Required sequential steps to generate a submission using the provided files

The following steps should allow you to generate a submission which comes close to the second place submission (private LB score of .0313157) of the "Santander Product Recommendation" Kaggle competition.

Running all steps on my 48GB workstation would take about seven days. Of these seven days, about four days are required to generate the feature ranking when using 5 folds combined with 100 xgboost rounds for the base models. The next three days are required for the base model generation when using 10 folds combined with 200 xgboost rounds trained on the top 100 features for each lag-product pair. The first four days can however be skipped since I already ran the feature ordering for you (product month feature order.rds).

Generating a ~.0310 private leaderboard score could however be achieved in about one hour by training using only the top 10 features for each lag-product pair and by dropping the 10 folds (only train on all of the train data).

General

Replace the working directory paths (always at the top of the scripts) to match your local file structure.

Replace other folder paths. The "targetDate" of the data files refers to the subfolder of the different iterations. I left this at "12-11-2016" but you could add iterations.

There are many options in the scripts at the top of the files. The options are mostly commented generously.

Overview

Required steps to generate a submission:

1. Prepare the data
2. Feature generation
3. Feature ordering (Optional)
4. Base model generation
5. Base test model predictions generation
6. Base test model predictions combination
7. Ensembling (Optional)

1) Prepare the data

Download the train_ver2 and test_ver2 csv files from Kaggle and paste them into the Data folder

Run **rawToRds.R** in the Data folder to convert the raw data into rds format

2) Feature generation

A) Generate common feature information

- Count the number of new products for each studied month
  - ⇨ Run **newProductsRate.R** in Exploratory data analysis
- Calculate the MAP and count fraction for the products in the test data set
  - ⇨ Run **fractionMAPProducts.R** in Exploratory data analysis
- List the clients that have records in May/June 2015

⇨ Run **listMayJun15Clients.R** in Feature Engineering

- List the clients that have at least one new product bought in the training data
  ⇨ Run **listClientsAnyPosFlank.R** in Feature Engineering
- Calculate the mean income by province
  ⇨ Run **incomeByProvince.R** in Feature Engineering
- List the families in the train data (same income and province)
  ⇨ Run **findFamilies.R** in Feature Engineering
- Load the mapping file that converts categorical features to numerical features
  ⇨ Run **catFeaturesToCont.R** in Feature Engineering

B) Generate features
⇨ Generate train features
Set summaryType to train in **createFeatures.R** in Feature Engineering
Run **createFeatures.R** in Feature Engineering
⇨ Generate test features
Set summaryType to test in **createFeatures.R** in Feature Engineering
Run **createFeatures.R** in Feature Engineering

3) Feature ordering (Optional)

A) Generate the first level base model relative weights
Run **weightGeneratorFirst.R** in Model weights

Default weights:

Product/Lag	3	4	5 Jun15	6	7	8	9	10	11	12	13	14	15	16 May16
Cco_fin	0.3	0.3	13	0.3	0.3	0.3	0.3	0.3	9	0.3	0.3	0.3	0.3	0.3
Ctma_fin	0	0	0	0	0	0	1.8	1.9	2.0	2.1	2.2	2.3	2.4	2.5
Ecue_fin	1.2	0	1.4	1.5	1.6	1.7	1.8	1.9	2.0	2.1	2.2	2.3	2.4	2.5
Reca_fin	1.2	1.3	52	1.5	1.6	1.7	1.8	1.9	2.0	2.1	2.2	2.3	2.4	2.5
Recibo	0	1.3	13	1.5	1.6	1.7	1.8	1.9	2.0	2.1	2.2	2.3	2.4	2.5
Other	1.2	1.3	13	1.5	1.6	1.7	1.8	1.9	2.0	2.1	2.2	2.3	2.4	2.5

The weights are normalized by product so that they sum to one for each of the 24 products

B) Generate the random folds
Set K1 to 5 in **subsetter.R** in first level learners
Run **subsetter.R** in first level learners

