On Building Efficient Recommender System

tooYoungTooSimple
Advisor: Dr. Shannon Quinn

Department of Statistics
The University of Georgia

Overview

Introduction

Feature Engineering

Sample Creation

Statistical Modeling

Ongoing Works

Introduction : Motivation

- **Santander** is challenging us to predict which products their existing customers will use/purchase in the next month based on their past behavior and that of similar customers
- Current recommender system is unsatisfactory with uneven customer experience
- Ongoing Kaggle competition

Introduction : Motivation

- Extremely common, and utilized in a variety of areas
- Statistical modeling is a key solution
- Balance between recommendation and customer experience (If recommender system is too aggressive, more products will be recommended correctly, but many “irrelevant” customers will be annoyed.)

Introduction : Data

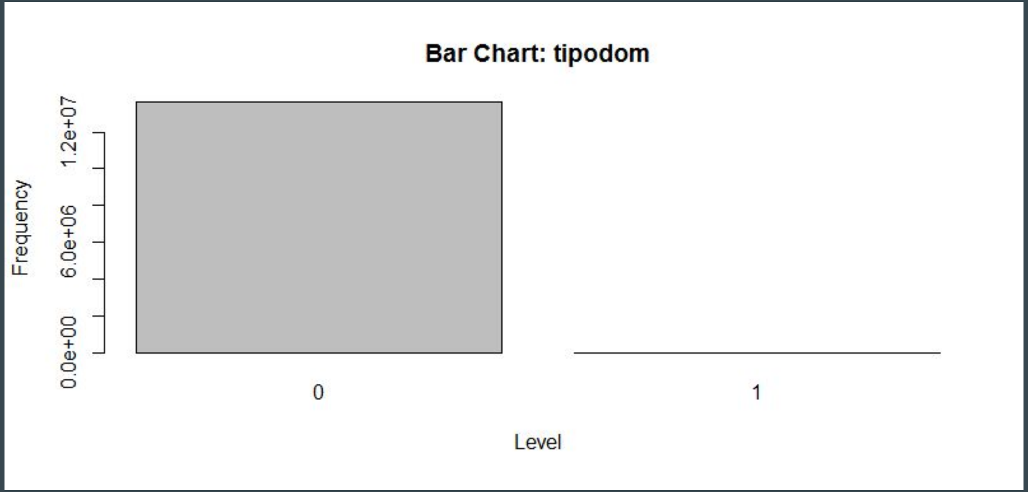
- Observations: 13,647,309 (training) and 929,615 (testing)
- Response: 24 Product Variables
  Binary and Sparse
- Predictor: 24 Demographic and Behaviour Variables
- All observations are from 17 months
  Training data from *2015/01 - 2016/05*
  Testing data from *2016/06*
- Good News! No new customers appear in testing data

Introduction : Challenging

- Big and Dirty Data
- Imbalance: Very Low Proportion of Products used
- Multiple Products used by Single Customer
- Status of using May Change in Different Patterns
- Missing Values in Response and Predictors

Feature Engineering

- Remove Extremely Imbalanced Variable
  - E.g., AddressType, with 1 category



Feature Engineering

- Impute Missing Values with Different Strategies
  - Predictors: Mode for Categorical Variables
    Median for Continuous Variables

E.g. ind_empleado

Level	(Missing)	A	B	F	N	S
Frequency	27734	2492	3566	2523	13610977	17

↓

Level	A	B	F	N	S
Frequency	2492	3566	2523	13638711	17

Feature Engineering

- Impute Missing Values with Different Strategies
  - Response: estimated sample expectation

  E.g. Ind_nomina_ult1 (Y_22)

