9166667 0.9333333 1.0000000    1    0
## RF   0.8333333 0.9166667 0.9166667 0.9416667 1.0000000    1    0
##
## Kappa
##       Min. 1st Qu. Median   Mean 3rd Qu. Max. NA's
## LDA  0.875 0.90625  1.000 0.9625   1.000    1    0
## CART 0.750 0.87500  0.875 0.8875   0.875    1    0
## KNN  0.750 0.87500  0.875 0.8875   0.875    1    0
## SVM  0.750 0.87500  0.875 0.9000   1.000    1    0
## RF   0.750 0.87500  0.875 0.9125   1.000    1    0
```

What was the most accurate model?

Task7: Make Prediction (Confusion Matrix) Now we want to get an idea of the accuracy of the best model on our validation set. Use 'predict' and confusionMatrix functions to complete this task.

```
#for lda
predicted = predict(model_LDA, testData)
confusionMatrix(reference = testData$Species, predicted)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         10          0         0
##   versicolor      0         10         0
##   virginica       0          0        10
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.8843, 1)
##     No Information Rate : 0.3333
##     P-Value [Acc > NIR] : 4.857e-15
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                 1.0000            1.0000           1.0000
## Specificity                 1.0000            1.0000           1.0000
## Pos Pred Value              1.0000            1.0000           1.0000
## Neg Pred Value              1.0000            1.0000           1.0000
## Prevalence                  0.3333            0.3333           0.3333
## Detection Rate              0.3333            0.3333           0.3333
## Detection Prevalence        0.3333            0.3333           0.3333
## Balanced Accuracy           1.0000            1.0000           1.0000
```

```
#for knn
predicted = predict(model_KNN, testData)
confusionMatrix(reference = testData$Species, predicted)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         10          0         0
##   versicolor      0         10         1
##   virginica       0          0         9
##
## Overall Statistics
##
##                Accuracy : 0.9667
##                  95% CI : (0.8278, 0.9992)
##     No Information Rate : 0.3333
##     P-Value [Acc > NIR] : 2.963e-13
##
##                   Kappa : 0.95
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                 1.0000            1.0000           0.9000
## Specificity                 1.0000            0.9500           1.0000
## Pos Pred Value              1.0000            0.9091           1.0000
## Neg Pred Value              1.0000            1.0000           0.9524
```

```
## Prevalence                0.3333           0.3333           0.3333
## Detection Rate            0.3333           0.3333           0.3000
## Detection Prevalence      0.3333           0.3667           0.3000
## Balanced Accuracy         1.0000           0.9750           0.9500
```

```
#for CART
predicted = predict(model_CART, testData)
confusionMatrix(reference = testData$Species, predicted)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         10          0         0
##   versicolor      0          9         1
##   virginica       0          1         9
##
## Overall Statistics
##
##                Accuracy : 0.9333
##                  95% CI : (0.7793, 0.9918)
##     No Information Rate : 0.3333
##     P-Value [Acc > NIR] : 8.747e-12
##
##                   Kappa : 0.9
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                 1.0000            0.9000           0.9000
## Specificity                 1.0000            0.9500           0.9500
## Pos Pred Value              1.0000            0.9000           0.9000
## Neg Pred Value              1.0000            0.9500           0.9500
## Prevalence                  0.3333            0.3333           0.3333
## Detection Rate              0.3333            0.3000           0.3000
## Detection Prevalence        0.3333            0.3333           0.3333
## Balanced Accuracy           1.0000            0.9250           0.9250
```

```
#for SVM
predicted = predict(model_SVM, testData)
confusionMatrix(reference = testData$Species, predicted)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa          9          0         0
##   versicolor      0          8         0
##   virginica       1          2        10
##
## Overall Statistics
##
##                Accuracy : 0.9
##                  95% CI : (0.7347, 0.9789)
##     No Information Rate : 0.3333
##     P-Value [Acc > NIR] : 1.665e-10
##
##                   Kappa : 0.85
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                 0.9000            0.8000           1.0000
## Specificity                 1.0000            1.0000           0.8500
## Pos Pred Value              1.0000            1.0000           0.7692
## Neg Pred Value              0.9524            0.9091           1.0000
## Prevalence                  0.3333            0.3333           0.3333
## Detection Rate              0.3000            0.2667           0.3333
## Detection Prevalence        0.3000            0.2667           0.4333
## Balanced Accuracy           0.9500            0.9000           0.9250
```

```
#for RF
predicted = predict(model_RF, testData)
confusionMatrix(reference = testData$Species, predicted)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         10          0         0
##   versicolor      0          9         0
##   virginica       0          1        10
##
## Overall Statistics
##
##                Accuracy : 0.9667
##                  95% CI : (0.8278, 0.9992)
##     No Information Rate : 0.3333
##     P-Value [Acc > NIR] : 2.963e-13
##
##                   Kappa : 0.95
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                 1.0000            0.9000           1.0000
## Specificity                 1.0000            1.0000           0.9500
```

```
## Pos Pred Value              1.0000           1.0000           0.9091
## Neg Pred Value              1.0000           0.9524           1.0000
## Prevalence                  0.3333           0.3333           0.3333
## Detection Rate              0.3333           0.3000           0.3333
## Detection Prevalence        0.3333           0.3000           0.3667
## Balanced Accuracy           1.0000           0.9500           0.9750
```