C) Generate the base models
Set

- ⇨ saveFolderExtension to " All features 100 rounds 5 Folds"
- ⇨ featureSelection to FALSE
- ⇨ nrounds to 100
- ⇨ baseK and K to 5
- ⇨ saveBaseModels to TRUE
- ⇨ skipCommonModel to FALSE

in **xgboost.R** in first level learners

Run xgboost.R in first level learners – Bottleneck: this will probably run for several days

D) Generate the feature ordering
Set modelGroup to "trainTrainAll All features 100 rounds 5 Folds" in **featureSelection.R** in first level learners
Run **featureSelection.R** in first level learners

4) Base model generation

A) Generate the first level base model relative weights (not necessary if already executed in optional step 3: Feature ordering)
Run **weightGeneratorFirst.R** in Model weights

B) Generate the random folds
Set K1 to 10 in **subsetter.R** in first level learners
Run **subsetter.R** in first level learners

C) Generate the base models
Set
- ⇨ saveFolderExtension to " Top 100 monthProduct 200 rounds 10 Folds"
- ⇨ featureSelection to TRUE
- ⇨ featureSelectionMode to "monthProduct"
- ⇨ nrounds to 200
- ⇨ baseK and K to 10
- ⇨ saveBaseModels to TRUE
- ⇨ skipCommonModel to FALSE

in **xgboost.R** in first level learners

Run xgboost.R in first level learners – Bottleneck: this will probably run for several days

5) Base test model predictions generation

Set
- ⇨ trainModelsFolder to "trainTrainAll Top 100 monthProduct 200 rounds 10 Folds"
- ⇨ submissionDate to date_XXX
- ⇨ submissionFile to submission_YYY

in **Submission/20-12-2016/xgboost weighted 22.R**

Run **Submission/20-12-2016/xgboost weighted 22.R**

This script takes about 30 minutes – several hours depending on whether the folds were also generated or not (skipCommonModel in Step 4) and generates a csv submission file as well as base model predictions which are further processed in step 6.

6) Base test model predictions combination

This step builds on step 5 in order to reduce the submission generation turnaround time when trying new base model weights or post-processing steps using the base model predictions

A) Combine base model predictions to a data table structure
Set
- ⇨ onlyPreds to TRUE
- ⇨ submissionDate to date_XXX
- ⇨ modelName to submission_YYY
- ⇨ modelFolder to "trainTrainAll Top 100 monthProduct 200 rounds 10 Folds"

In **Submission/PredictionsComp/monthProductProbs.R**

Run **Submission/PredictionsComp/monthProductProbs.R**

B) Combine the data table base test model predictions to a submission
Set
- ⇨ baseModelsDate to date_XXX
- ⇨ submissionDate to date_XXX
- ⇨ loadPredsFile to submission_YYY**.rds**
- ⇨ submissionFile to my_submission
- ⇨ trainModelsFolder to "trainTrainAll Top 100 monthProduct 200 rounds 10 Folds"

In **Submission/20-12-2016/basePredictionsCombiner.R**

Run **Submission/20-12-2016/basePredictionsCombiner.R**

All different last 26 submissions can now easily be generated by making parameter adjustments in **basePredictionsCombiner.R** and running that same script with different submissionFile names. Running **weightGeneratorFirst.R** with different base model weights will makes sure that the base models are weighted using the most recent weights.

7) Ensembling (Optional)

A) Point to the path of all base submissions by adjusting/copying and renaming Ensembling/Submission Inventory.xlsx and update the Model paths sheet. The ensembling logic uses columns D, E and F

B) Generate the base submission agreement matrix
Adapt inventoryFn to the new base submissions mapping excel file if a new mapping file was created in A in **submissionAgreementCalc.R** in Ensembling
Run **submissionAgreementCalc.R** in Ensembling

C) Generate the ensembles

D) Adapt inventoryFn to the new base submissions mapping excel file if a new mapping file was created in A in **ensembler.R** in Ensembling
Modify the ensemble parameters (submissionSubset – scoreExponent) to your own taste in **ensembler.R** in Ensembling
Run **ensembler.R** in Ensembling