CAPSTONE PROJECT FOR SPRINGBOARD DATA SCIENCE INTENSIVE PROGRAM

Santander Bank Product Prediction

Wei-Chun Hsieh

- Introduction
- Data exploratory analyses
- Build the base model
- Parameter tuning vis grid search
- Feature Importance
- Ensemble learning
- Summary
- Usage of results

Introduction

- Santander is a bank which offers financial products and services. The whole dataset is 1.5 years and has monthly records of multiple columns that store customer's information, such as customer code, status of employment, customer's country residence, age, gender of customer, gross income of the household, etc. There are also 24 product columns storing product items a customer has, such as "credit card", "savings account", "mortgage", "short-term deposits", "medium-term deposits", "long-term deposits", etc.

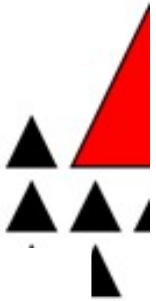
- Data source: https://www.kaggle.com/c/santander-product-recommendation

- Business problem: to predict if a customer has an account or not based on the customer's information.

24 products

1	saving_account	13	e-account
2	guarantees	14	funds
3	current_account	15	mortgage
4	derivada_account	16	pensions
5	payroll_account	17	loans
6	junior_account	18	taxes
7	mas_particular_account	19	credit_card
8	particular_account	20	securities
9	particular_plus_account	21	home_account
10	short-term_deposits	22	payroll
11	medium-term_deposits	23	pensions
12	long-term_deposits	24	direct_debit

Goal of project: prediction of current account holder



- The kaggle challenge: to predict what products Santander' customers will purchase in the next month based on past data
- 23 out of 24 products are highly imbalanced datasets
- To simplify the modeling problem, we built the prediction model on one product
- Select the product "current account" which has the most coverage
- Make prediction on if a customer hold the account with bank or not based on customer's info

Data exploratory analyses

- The downloaded training data has 13,647,309 rows. We reduced the input data to 20% of the original training data by randomly sampling the raw data.
- Original dataset has Spanish header, I changed the Spanish header to English header so we know fields better.
- Found there are NAN and missing values.
- Age and seniority have non-numeric values. So we forced non-numeric values to be NaN and removed them.

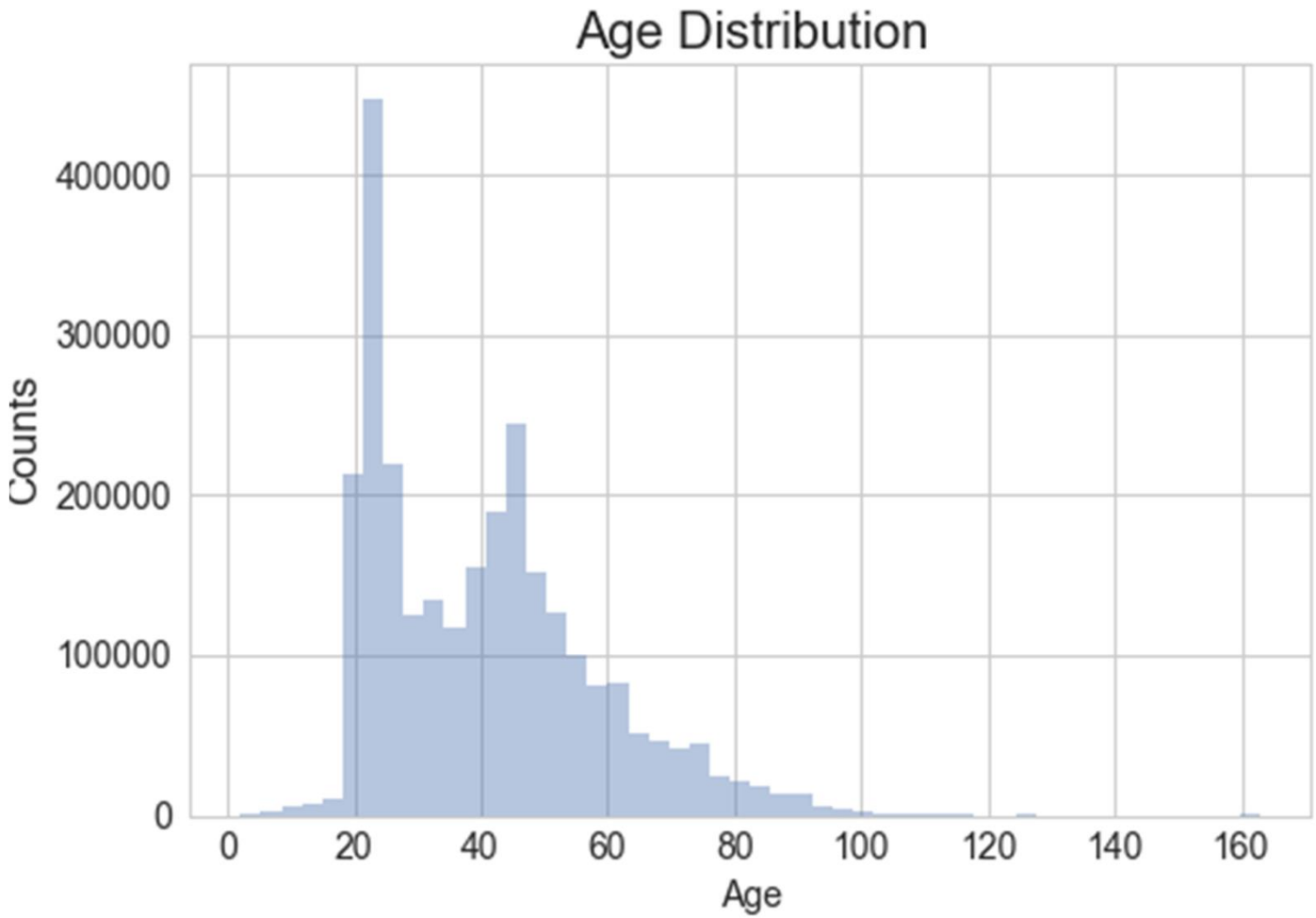
Description of data fields

Column	Description
employee_index	A active, B ex employed, F filial, N not employee, P pasive
country_residence	Customer's Country residence
sex	Customer's sex
age	Age
first_join_date	The date in which the customer became as the first holder of a contract in the bank
new_customer_index	New customer Index. 1 if the customer registered in the last 6 months
seniority	Customer seniority (in months)
primary	1 (First/Primary), 99 (Primary customer during the month but not at the end of the month)
customer_type	Customer type at the beginning of the month ,1 (First/Primary customer), 2 (co-owner),P (Potential),3 (former primary), 4(former co-owner)
customer_relation	Customer relation type at the beginning of the month, A (active), I (inactive), P (former customer),R (Potential)
residence_idx	Residence index (S (Yes) or N (No) if the residence country is the same than the bank country)
foreigner_idx	Foreigner index (S (Yes) or N (No) if the customer's birth country is different than the bank country)
channel	Channel used by the customer to join
deceased	Deceased index. N/S
address	Address type. 1, primary address
province_code	Province code (customer's address)
province_name	Province name
activity_idx	Activity index (1, active customer; 0, inactive customer)
gross_income	Gross income of the household
segment	segmentation: 01 - VIP, 02 - Individuals 03 - college graduated

Raw data

employee_index	N, A, S, F, B
country_residence	ES, BO, AR, IN, RO, PY, US, …, etc.
sex	H, V,
age	68, 42, 44, 49, 32, 36, 41, 45, 31, 34, 37, 40, 57, 51, 66, 35, 38, 95, 43, 29, 47, 33, 52, 39, …, etc
first_join_date	2006-11-17, 2006-11-18, 2006-11-19, 2006-11-15, ..., etc]
new_customer_index	0, 1
seniority	114, 91, 73, 44, 18, 0, 55, 112, 33, 68, 94, 110, 4, 113, 19, 58, …, etc
primary	1, 99
customer_type	1, 1.0, P, 3, 3.0, 2.0, , 2, 4.0, 4
customer_relation	I, A, R, P,
residence_idx	S, N
foreigner_idx	N, S
channel	KAQ, KBF, KAP, KAT, KFC, KFA, KAE, KAA, KAL, KAF, KCI, KHK, 013, …, etc.
deceased	N, S
address	1
province_code	28, 26, 48, 8, 50, 41, 35, 33, 51, 25, 43, 5, 47, 15, 7, …, etc.
province_name	MADRID, RIOJA, LA, BIZKAIA, BARCELONA, ZARAGOZA, SEVILLA, …,etc.
activity_idx	0, 1
segment	02 - PARTICULARES, 01 - TOP, , 03 - UNIVERSITARIO

