Santander Product Recommendation

ECS 171 PROJECT TEAM 2

XINYI HOU, YINGXI YU, YUAN XU, WENYU LI, QIAOJUAN NIU, HUIYU BI, MENG LI, ZHONGYU FAN, AORAN ZHANG, YUAN TIAN, WANG MIAO, BOWEN HE, MARKHAM ANDERSON, CHARLTON LIN, ZORAN DABIC

Introduction

Approaches

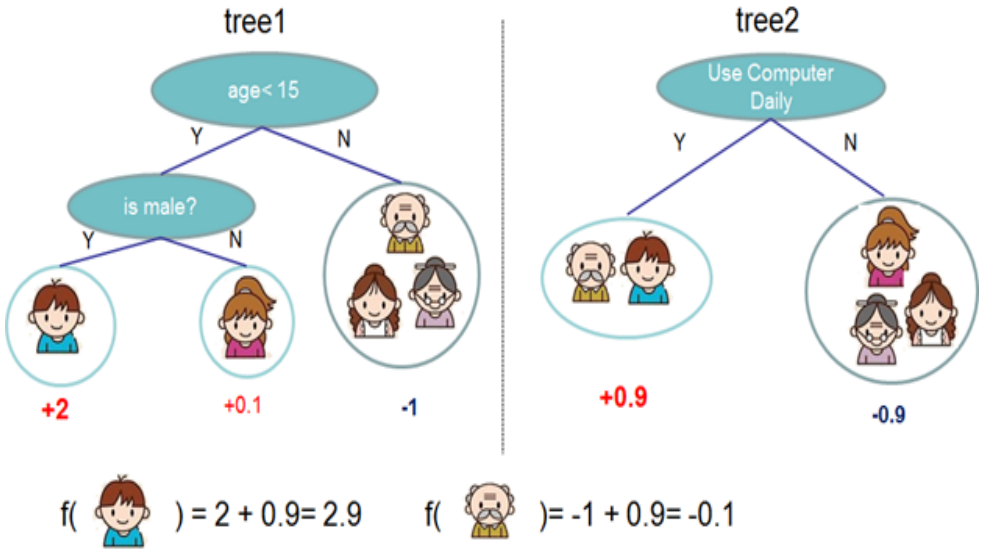
We tried 4 different machine learning methods: XGBoost, Random Forest, SVM, Logistic Regression. Since the data size is large and there is also no clear response in the training set, different subgroup used their own subset of the original training set and response variables.

Results

The logistic regression and SVM subteams both built 24 classifiers for the 24 products and the accuracy rate is above 90%. In contrast, SVM and XGBoost team both only build on multiclass classifier. The accuracy are 0.9 and 0.63 respectively. However, these two teams were using different cleaning method and response variable, merely comparing the accuracy rate is not meaningful. In summary, all four methods could provide reasonable prediction.

XGBoost

XGBoost is short for “Extreme Gradient Boosting”. In this case, the classifier is a collection of weak classifiers which are decision trees.



Example: 20 students cooperate on one exam

$Obj(\Theta) = L(\theta) + \Omega(\theta)$

$L(\theta)$: predictive performance,

$\Omega(\theta)$: model complexity

Adding new trees: gradient descent

XGBoost-Data Pre-processing

Since there are many missing values and outliers, the first thing is to delete 27334 records with 9 consecutive NA values and extreme values (for an instance, the 99 percent quantile of the age variable is 88 years old). Then, we decide to only use 32 features to predict the response. For the "age", "antiguedad" and "renta", we cut them into 5 intervals and they also become categorical features now.

The response is the additional product which the customer would add in the next month. If the customer adds more than one product, we only randomly pick one of them. To make the data size smaller and the problem easier, we subset the samples with nonzero response, which means we only consider the records with additional product.