  Assume a *Bernoulli Distribution*

  $Pr(Y\_22 = 0) = 0.95$

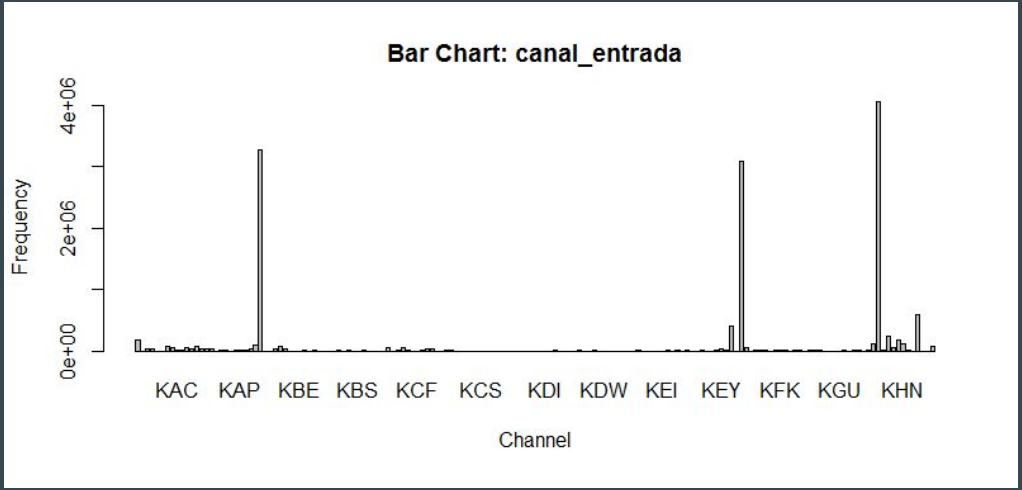
  $Pr(Y\_22 = 1) = 0.05$

  $E(Y\_22) = 0.05*1 + 0.95*0 = 0.05$

  Thus we impute *NAs* in *Y_22* as 0.05

Feature Engineering

- Collapse Levels with Low Frequencies
  E.g. canal_entrada
  163 levels: 004 007 013 025 K00 KAA … KAH KAI RED



Feature Engineering

- Collapse Levels with Low Frequencies
  E.g. canal_entrada

  163 levels: 004 007 013 025 K00 KAA … KAH KAI RED

  ↓

  13 levels: KHE KAT KFC KHQ KFA KHK KHM KHD
  KHN KAS KAG RED and Others

Feature Engineering

- Create 24 Customer Historical Product (*CHP*) Features
  - Past purchase behaviour influence the future purchase
  - Define CHP as weighted sum of past three years' purchase
  - *CHP(Year_i) = (⅙)*Y(Year_(i-3)) + (⅓)*Y(Year_(i-2)) + (½)*Y(Year_(i-1))*

    If i = 3, we use ⅓ and ⅔ as two weights

    If i = 2, we use the previous year's purchase

    If i = 1, we keep current year's purchase

Sample Creation

- Different response variable has different Pr(Y=1)
- Thus we create 24 samples based on corresponding Pr(Y=1)

For Y_i, keep all observations for Y_i = 1

Sampling from the observations with Y_i = 0

Such that *Pr(Y_i = 1) : Pr(Y_i = 0) = 1:5*

E.g., ind_nom_pens_utl1 (Y_23)

Original Data: #(Y_23=1) = 810,085, #(Y_23=0) = 12,821,161

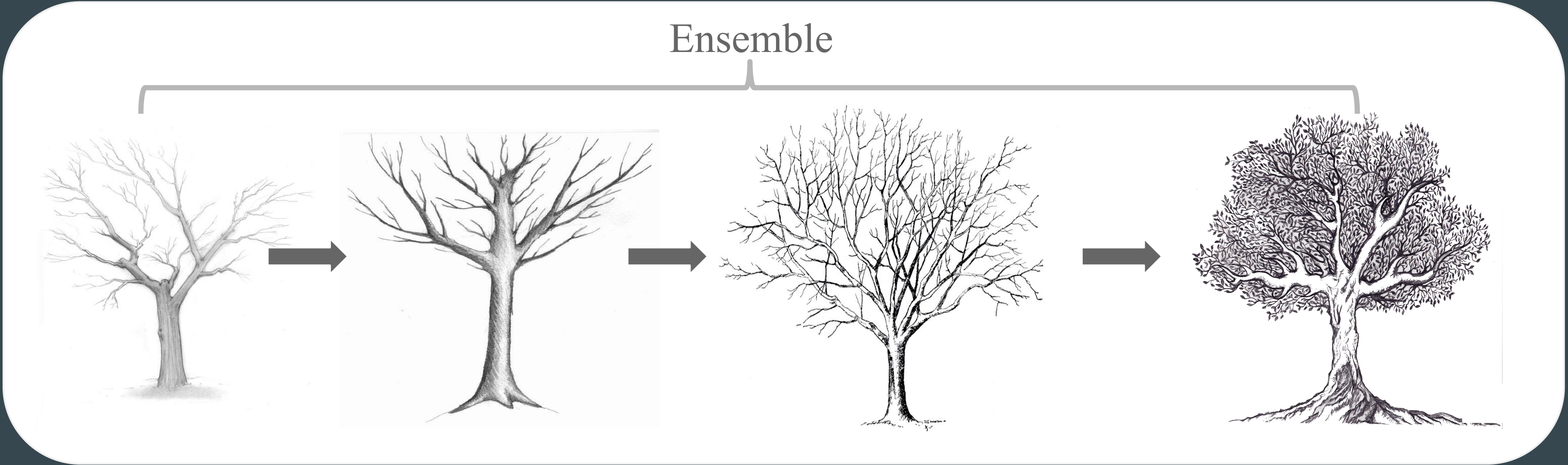
Sample Data: #(Y_23=1) = 810,085, #(Y_23=0) = 4,050,425