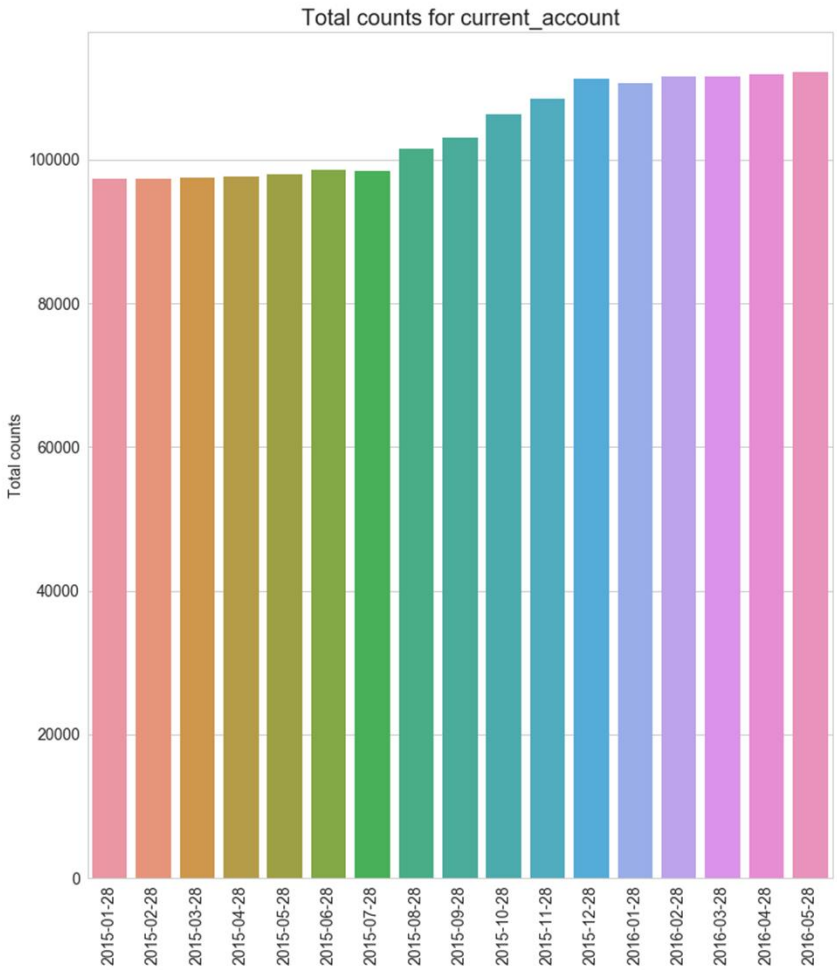
Age distribution: bimodal

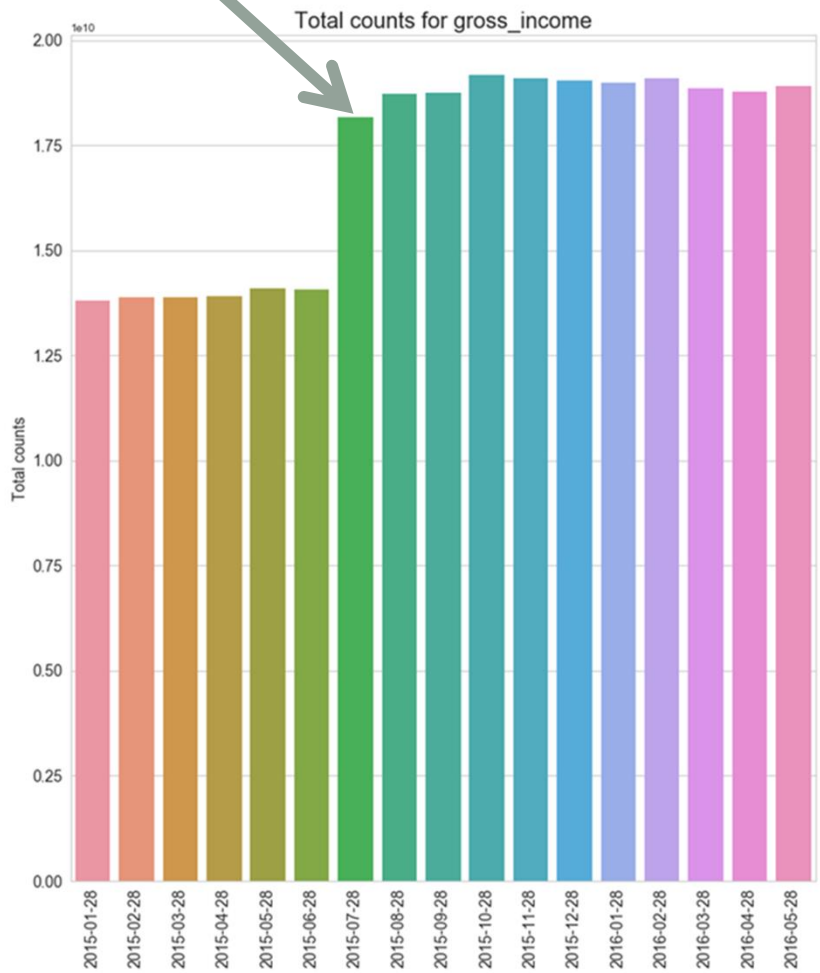


Peak for age distribution is around 25 and 45

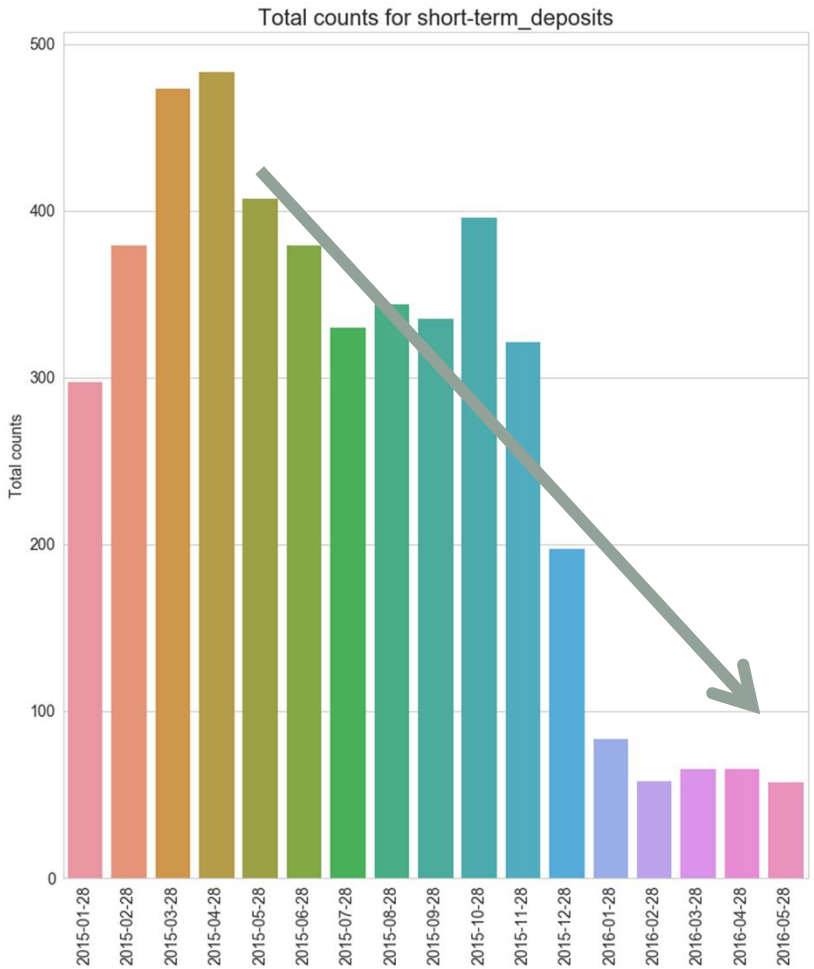
Total counts for current and gross income accounts

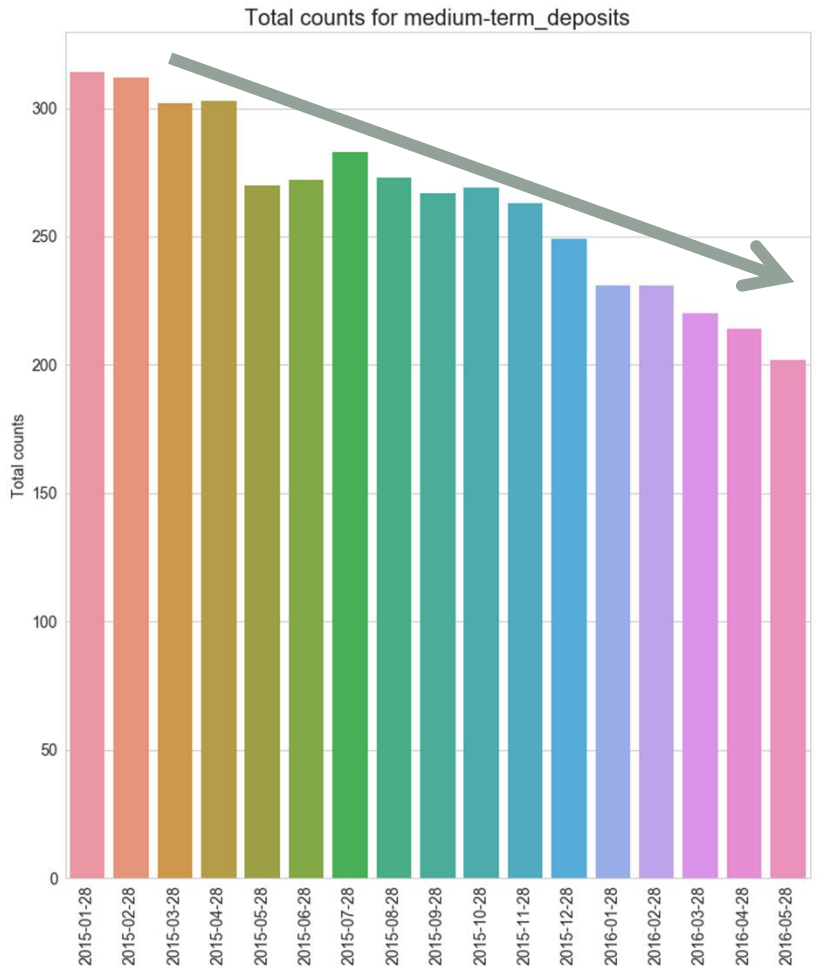
Jump on number of gross income accounts after June 2015