XGBoost

Hyperparameter tuning

Total Number of Trees: 10,50,100

Depth of a tree: 5-7

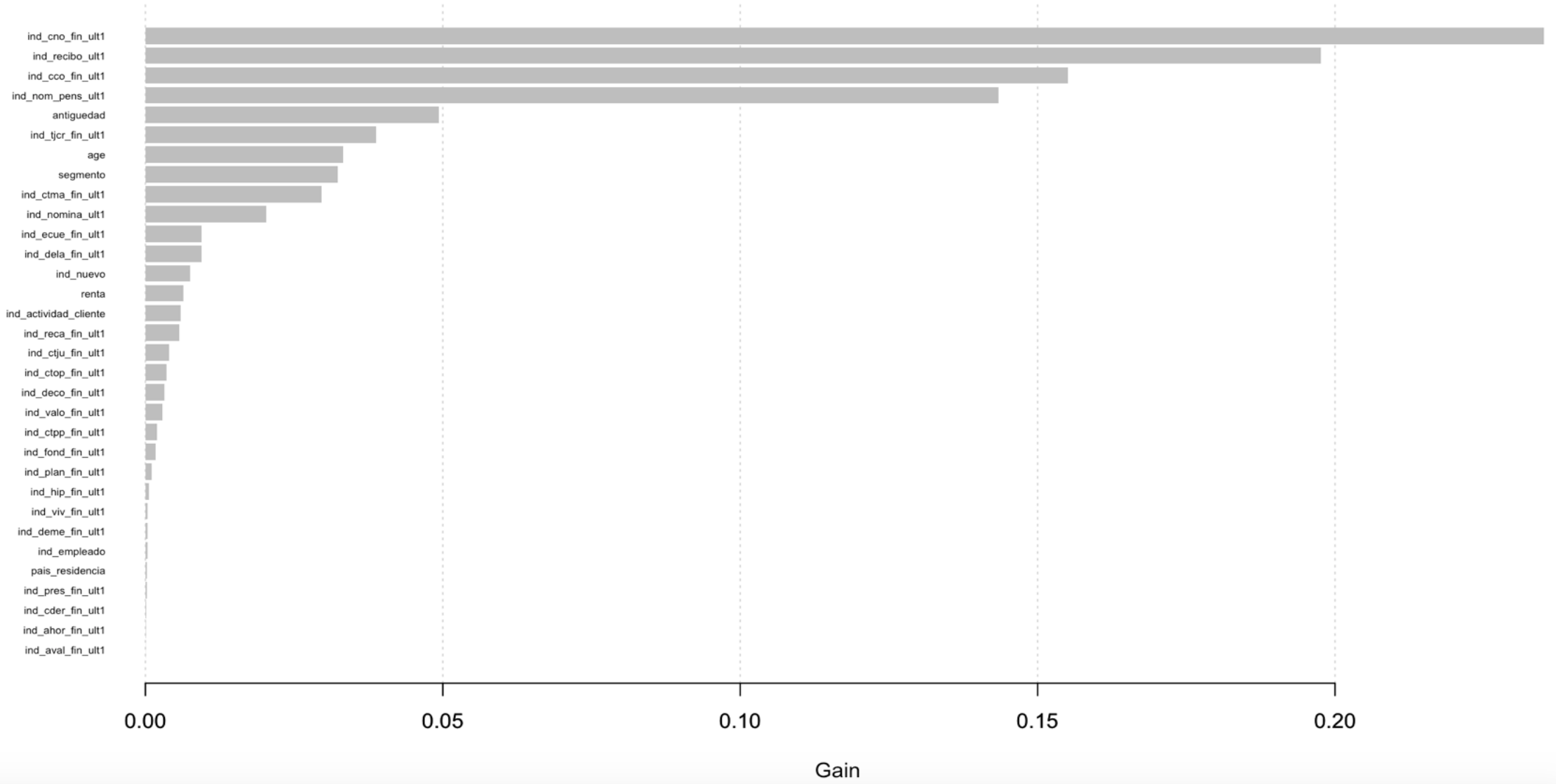
Shrinkage: {0.1, 0.5, 1}

Optimal combination: 100 trees with depth equals 6 and shrinkage equals 0.5

Results

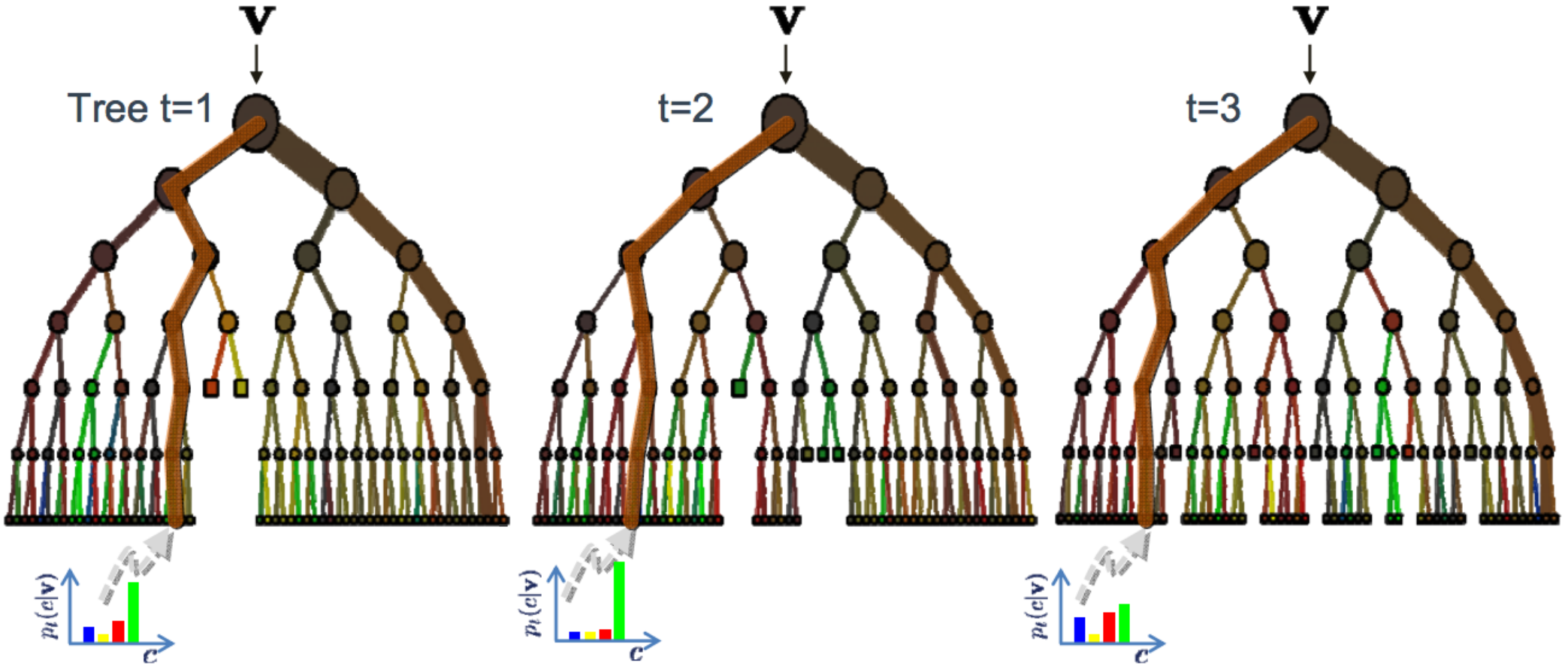
64.8% on training set and 62.94% on test set.

Kaggle: 665 out of 995，temporarily the best one over all subgroups


ind_cno_fin_ult1
ind_recibo_ult1
ind_cco_fin_ult1
ind_nom_pens_ult1
antiguedad
ind_tjcr_fin_ult1
age
segmento
ind_ctma_fin_ult1
ind_nomina_ult1
ind_ecue_fin_ult1
ind_dela_fin_ult1
ind_nuevo
renta
ind_actividad_cliente
ind_reca_fin_ult1
ind_ctju_fin_ult1
ind_ctop_fin_ult1
ind_deco_fin_ult1
ind_valo_fin_ult1
ind_ctpp_fin_ult1
ind_fond_fin_ult1
ind_plan_fin_ult1
ind_hip_fin_ult1
ind_viv_fin_ult1
ind_deme_fin_ult1
ind_empleado
pais_residencia
ind_pres_fin_ult1
ind_cder_fin_ult1
ind_ahor_fin_ult1
ind_aval_fin_ult1
0.00
0.05
0.10
0.15
0.20
Gain

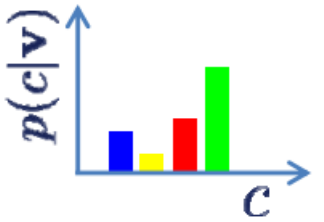
Random Forest

- Is an ensemble machine learning method.
- Produces many decision trees using bagging with replacement.
- At each split a random subset of features are selected without replacement
- Allowed to grow without pruning.
- Each tree results in a classification.
- SKlearn takes the class with the highest mean probability estimate.



