```
> library(caret)
> library(dslabs)
> data("mnist_27")
>
> names(mnist_27)
[1] "train"       "test"        "index_train" "index_test"  "true_p"
> dim(mnist_27$train)
[1] 800   3
>
> suppressWarnings(set.seed(1, sample.kind = "Rounding"))
> models <- c("glm", "lda", "naive_bayes", "svmLinear", "knn", "gamLoess", "multinom", "qda", "rf", "adabo
ost")
> fits <- lapply(models, function(model){
+ print(model)
+ train(y ~ ., method = model, data = mnist_27$train)
+ })
[1] "glm"
[1] "lda"
[1] "naive_bayes"
[1] "svmLinear"
[1] "knn"
[1] "gamLoess"
[1] "multinom"
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 384.794809
final  value 384.794775
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 421.251454
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 384.848555
final  value 384.848522
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 358.466023
final  value 358.466014
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 400.257332
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 358.528966
final  value 358.528958
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 345.361326
final  value 345.361319
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 389.162400
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 345.427631
final  value 345.427624
converged
```

```
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 370.819967
iter  10 value 370.819967
iter  10 value 370.819967
final  value 370.819967
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 411.520894
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 370.881269
iter  10 value 370.881269
iter  10 value 370.881269
final  value 370.881269
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 338.339240
final  value 337.642174
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 389.552735
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 337.725860
final  value 337.725851
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 362.651997
iter  10 value 362.651996
iter  10 value 362.651996
final  value 362.651996
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 404.947235
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 362.716896
iter  10 value 362.716895
iter  10 value 362.716894
final  value 362.716894
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 353.360649
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 396.615883
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 353.427369
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 331.505876
```

```
final  value 331.505837
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 382.233327
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 331.587049
final  value 331.587010
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 364.158073
iter  10 value 364.158073
iter  10 value 364.158073
final  value 364.158073
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 400.438283
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 364.210111
iter  10 value 364.210111
iter  10 value 364.210111
final  value 364.210111
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 343.760429
final  value 343.760410
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 387.083157
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 343.826126
final  value 343.826108
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 377.277862
iter  10 value 377.277862
iter  10 value 377.277861
final  value 377.277861
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 413.479657
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 377.330740
iter  10 value 377.330739
iter  10 value 377.330738
final  value 377.330738
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 363.527477
final  value 363.527449
```

```
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 405.904614
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 363.591426
final  value 363.591399
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 346.706756
iter  10 value 346.706754
iter  10 value 346.706754
final  value 346.706754
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 393.064300
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 346.778579
iter  10 value 346.778577
iter  10 value 346.778577
final  value 346.778577
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 350.308158
final  value 350.308124
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 394.686750
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 350.376208
final  value 350.376174
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 365.423988
final  value 365.423967
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 407.046095
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 365.486830
final  value 365.486809
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 375.942875
final  value 375.942868
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 412.738783
converged
```

```
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 375.996860
final  value 375.996853
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 369.004020
final  value 369.003531
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 407.374841
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 369.060934
final  value 369.060455
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 360.551961
iter  10 value 360.551959
iter  10 value 360.551959
final  value 360.551959
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 400.866217
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 360.611945
iter  10 value 360.611943
iter  10 value 360.611943
final  value 360.611943
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 370.467778
final  value 370.414135
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 406.680836
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 370.519928
final  value 370.466715
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 355.236387
final  value 355.236347
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 401.370189
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 355.308279
final  value 355.308240
converged
```

```
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 364.714111
final  value 364.714051
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 407.312950
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 364.779508
final  value 364.779448
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 347.812292
final  value 347.812150
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 389.764148
iter  10 value 389.764145
iter  10 value 389.764145
final  value 389.764145
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 347.875247
final  value 347.875105
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 319.870357
final  value 319.870338
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 372.994080
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 319.955663
final  value 319.955644
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 312.576095
final  value 312.576064
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 367.284329
iter  10 value 367.284329
iter  10 value 367.284329
final  value 367.284329
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 312.666550
final  value 312.666520
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 363.313712
```

```
iter  10 value 363.313712
iter  10 value 363.313712
final  value 363.313712
converged
# weights:  4 (3 variable)
initial  value 554.517744
final  value 403.175943
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 363.373575
iter  10 value 363.373575
iter  10 value 363.373575
final  value 363.373575
converged
# weights:  4 (3 variable)
initial  value 554.517744
iter  10 value 358.900453
iter  10 value 358.900452
iter  10 value 358.900452
final  value 358.900452
converged
[1] "qda"
[1] "rf"
note: only 1 unique complexity parameters in default grid. Truncating the grid to 1 .

[1] "adaboost"
There were 50 or more warnings (use warnings() to see the first 50)
> names(fits) <- models
>
> -------------------------------------------------------------------------------------------------
>
> class(fits)
[1] "list"
> fits
$glm
Generalized Linear Model

800 samples
  2 predictor
  2 classes: '2', '7'

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 800, 800, 800, 800, 800, 800, ...
Resampling results:

  Accuracy   Kappa
  0.8010386  0.5999105


$lda
Linear Discriminant Analysis

800 samples
  2 predictor
  2 classes: '2', '7'

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 800, 800, 800, 800, 800, 800, ...
Resampling results:

  Accuracy   Kappa
  0.7987611  0.5957857
```

```
$naive_bayes
Naive Bayes

800 samples
  2 predictor
  2 classes: '2', '7'

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 800, 800, 800, 800, 800, 800, ...
Resampling results across tuning parameters:

  usekernel  Accuracy   Kappa
  FALSE      0.8128524  0.6238659
   TRUE      0.8248359  0.6477626

Tuning parameter 'laplace' 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 laplace = 0, usekernel = TRUE
 and adjust = 1.