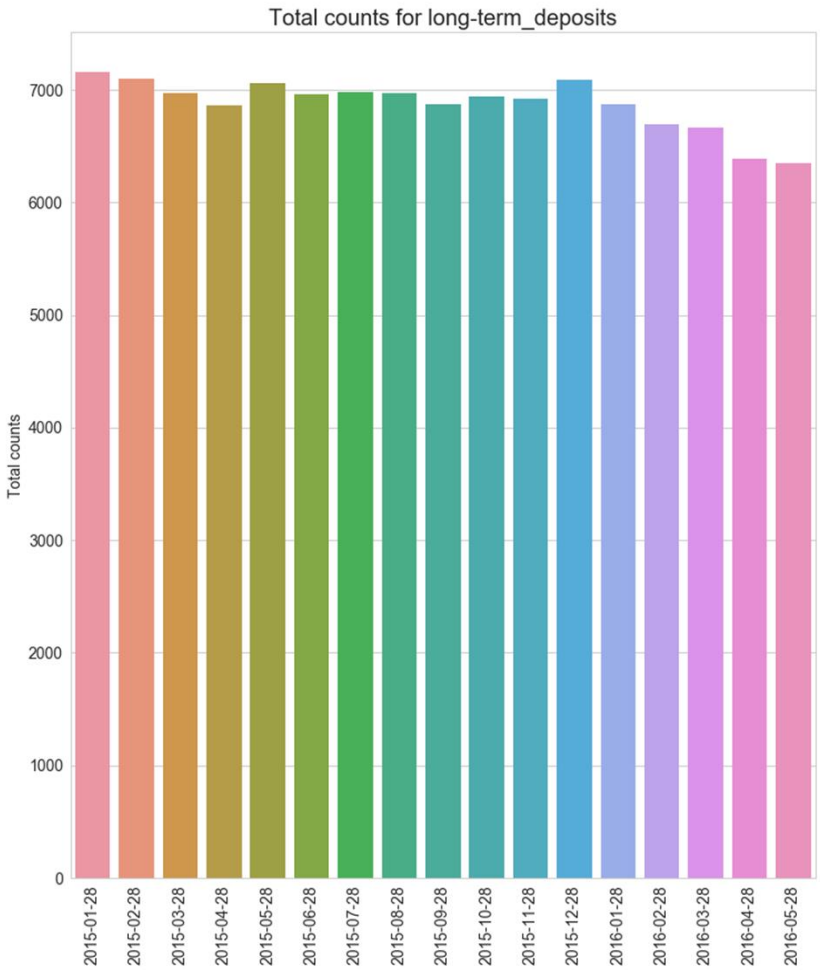


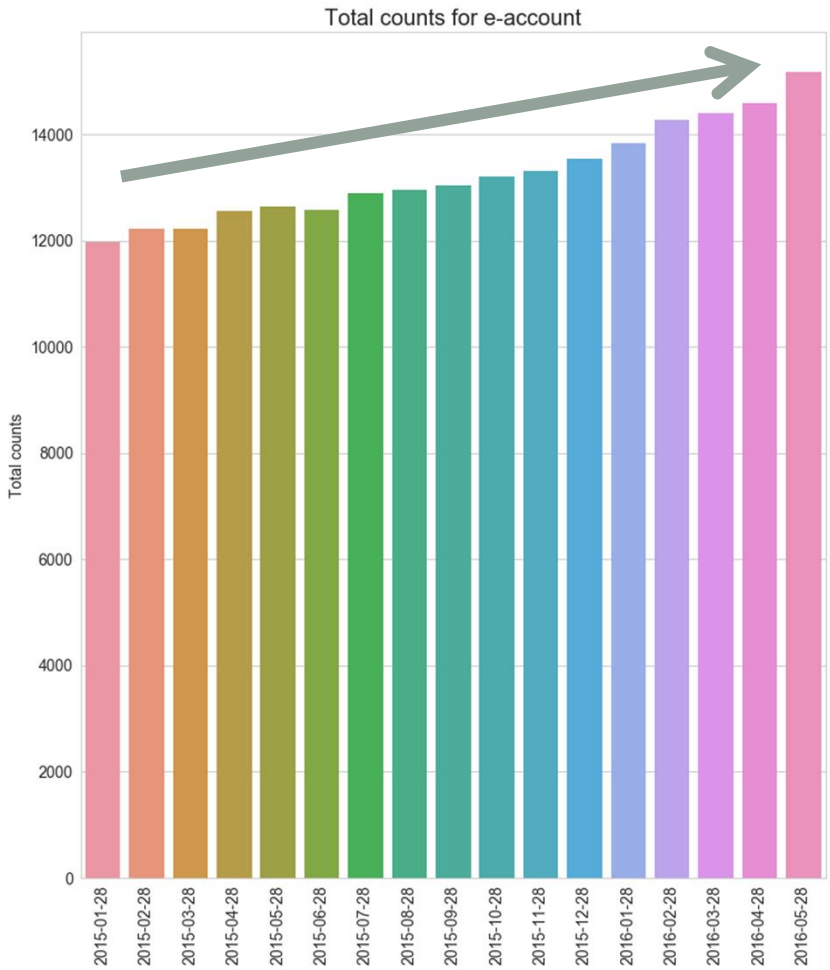
Total counts for short-term and medium-term deposits accounts





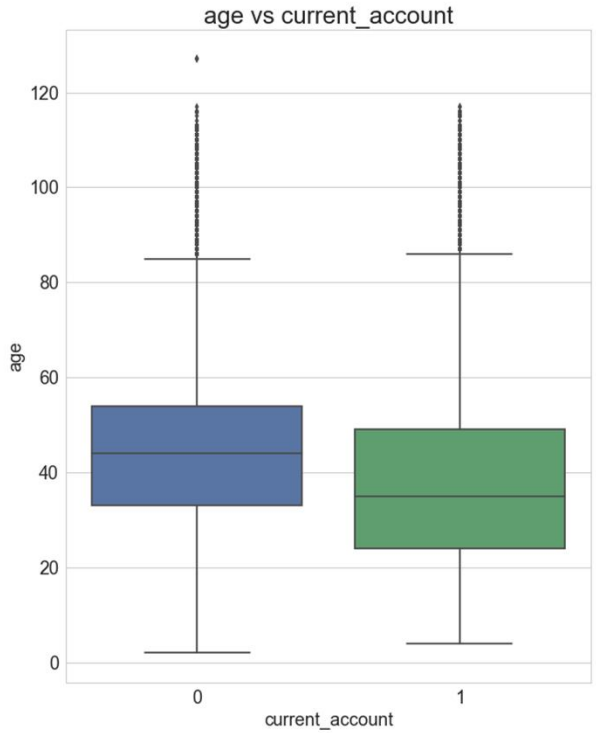
Total counts for long-term deposits and e-account accounts

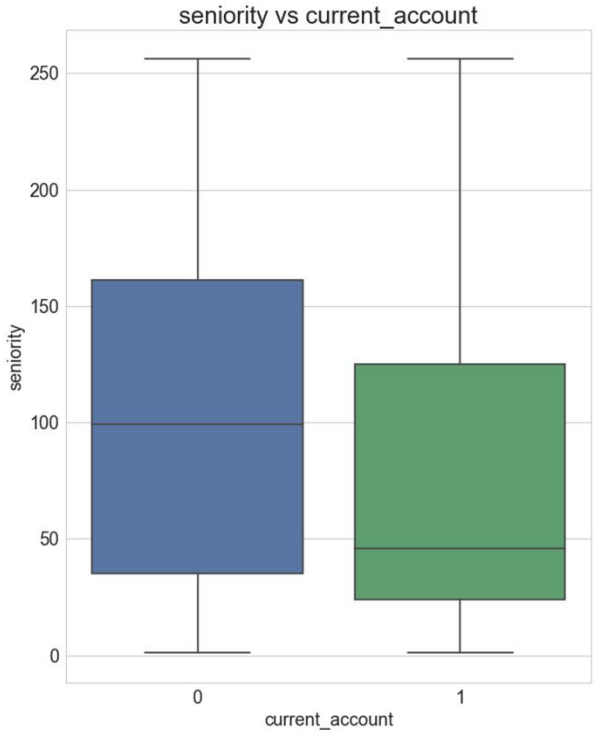


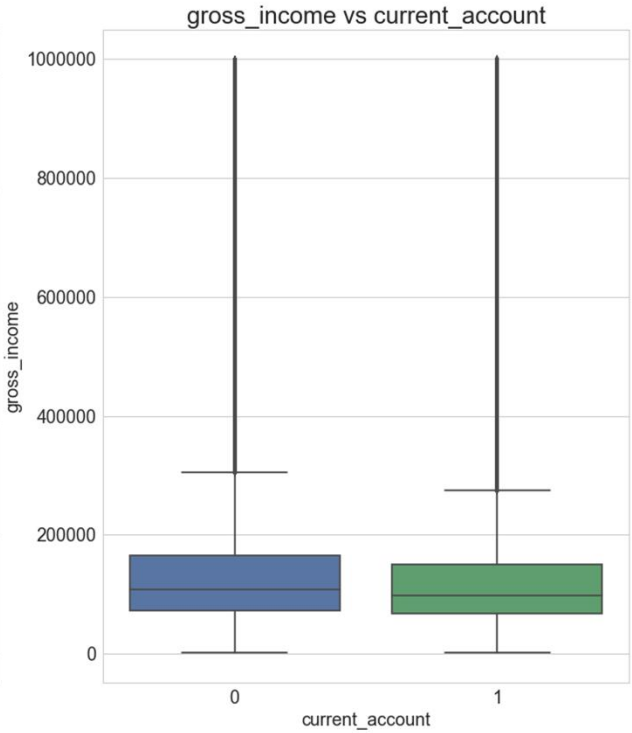


Age, seniority and gross income against current account

Plot data with gross income < 1M so clarity for gross income box plot is better.