The ensemble model

Forest output probability $p(c|\mathbf{v}) = \frac{1}{T}\sum_{t}^{T} p_t(c|\mathbf{v})$



Random Forest-Flattening

Sample Month	Customer ID	Demographic Columns	Behavioral Columns
1	1		
1	2		
1	3		
2	1		
2	2		
2	3		
		.	
		.	
		.	
N	1		
N	2		
N	3		

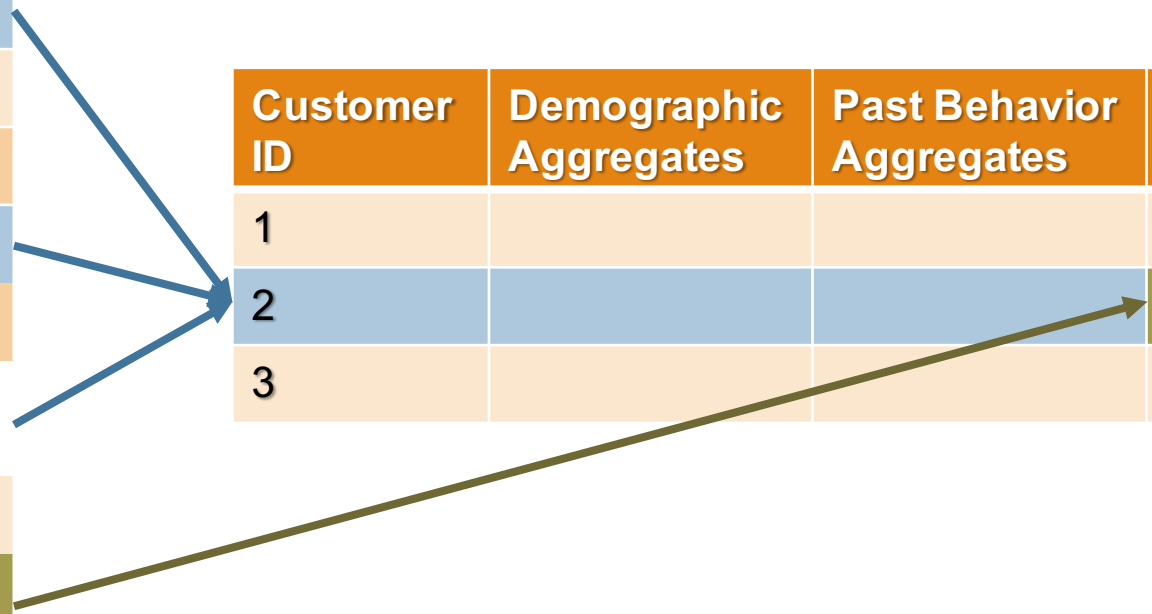
Random Forest-Flattening

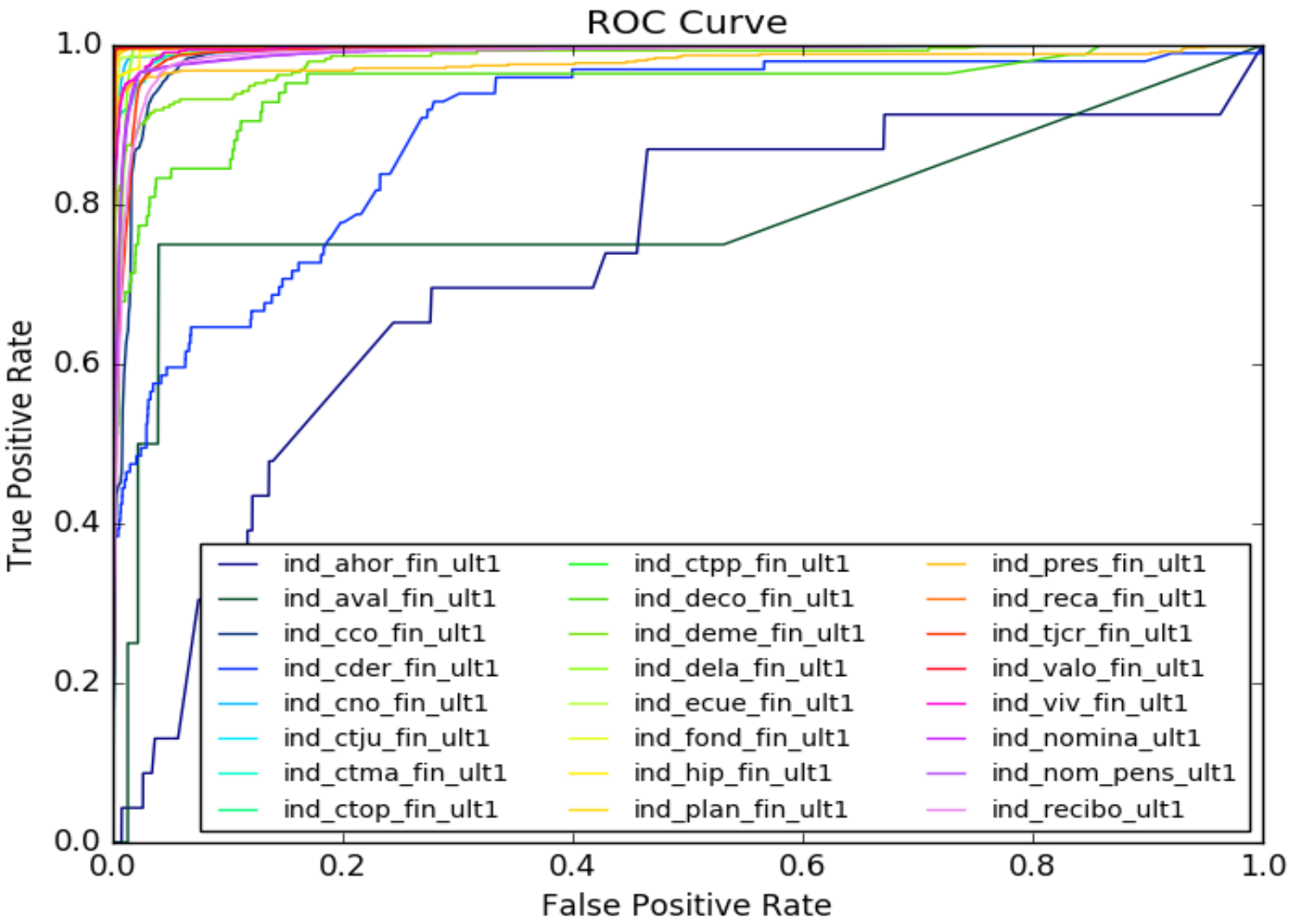
Sample Month	Customer ID	Demographic Columns	Behavioral Columns
1	1		
1	2		
1	3		
2	1		
2	2		
2	3		
		. . .	
N	1		
N	2		
N	3		

Random Forest-Flattening

Sample Month	Customer ID	Demographic Columns	Behavioral Columns
1	1		
1	2		
1	3		
2	1		
2	2		
2	3		
		. . .	
N	1		
N	2		
N	3		

Customer ID	Demographic Aggregates	Past Behavior Aggregates	Month N Behavior
1			
2			
3			





Accuracy Score:

Trees: 1, 10, 100;

Depth: 10, 100, infinity;

Best accuracy: 90.44%
with trees = 100,
depth = 100

SVM - Data Processing

1. Independent Variable Selection

- Delete discrete variables with too many classes.
- Leave out independent variables with repeated information.
- Removed variables with predominantly missing values.

Remove : pais residencia, fecha alta, indext, conyuemp, canal entrada, tipodom, cod prov and nomprov.

2. Missing values

3. Data Transformation

- Transform categorical variables to dummy variables.
- Standardize the continuous variables.

SVM-Model

Introduce slack variables that allow samples to be misclassified or be within the margin.

To make the samples more separable, we introduce the idea of kernel.

Decision function: $f(x) = \sum_i \alpha_i \, \phi(x_i)\phi(x) + b = \sum_i \alpha_i K(x_i, x) + b$

Dual function: $\min P(w, b) = \frac{1}{2} || \sum_{i=1}^{m} \alpha_i \phi(x_i) ||^2 + C \sum_i [H_1 y_i f(x_i)]$

SVM -Result

Criterion:

$$Accuracy = \frac{TP + TN}{P + N}$$

For each of the 24 estimation, the accuracy is higher than 99%

Discussion Highly skewed classes Problem

- Class 0 is predominant (more than 98%) in every response variable, therefore the svm classifier tends to predict 0.
- Confusion matrix for response variable ind_cco_fin_ult1

