data modelling

Don Li

12/06/2020

```
library( data.table )
library( caret )
```

```
## Warning: package 'caret' was built under R version 3.6.3
```

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

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

```
library( mgcv )
```

```
## Loading required package: nlme
```

```
## This is mgcv 1.8-28. For overview type 'help("mgcv-package")'.
```

```
library( pls )
```

```
## Warning: package 'pls' was built under R version 3.6.3
```

```
##
## Attaching package: 'pls'
```

```
## The following object is masked from 'package:caret':
##
##     R2
```

```
## The following object is masked from 'package:stats':
##
##     loadings
```

```
library( ranger )
```

```
## Warning: package 'ranger' was built under R version 3.6.3
```

```
library( randomGLM )
```

```
## Warning: package 'randomGLM' was built under R version 3.6.3
```

```
## Loading required package: MASS
```

```
## Loading required package: foreach
```

```
## Loading required package: doParallel
```

```
## Loading required package: iterators
```

```
## Loading required package: parallel
```

```
library( xgboost )
```

```
## Warning: package 'xgboost' was built under R version 3.6.3
```

```
library( brnn )
```

```
## Warning: package 'brnn' was built under R version 3.6.3
```

```
## Loading required package: Formula
```

```
## Loading required package: truncnorm
```

```
## Warning: package 'truncnorm' was built under R version 3.6.3
```

```
library( mgcv )
```

Prepare the data

Modelling with all the data and a small OOB sample.

```
load( "G:/azure_hackathon/datasets2/trip_summary/trip_summary2_landmark.RData" )
source( "G:/azure_hackathon/data/Don2/historical.R" )

set.seed(999)
inTrain = createDataPartition( trip_summary$trj_id, p = 0.75 )

historical_data = make_historical_data( trip_summary[inTrain$Resample1] )
save( historical_data,
  file = "G:/azure_hackathon/datasets2/model_final/historical.RData" )

trip_other_vars = trip_summary[ , list( trj_id, path_dist, mean_speed, timediff, path_dist2 ) ]
vars_to_rm = c("timediff", "path_dist", "path_dist2",
    "sampling_rate_var", "log_sampling_rate_var", "mean_speed", "log_var_speed",
    "sampling_rate",
    "trip_start", "trip_end")
trip_summary[ , eval(vars_to_rm) := NULL ]

# Save the out-of-bag set
OOB_data = trip_summary[ -inTrain$Resample1 ]
OOB_extra = trip_other_vars[ -inTrain$Resample1 ]

save( OOB_data, OOB_extra,
  file = "G:/azure_hackathon/datasets2/model_final/OOB.RData" )

# Get the other data
trip_data = trip_summary[ inTrain$Resample1 ]
trip_data_extra = trip_other_vars[ inTrain$Resample1 ]

set.seed( 843 )
cv_folds = 7
cv_fold_id = createFolds( trip_data$trj_id, k = cv_folds, returnTrain = T )
train_control = trainControl(
    method = "cv", number = cv_folds,
    verboseIter = TRUE,
    search = "grid",
    index = cv_fold_id, savePredictions = "final",
    returnData = FALSE
)

save( trip_data, trip_data_extra, train_control,
  file = "G:/azure_hackathon/datasets2/model_final/training.RData" )
```

Imputations

```
library(doParallel)
cl <- makePSOCKcluster(8, outfile = "out.txt")
registerDoParallel(cl)
# stopCluster(cl)
```

Impute path distance

```
load( "G:/azure_hackathon/datasets2/model_final/training.RData" )
source( "G:/azure_hackathon/data/Don2/impute_distances.R" )

trip_data[ trip_data_extra, path_dist2 := i.path_dist2, on = "trj_id" ]
trip_data[ , trj_id := NULL ]

impute_path_dist2 = train_dist_impute_model( 2, train_control, trip_data )
imputed_path_dist2 = dist_impute_model_predict( impute_path_dist2, trip_data )

trip_data_extra[ , path_dist2_impute := imputed_path_dist2 ]
trip_data[ , path_dist2_impute := imputed_path_dist2 ]
trip_data[ , path_dist2 := NULL ]

trip_data[ , path_dist := trip_data_extra$path_dist ]
impute_path_dist = train_dist_impute_model( 1, train_control, trip_data )
imputed_path_dist = dist_impute_model_predict( impute_path_dist, trip_data )

trip_data_extra[ , path_dist_impute := imputed_path_dist ]
trip_data[ , path_dist_impute := imputed_path_dist ]
trip_data[ , path_dist := NULL ]

save( impute_path_dist,
  file = "G:/azure_hackathon/datasets2/model_final/impute_dist.RData" )
save( impute_path_dist2,
  file = "G:/azure_hackathon/datasets2/model_final/impute_dist2.RData" )
save( trip_data, trip_data_extra, train_control,
  file = "G:/azure_hackathon/datasets2/model_final/training2_distimpute.RData" )
```

Join historical data and impute speed

```
load( "G:/azure_hackathon/datasets2/model_final/training2_distimpute.RData" )
load( "G:/azure_hackathon/datasets2/model_final/historical.RData" )
source( "G:/azure_hackathon/data/Don2/historical.R" )
source( "G:/azure_hackathon/data/Don2/impute_speed.R" )

trip_data = data.table(trip_data)
join_historical_data( trip_data, historical_data )
trip_data[ , mean_speed := trip_data_extra$mean_speed ]

impute_speed = train_speed_impute_model( train_control, trip_data )
imputed_speed = speed_impute_model_predict( impute_speed, trip_data )
```

```
trip_data_extra[ , mean_speed_impute := imputed_speed ]
trip_data[ , mean_speed_impute := imputed_speed ]
trip_data[ , mean_speed := NULL ]

save( impute_speed,
  file = "G:/azure_hackathon/datasets2/model_final/impute_speed_model.RData"
)
save( trip_data, trip_data_extra, train_control,
  file = "G:/azure_hackathon/datasets2/model_final/training3_speedimpute.RData" )
```