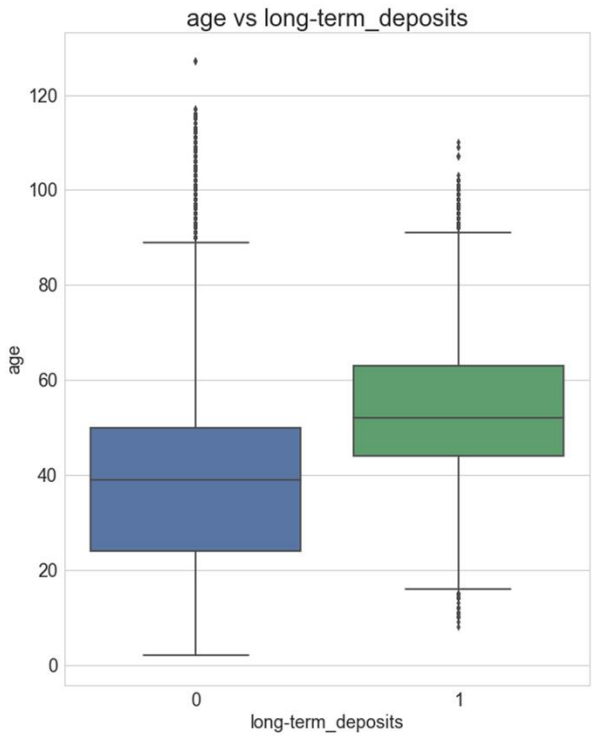


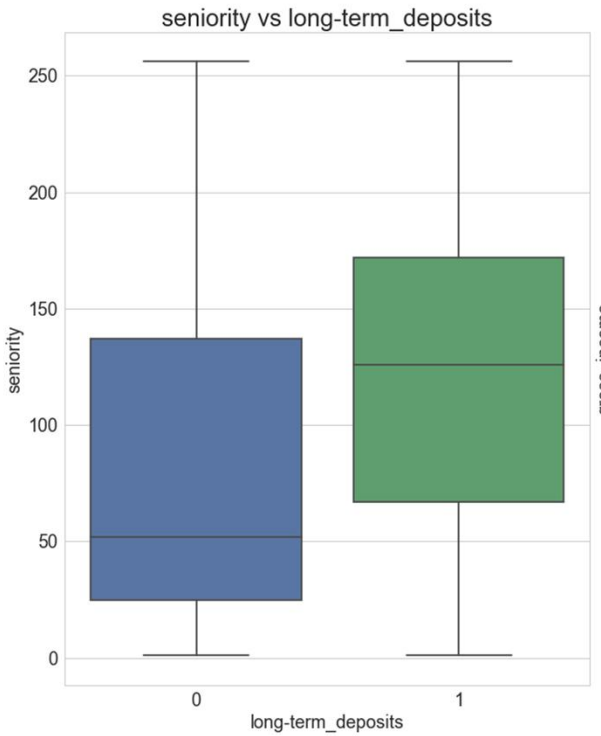


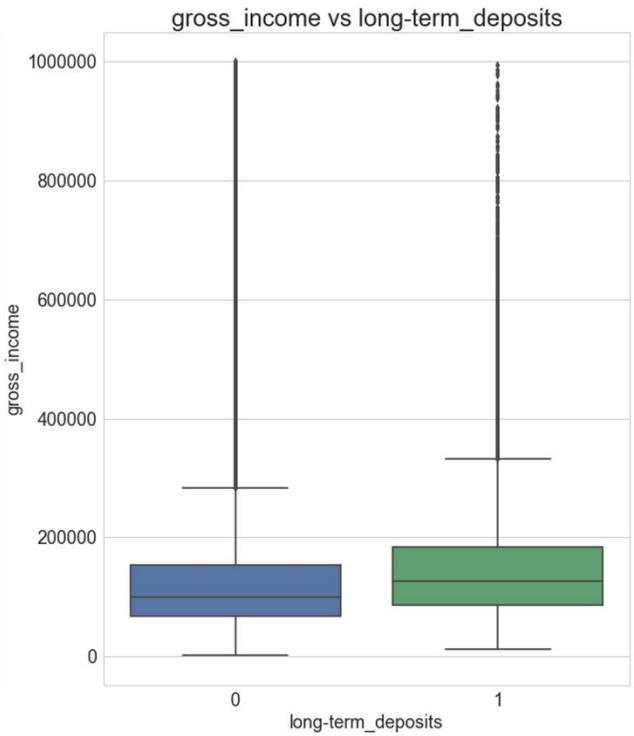


Differences between with account (1) and without account (0).
Age and seniority tend to be smaller for current account holders.

Age, seniority and gross income against long-term deposits account

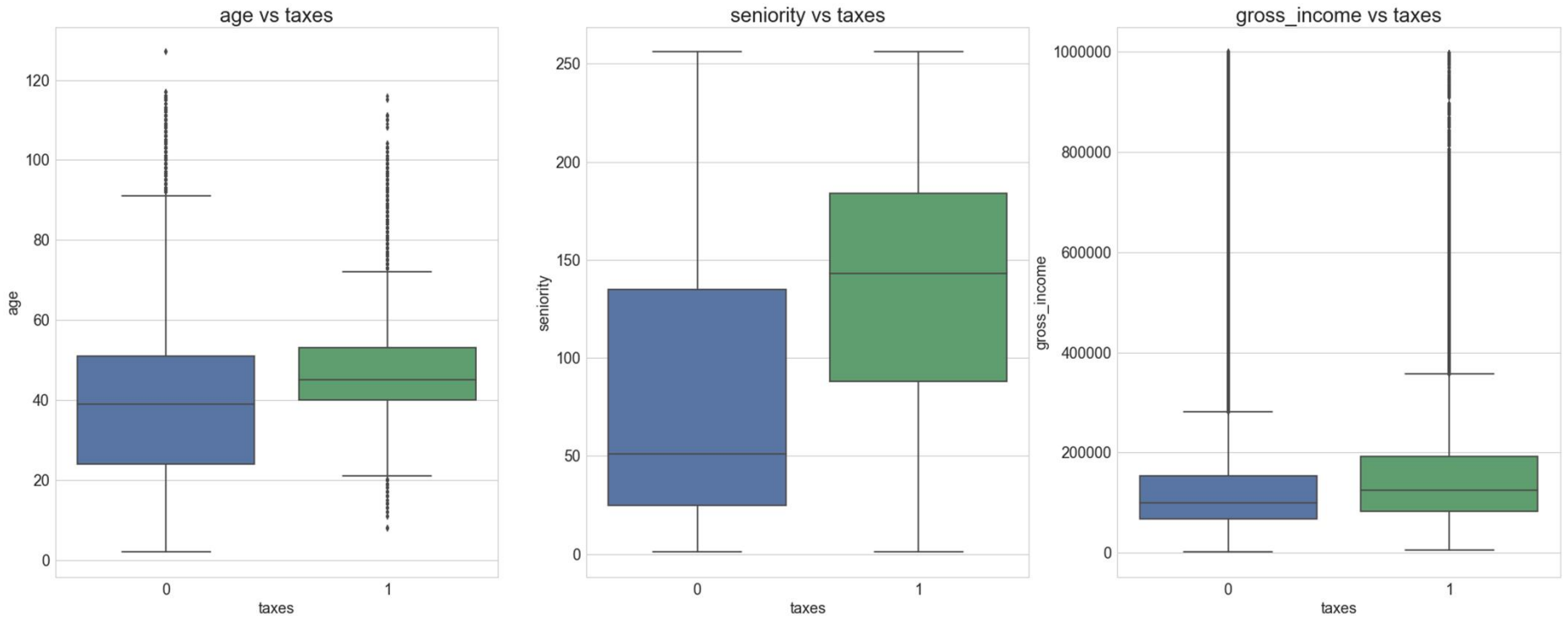






Differences between with account (1) and without account (0).
Age and seniority tend to be larger for long-term deposits account holders.