	Really True	Really False
Predicted True	0	0
Predicted False	558	99442

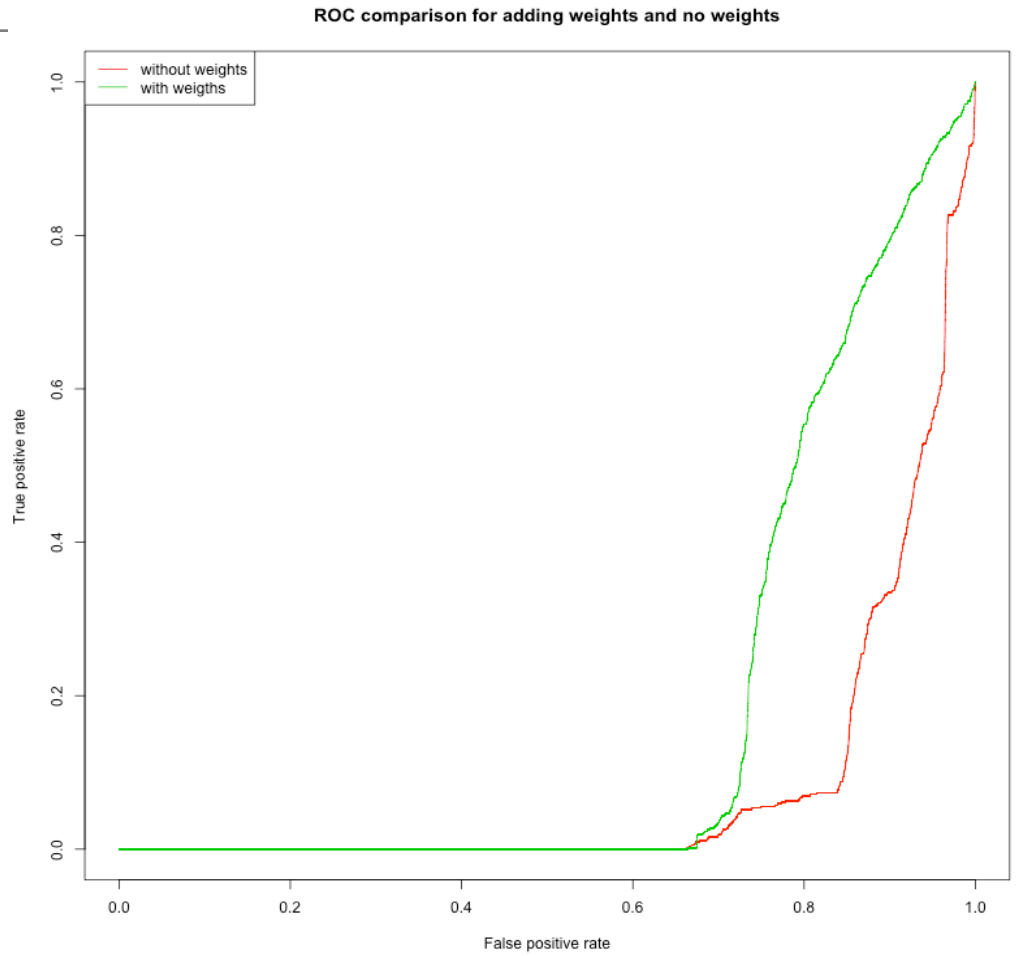
- Accuracy is 99.44%, recall is 0, precision is NA, the model is useless.
- First priority is recall, since the bank wants to predict what additional products a customer will purchase.

Solution

- Stratified Sampling to make the distribution of two classes more balanced in training set
- Adding weights to the two classes which are inverse proportional to their fractions.

	Really True	Really False
Predicted True	558	33488
Predicted False	0	65954

Accuracy is 66.51%, recall is 100%, precision is 1.64%



Logistic Regression

Why choose logistic regression?

- Classification Problem
- Large Data Scale
- Easy to understand

Logistic Regression

X（Predictors）

Features of Current Month + Purchase History

Y （Predictions）

Difference of products held between Current and Next Month

Logistic Regression

ETL:

1. Y-labels:

Binary, 1 if the purchase took place in the month after the data of sample, 0 otherwise.

Product / Month	1	2	3	….(4 to 21)	22	23	24
Month of that Sample (Ex: Jan)	1	0	0	……	1	0	1
The next month (Ex: Feb)	1	1	0	…..	1	0	1
Y-Label for that sample	**0**	**1**	**0**	……	**0**	**0**	**0**

Logistic Regression

ETL:

2. X-values (417 features):

1) **Purchase History**: For each product, add one column to hold number of months since the customer first had this product till the sample date.