$svmLinear
Support Vector Machines with Linear Kernel

800 samples
  2 predictor
  2 classes: '2', '7'

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 800, 800, 800, 800, 800, 800, ...
Resampling results:

  Accuracy   Kappa
  0.7969828  0.5914144

Tuning parameter 'C' was held constant at a value of 1

$knn
k-Nearest Neighbors

800 samples
  2 predictor
  2 classes: '2', '7'

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 800, 800, 800, 800, 800, 800, ...
Resampling results across tuning parameters:

  k  Accuracy   Kappa
  5  0.8067598  0.6123030
  7  0.8176592  0.6342788
  9  0.8180268  0.6347711

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was k = 9.

$gamLoess
Generalized Additive Model using LOESS
```

```
800 samples
  2 predictor
  2 classes: '2', '7'

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 800, 800, 800, 800, 800, 800, ...
Resampling results:

  Accuracy   Kappa
  0.8455962  0.6900868

Tuning parameter 'span' was held constant at a value of 0.5
Tuning
 parameter 'degree' was held constant at a value of 1

$multinom
Penalized Multinomial Regression

800 samples
  2 predictor
  2 classes: '2', '7'

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 800, 800, 800, 800, 800, 800, ...
Resampling results across tuning parameters:

  decay  Accuracy   Kappa
  0e+00  0.7911730  0.5788772
  1e-04  0.7911730  0.5788772
  1e-01  0.7899546  0.5762046

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was decay = 1e-04.

$qda
Quadratic Discriminant Analysis

800 samples
  2 predictor
  2 classes: '2', '7'

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 800, 800, 800, 800, 800, 800, ...
Resampling results:

  Accuracy   Kappa
  0.8336128  0.6641262


$rf
Random Forest

800 samples
  2 predictor
  2 classes: '2', '7'

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 800, 800, 800, 800, 800, 800, ...
Resampling results:

  Accuracy   Kappa
```

```
  0.8057573  0.6095341

Tuning parameter 'mtry' was held constant at a value of 2

$adaboost
AdaBoost Classification Trees

800 samples
  2 predictor
  2 classes: '2', '7'

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 800, 800, 800, 800, 800, 800, ...
Resampling results across tuning parameters:

  nIter  method         Accuracy   Kappa
   50    Adaboost.M1    0.7943614  0.5870302
   50    Real adaboost  0.8121324  0.6227061
  100    Adaboost.M1    0.7969295  0.5924086
  100    Real adaboost  0.8119475  0.6222216
  150    Adaboost.M1    0.7955893  0.5896559
  150    Real adaboost  0.8124907  0.6232505

Accuracy was used to select the optimal model using the largest value.
The final values used for the model were nIter = 150 and method = Real adaboost.

>
>
>
> names(fits)
 [1] "glm"         "lda"         "naive_bayes" "svmLinear"   "knn"
 [6] "gamLoess"    "multinom"    "qda"         "rf"          "adaboost"
> preds <- sapply(fits, function(fit) {
+ predict(fit, newdata = mnist_27$test)
+ })
> dim(preds)
[1] 200  10
> preds[1:5,]
     glm lda naive_bayes svmLinear knn gamLoess multinom qda rf  adaboost
[1,] "2" "2" "2"         "2"       "2" "2"      "2"      "2" "2" "2"
[2,] "7" "7" "7"         "7"       "7" "7"      "7"      "7" "7" "7"
[3,] "7" "7" "7"         "7"       "7" "7"      "7"      "7" "7" "7"
[4,] "7" "7" "7"         "7"       "7" "7"      "7"      "7" "7" "7"
[5,] "7" "7" "7"         "7"       "7" "7"      "7"      "7" "7" "7"
>
>
>
> accuracy <- colMeans(preds == mnist_27$test$y)
> accuracy
        glm         lda naive_bayes   svmLinear         knn    gamLoess
      0.750       0.750       0.795       0.755       0.840       0.845
   multinom         qda          rf    adaboost
      0.750       0.820       0.780       0.805
> mean(accuracy)
[1] 0.789
>
>
>
> votes <- rowMeans(preds == "7")
> y_hats <- ifelse(votes > 0.5, "7", "2")
> mean(y_hats == mnist_27$test$y)
[1] 0.815
>
>
```

```
>
> ind <- accuracy > mean(y_hats == mnist_27$test$y)
> accuracy[ind]
     knn gamLoess      qda
   0.840    0.845    0.820
> models[ind]
[1] "knn"      "gamLoess" "qda"
>
>
>
> accuracy_hat <- sapply(fits, function(fit) {
+ min(fit$results$Accuracy)
+ })
> accuracy_hat
        glm         lda naive_bayes   svmLinear         knn    gamLoess
  0.8010386   0.7987611   0.8128524   0.7969828   0.8067598   0.8455962
   multinom         qda          rf    adaboost
  0.7899546   0.8336128   0.8057573   0.7943614
> mean(accuracy_hat)
[1] 0.8085677
>
>
>
> ind <- accuracy_hat >= 0.8
> accuracy_hat[ind]
        glm naive_bayes         knn    gamLoess         qda          rf
  0.8010386   0.8128524   0.8067598   0.8455962   0.8336128   0.8057573
> models[ind]
[1] "glm"         "naive_bayes" "knn"         "gamLoess"    "qda"
[6] "rf"
> votes <- rowMeans(preds[,ind] == "7")
> y_hats <- ifelse(votes > 0.5, "7", "2")
> mean(y_hats == mnist_27$test$y)
[1] 0.83
>
```