Age, seniority and gross income against taxes account

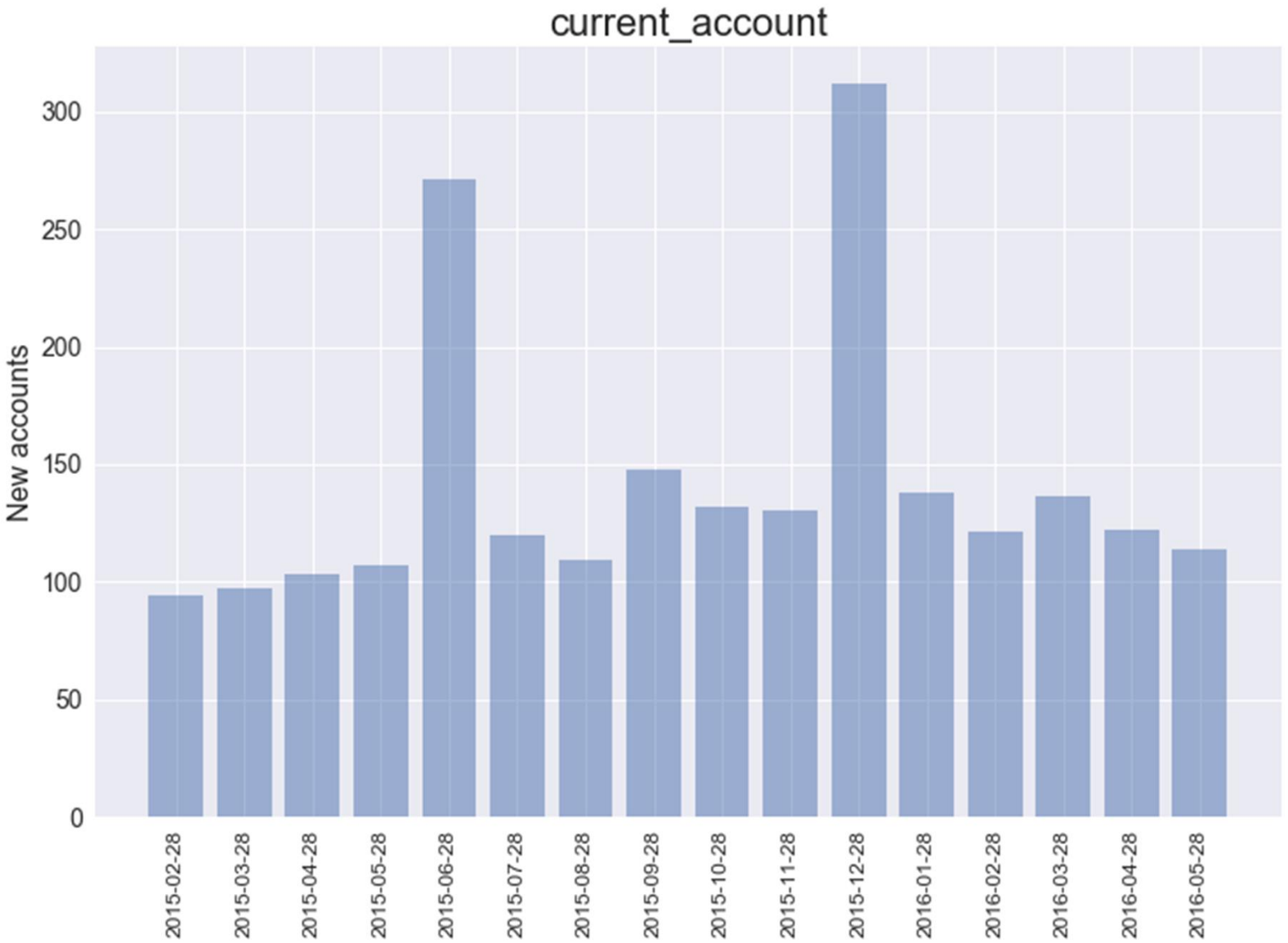


Differences between with account (1) and without account (0). Data has predicting power.

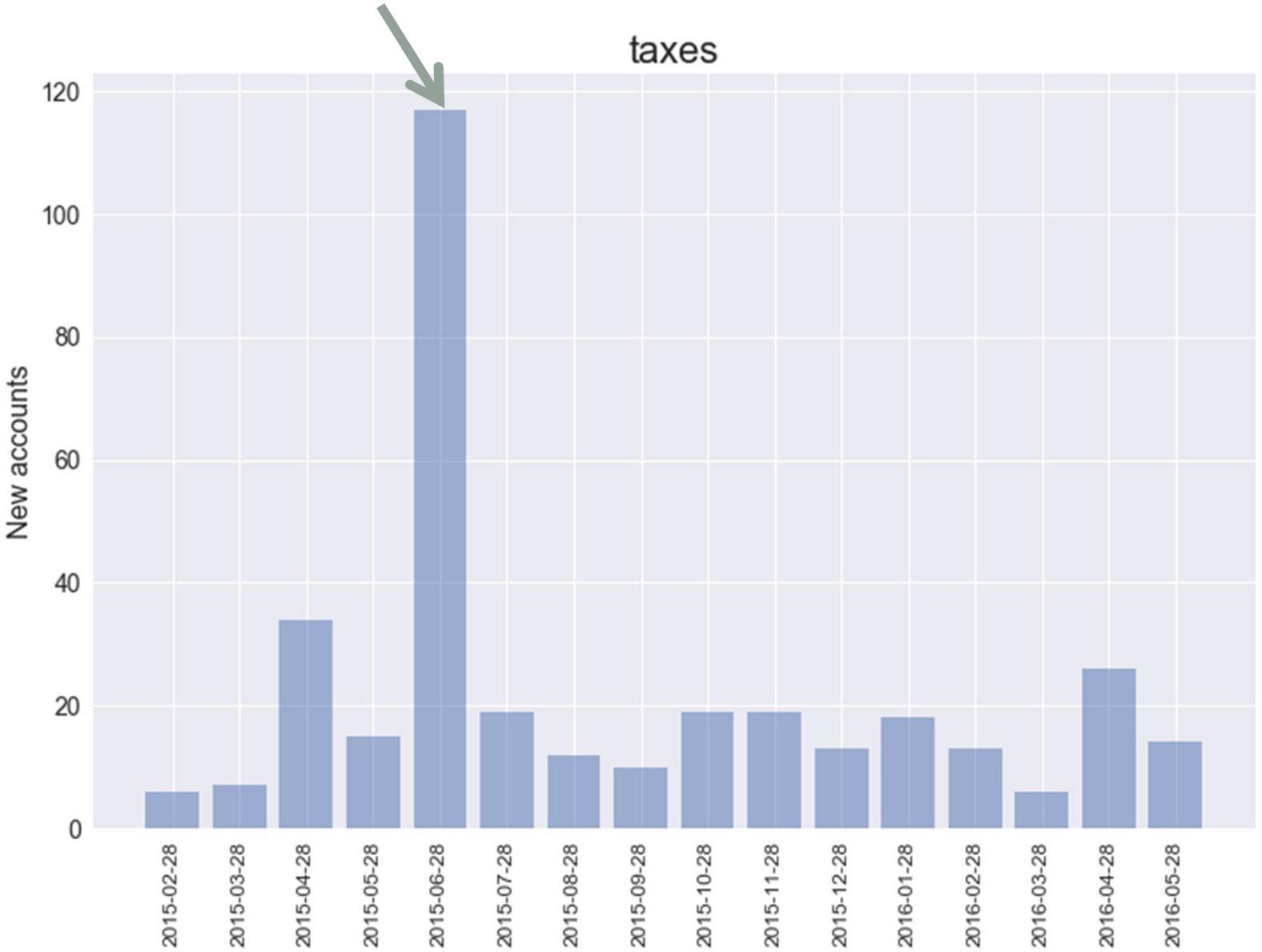
Find new accounts

- How many new accounts are added comparing to the previous month based on the same population.
- Compute for the same customers, given that they do not hold account previous month, how many of them add that account in the following month.
- How about the time lag for the new accounts?
- How about new customers with new accounts?