Product / Month	1	2	3	….(4 to 21)	22	23	24
The purchase Month (Ex: Jan)	1	0	0	……	1	0	1
The next month (Ex: Feb)	1	1	0	…..	1	0	1
The history for that sample	**2**	**1**	**0**	……	**2**	**0**	**2**

Logistic Regression

ETL:

2) **Memberdays**: difference between “fecha_alta” (the date in which the customer first acquired a contract with the bank) and “fecha_dato” (the sample date).

3) **Catergorial variables** : replaced with dummy variables.

4) **Renta**: missing values filled by using averages based on province code.

5) **Omitted**: “tipodom”, “nomcodpers”, “fecha_dato”, “nom_prov”.

6) Eliminated 27334 rows which had exactly 9 NAs in all the same columns

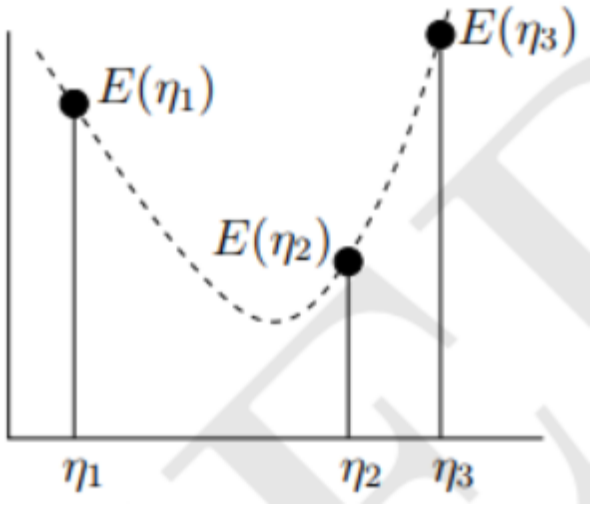
Logistic Regression

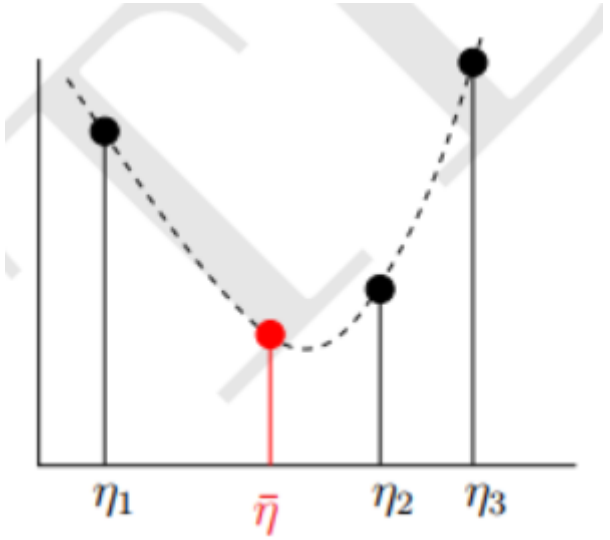
MSE = 0	ahor_fin:	cder_fin:	deme_fin:	hip_fin:	viv_fin:		
	0	0	0	0	0		
0 <MSE <0.001	ctma_fin:	ctop_fin:	deco_fin:	fond_fin:	plan_fin:	reca_fin:	valo_fin:
	0.000303	0.000455	0.000253	0.000253	0.000101	0.000707	0.000354
0.001 <MSE <0.01	cco_fin:	cno_fin:	ecue_fin:	dela_fin:	tjcr_fin:	nomina:	nom_pens:
	0.006567	0.002576	0.001667	0.001061	0.005203	0.005607	0.006921
0.01 <MSE <1	ctju_fin:	ctpp_fin:	recibo:				
	0.999949	0.999899	0.011517				
MSE = 1	aval_fin:	pres_fin:					
		1	1				

5 y-variables with zero error
2 y-variables with max error

Logistic Regression

Steepest Descent: Choosing the learning rate on an update-by-update basis





Conclusion-Kaggle Results

	Method	Score
1	XGboost	0.0178739
2	SVD Ensemble	0.0159378
3	Logistic Regression	0.0142893
4	Naïve Bayes	0.0122565
5	Random Forest	0.0103280
6	SVM	0.0094804
7	ANN	0.0092623
8	Collaborative Filtering	0.0057200

Q&A