Modeling : Strategy and Procedure

- We have 24 samples in total
- For *sample_i*, taking *Y_i* as Response Variable
  taking all *X* and *CHP(-i)* as Predictors
- *model_i* trained with *sample_i* to predict *Y_i* in testing set

Modeling : Gradient Boosted Tree

- Building up the final model by utilizing the error of previous ones.
- Apply l-1 regularization to control overfitting.

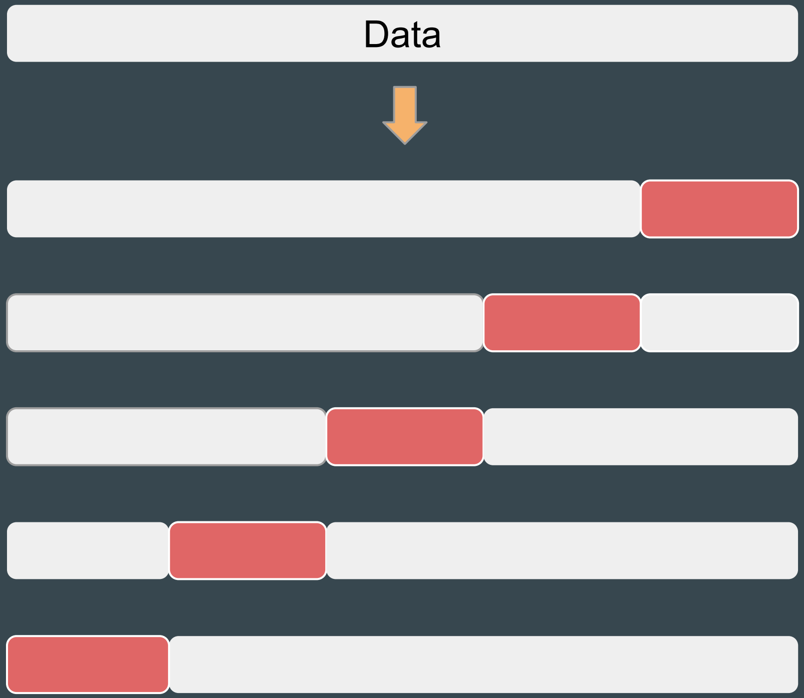


Modeling : XGBoost

XGBoost is an optimized distributed gradient boosting library designed to be highly ***efficient***, ***flexible*** and ***portable***.

System	exact greedy	approximate global	approximate local	out-of-core	sparsity aware	parallel
XGBoost	yes	yes	yes	yes	yes	yes
SparkMLLib	no	yes	no	no	partially	yes
scikit-learn	yes	no	no	no	no	no
R-GBM	yes	no	no	no	partially	no

Modeling : Cross Validation



5-fold CV for each sample:

- Training model with train set.
- Testing model with test set.
- Aggregate over 5 validations.

Test:

Training:

Ongoing Works

- Will train 24 models with corresponding samples and predict all 24 Responses in testing data
- Kaggle Submission!

Thank you !