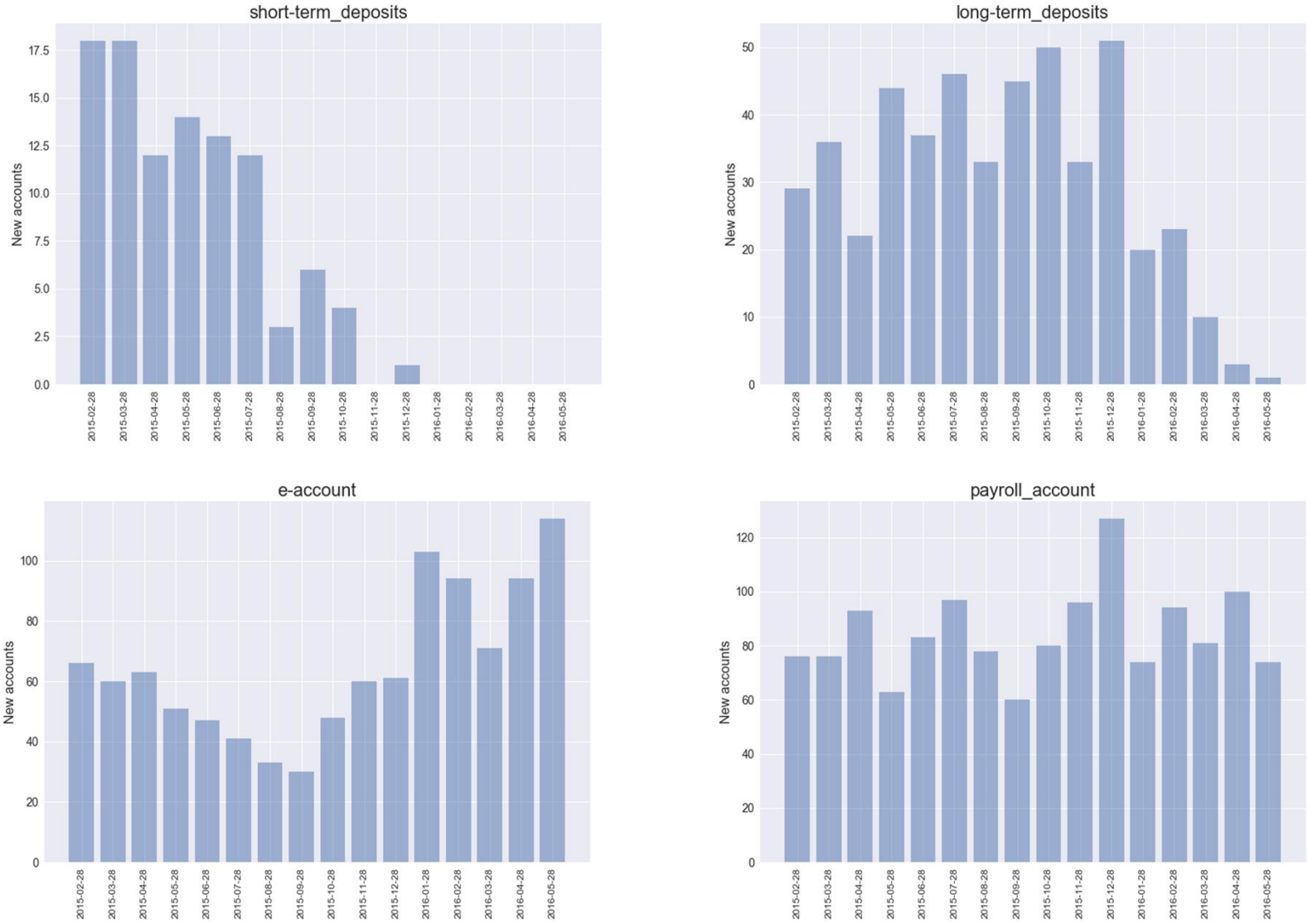
New accounts for the same population
Two peaks at 2015-06-28 and 2015-12-28.



A remarkable peak at 2015-06-28 for the taxes account.
(June marks the end of tax year for Spain)



Time series of new accounts



A increasing trend of new accounts for e-account.

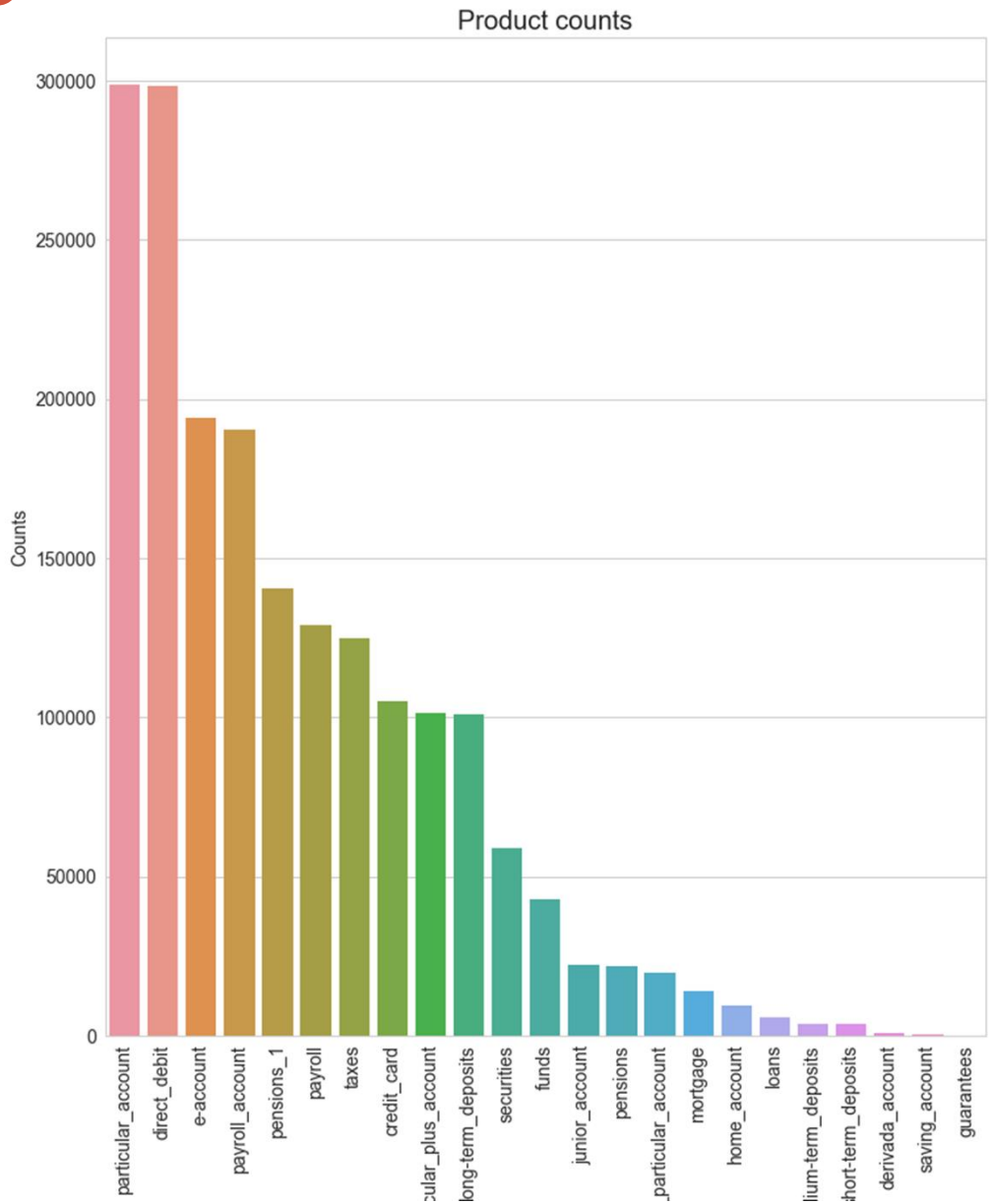
Summary for new accounts analyses

- There is a remarkable peak at 2015-06-28 for the taxes account.
- For the current account, we saw two peaks at 2015-06-28 and 2015-12-28.
- There is a increasing trend of new accounts for e-account and a decreasing trend for short-term deposits (no more new accounts in the year 2016).

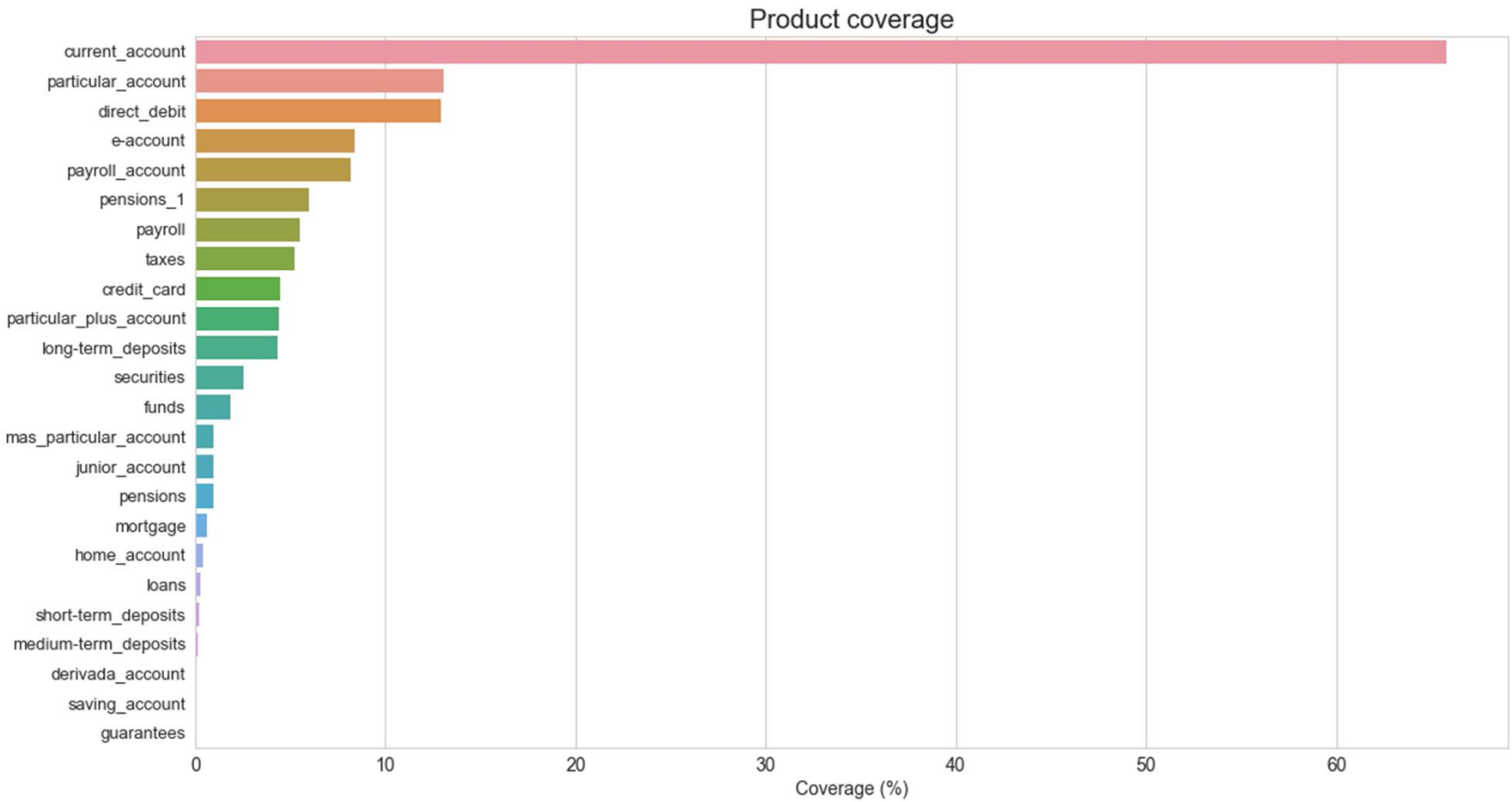
Product counts

Counts is calculated based on 20% of training data from 16 months.