Train the model

```
load( "G:/azure_hackathon/datasets2/model_final/training3_speedimpute.RData" )
source( "G:/azure_hackathon/data/Don2/stack_training.R" )

trip_data = data.table( trip_data )
trip_data[ , timediff := trip_data_extra$timediff ]

stacking_model = train_0_models( train_control, trip_data,
  savefile = "G:/azure_hackathon/datasets2/model_final/submodel/model_part.RData" )
save( stacking_model,
  file = "G:/azure_hackathon/datasets2/model_final/stack_model.RData")
stack = stacking_model$model1
save( stack,
  file = "G:/azure_hackathon/datasets2/model_final/just_stack.RData" )
```

```
load( "G:/azure_hackathon/datasets2/model_final/just_stack.RData" )
stack
```

```
## Principal Component Analysis
##
## No pre-processing
## Resampling: Cross-Validated (7 fold)
## Summary of sample sizes: 18000, 18000, 18000, 18000, 18000, 18000, ...
## Resampling results across tuning parameters:
##
##   ncomp  RMSE      Rsquared   MAE
##    1     239.9867  0.4726026  163.6662
##    2     239.9895  0.4725862  163.6188
##    3     238.8435  0.4776975  162.3978
##    4     236.7183  0.4868504  160.5904
##    5     235.5201  0.4921152  160.1458
##    6     235.2196  0.4934364  160.0468
##    7     234.9992  0.4944190  159.8063
##    8     234.7119  0.4956369  159.6783
##    9     234.7337  0.4955443  159.6912
##   10     234.7247  0.4956013  159.7387
##   11     234.7306  0.4955453  159.7423
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was ncomp = 8.
```

I decided to take a lot of the linear models out of the stack.

```
load( "G:/azure_hackathon/datasets2/model_final/stack_model.RData")
load( "G:/azure_hackathon/datasets2/model_final/training3_speedimpute.RData" )

OOF_predictions = lapply( stacking_model$models0, function(k){
  a = data.table( k$pred )
  setorder( a, rowIndex )
  a$pred
} )
OOF_predictions = as.data.table( OOF_predictions )
OOF_predictions[ , c("lm1", "enet1", "PLS1") := NULL ]
timediff_ = data.table( stacking_model$models0$lm1$pred )[ order(rowIndex ), obs ]
OOF_predictions$timediff = timediff_
cat( "Training stacking\n" )

stacked_tunegrid = data.frame( neurons = 1:10 )
stacking = train( timediff ~ ., data = OOF_predictions,
    metric = "RMSE", method = "brnn", trControl = train_control,
    tuneGrid = stacked_tunegrid
)
stacking_model$model1 = stacking
stacking_model$models0$lm1 = NULL
stacking_model$models0$enet1 = NULL
stacking_model$models0$PLS1 = NULL

save( stacking_model,
  file = "G:/azure_hackathon/datasets2/model_final/stack_model2.RData")
```

Predictions

```
load( "G:/azure_hackathon/datasets2/model_final/OOB.RData" )
load( "G:/azure_hackathon/datasets2/model_final/impute_dist2.RData" )
source( "G:/azure_hackathon/data/Don2/impute_distances.R" )
OOB_imputed_path_dist2 = dist_impute_model_predict( impute_path_dist2, OOB_data )
OOB_data[ , path_dist2_impute := OOB_imputed_path_dist2 ]
rm( impute_path_dist2, OOB_imputed_path_dist2 )

load( "G:/azure_hackathon/datasets2/model_final/impute_dist.RData" )
OOB_impute_path_dist = dist_impute_model_predict( impute_path_dist, OOB_data )
OOB_data[ , path_dist_impute := OOB_impute_path_dist ]
rm( impute_path_dist, OOB_impute_path_dist )

load( "G:/azure_hackathon/datasets2/model_final/historical.RData" )
source( "G:/azure_hackathon/data/Don2/historical.R" )
join_historical_data( OOB_data, historical_data )

load( "G:/azure_hackathon/datasets2/model_final/impute_speed_model.RData" )
source( "G:/azure_hackathon/data/Don2/impute_speed.R" )
OOB_imputed_speed = speed_impute_model_predict( impute_speed, OOB_data )
OOB_data[ , mean_speed_impute := OOB_imputed_speed ]
```

```
rm( impute_speed, speed_impute_model_predict, historical_data )

load( "G:/azure_hackathon/datasets2/model_final/stack_model2.RData" )
source( "G:/azure_hackathon/data/Don2/stack_training.R" )

final_predictions = stack_predictions( stacking_model, OOB_data )

save( final_predictions, OOB_data, OOB_extra,
  file = "G:/azure_hackathon/datasets2/model_final/final_predictions.RData" )
```

Evaluation

```
load( "G:/azure_hackathon/datasets2/model_final/final_predictions.RData" )
final_RMSE = sqrt( colMeans( (final_predictions - OOB_extra$timediff)^2 ) )
final_RMSE
```

```
##      stack       PCR1       KNN1     Rpart1       GAM1      brNN1        RF1
##   298.5570   309.5133   315.3801   314.5623   307.1952   306.0734   302.5596
##   treebag1 randomGLM1       XGB1
##   323.0851   309.4235   297.2055
```

The stack model is slightly higher RMSE than XGBoost, but should be better in the long run.