Counts for current_account holders: 1,417,101



Product coverage
Coverage: percentage of account holder



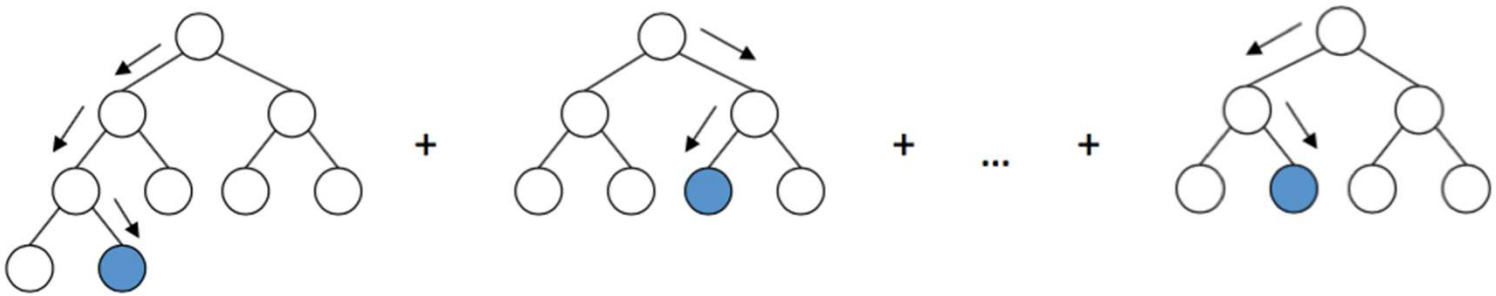
Product “current account” has the most coverage (~66%)

Build the base model

- Data preprocessing
  - Read data as an dictionary and save data as pandas data frame
  - Remove space and missing data
  - Check consistency in field values. Value “1.0” and “1” is the same
  - Check numerical fields only have numerical values. Age and seniority have non-numeric values.
  - Convert categorical data into numerical data using label encoder
  - Split data into training and test parts

Base model: Xgboost

1. Fit the model
2. Calculate training accuracy
3. Calculate training AUC
4. Calculate testing AUC
5. Plot feature importance



Base model performance

```
param = {}
seed_val=0
param['eta'] = 0.05
param['max_depth'] = 8
param['silent'] = 1
param['eval_metric'] = "auc"
param['min_child_weight'] = 1
param['subsample'] = 0.7
param['colsample_bytree'] = 0.7
param['seed'] = seed_val
param['objective']='binary:logistic'
```

Model Report
Accuracy (Train) : 0.7612
AUC Score (Train): 0.822797

Model evaluation on test data
Accuracy (Test) : 0.7608
AUC Score (Test): 0.820525

Parameter tuning vis grid search

Grid search parameter: max_depth and min_child_weight

```
cv_params = {'max_depth': [3,5,7], 'min_child_weight': [1,3,5]}
ind_params = {'learning_rate': 0.1, 'n_estimators': 1000, 'seed':0,
'subsample': 0.8, 'colsample_bytree': 0.8, 'objective': 'binary:logistic'}
```

Model with rank: 1
Mean validation score: 0.755 (std: 0.001)
Parameters: {'max_depth': 7, 'min_child_weight': 3}

Model with rank: 2
Mean validation score: 0.755 (std: 0.001)
Parameters: {'max_depth': 5, 'min_child_weight': 3}

Model with rank: 3
Mean validation score: 0.755 (std: 0.001)
Parameters: {'max_depth': 5, 'min_child_weight': 5}

Model score is insensitive to learning_rate and subsample

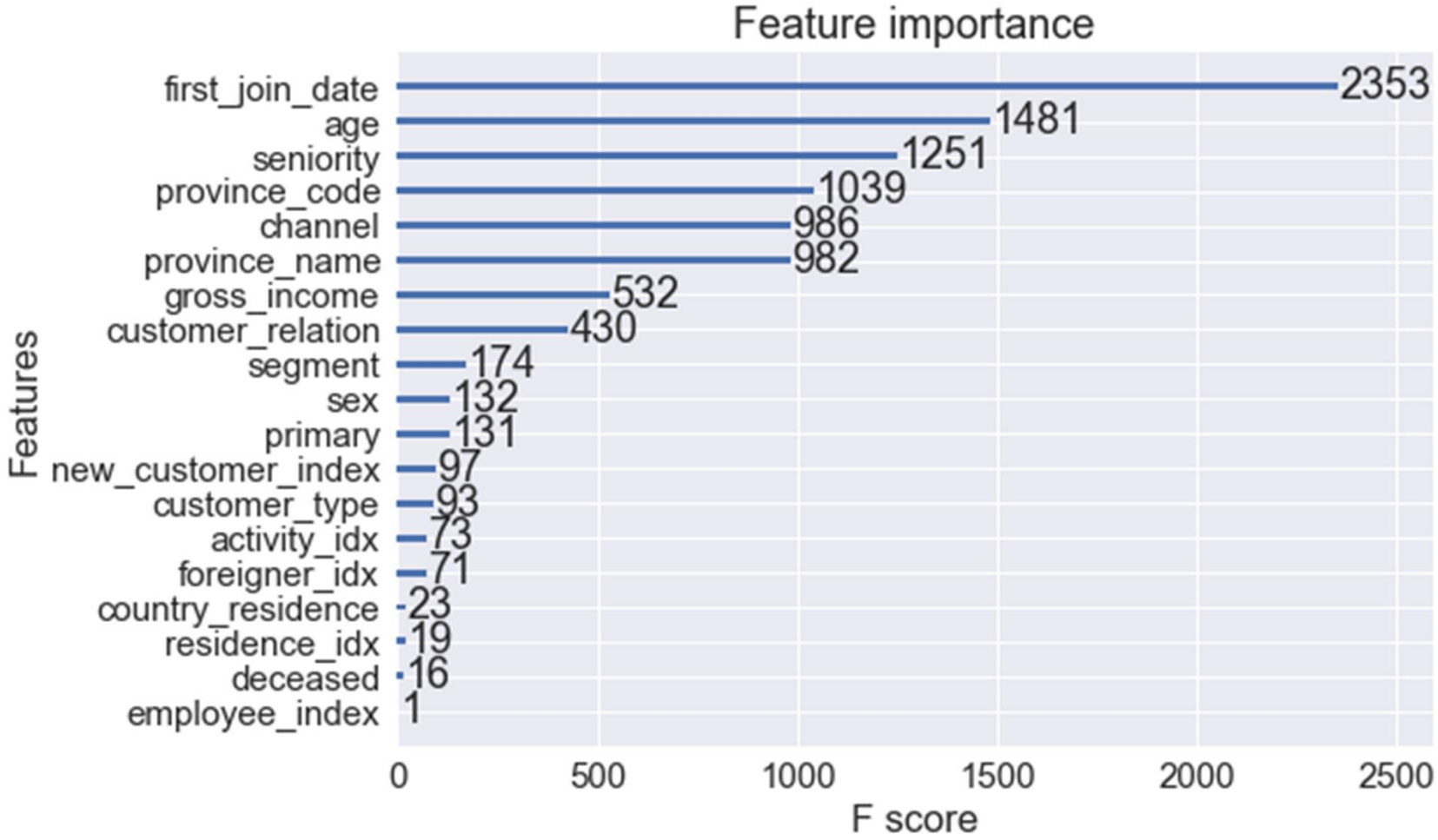
```
cv_params = {'learning_rate': [0.1, 0.01], 'subsample': [0.7,0.8,0.9]}
ind_params = {'n_estimators': 1000, 'seed':0, 'colsample_bytree': 0.8,
             'objective': 'binary:logistic', 'max_depth': 3, 'min_child_weight': 1}
```

Best score

mean: 0.75332, std: 0.00061, params: {'subsample': 0.7, 'learning_rate': 0.1},
mean: 0.75336, std: 0.00047, params: {'subsample': 0.8, 'learning_rate': 0.1},
mean: 0.75325, std: 0.00051, params: {'subsample': 0.9, 'learning_rate': 0.1},
mean: 0.74302, std: 0.00046, params: {'subsample': 0.7, 'learning_rate': 0.01},
mean: 0.74295, std: 0.00028, params: {'subsample': 0.8, 'learning_rate': 0.01},
mean: 0.74305, std: 0.00041, params: {'subsample': 0.9, 'learning_rate': 0.01}

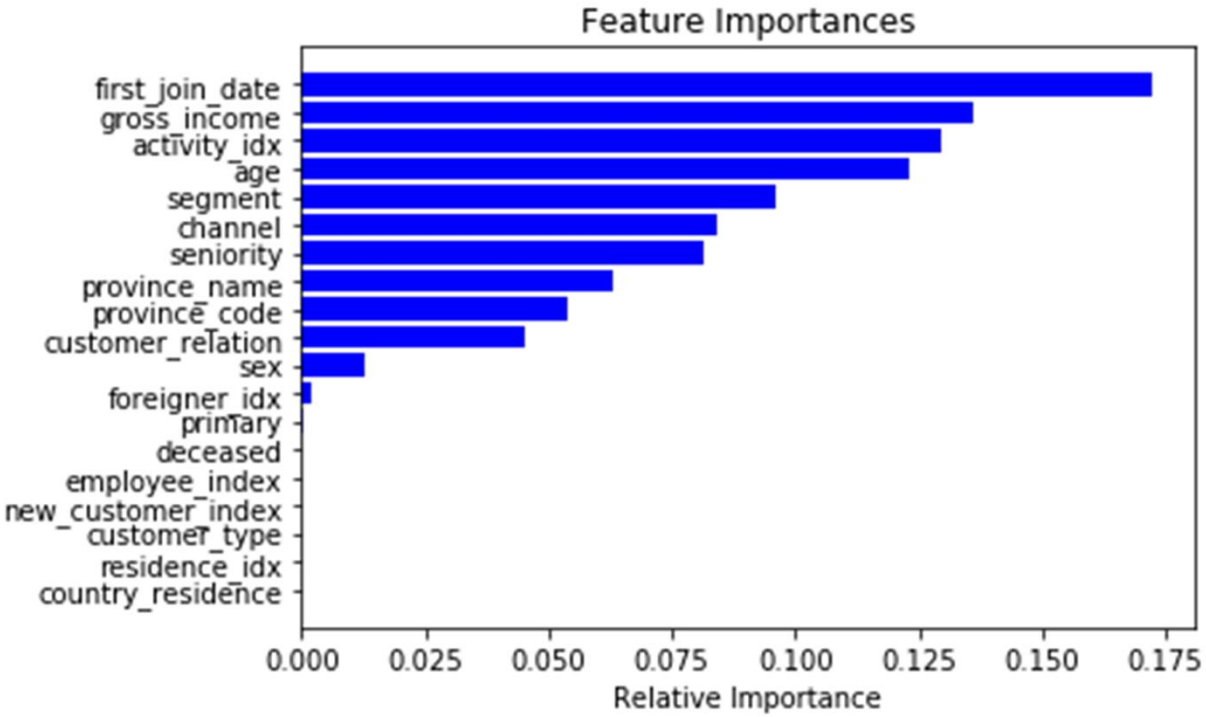
Feature importance

Quantify the most important features in the series of trees.



The ‘first_join_data’ feature seems to have the most importance. Age and seniority are also important. Employee index and deceased info are not as important.

Feature importance: random forest



Xgboost agrees with random forest model mostly in feature importance.

However, activity_idx receives higher importance value in random forest model but has much lower score in xgboost.

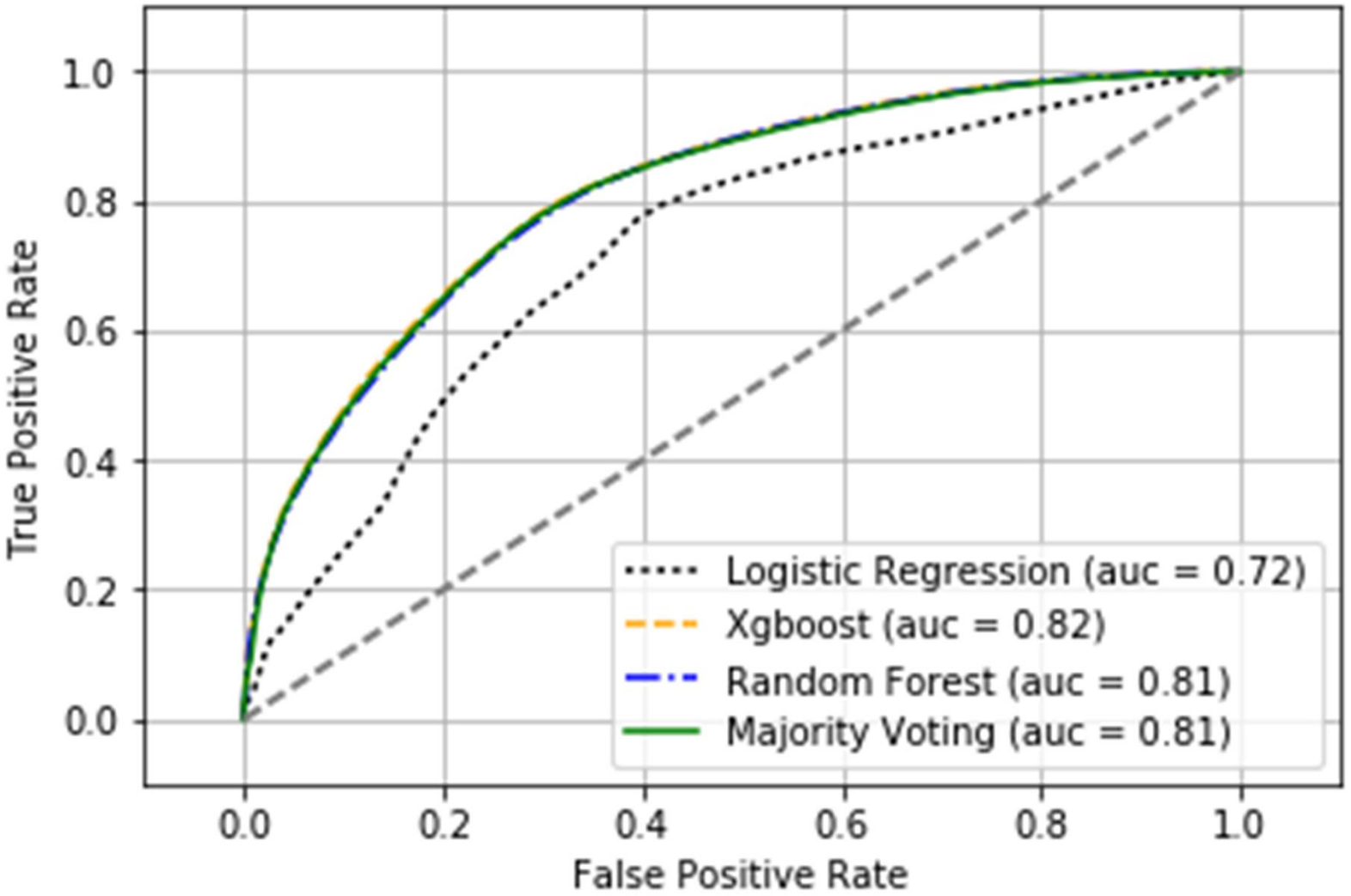
Ensemble Learning

- *Combining Different Models*
- *Use majority voting to predict the class label based on majority of classifiers*

Use (1) Logistic Regression, (2) Decision Tree (3) KNN
10-fold cross validation (training data):
ROC AUC: 0.73 (+/- 0.00) [Logistic Regression]
ROC AUC: 0.63 (+/- 0.00) [Decision Tree]
ROC AUC: 0.65 (+/- 0.00) [KNN]
ROC AUC: 0.74 (+/- 0.00) [Majority Voting]

Use (1) Logistic Regression, (2) xgboost (3) Random Forest
10-fold cross validation (training data):
ROC AUC: 0.73 (+/- 0.00) [Logistic Regression]
ROC AUC: 0.82 (+/- 0.00) [Xgboost]
ROC AUC: 0.81 (+/- 0.00) [Random Forest]
ROC AUC: 0.81 (+/- 0.00) [Majority Voting]

ROC curve


True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
Logistic Regression (auc = 0.72)
Xgboost (auc = 0.82)
Random Forest (auc = 0.81)
Majority Voting (auc = 0.81)

Summary for predictive modeling

- First joint day, age, and seniority being the most important features.
- Xgboost has the highest AUC score 0.82
- Next is random forest model, AUC score 0.81
  - Tree-base algorithm
- AUC value is insensitive to cross validation
  - Cross validation = 3, AUC ~ 0.82
  - Cross validation =10, AUC ~ 0.82
- Majority voting follows the best model

Summary

- Two peaks at 2015-06-28 and 2015-12-28 for new current accounts
- A remarkable peak at 2015-06-28 for the new tax accounts (June marks the end of the tax year for Spain)
- There is a increasing trend of new accounts for e-account and a decreasing trend for deposits.
- Feature against product plots show differences between 0 (without accounts) and 1 (with accounts). Data has predicting power.
- Tree-based algorithm outperform other models
  - Xgboost, random forest have the highest score
- Majority voting follows the best model

Usage of results

- Find potential customers
- Customers who own current accounts may be more interested in other types of accounts, so bank can provide recommendation to those customers
- For non-customers (who do not own current accounts), send advertisements to them
- Broadcast tax accounts when time is close the end of tax year

Ideas for improvement

- Analyze more on data
- Feature engineering
- Hyperparameter tuning for xgboost model
- Try different periods of training data
- Recommendation engine
- Predicting more products
- Try different algorithms
- Try penalized models

ACKNOWLEDGEMENT

Ikechukwu Okonkwo

XGBoost developers and contributors

scikit-learn developers

Anaconda

Springboard