Product Recommendation for Santander Bank
CSCI8360 final project

Team ChickenBurger

Mojtaba, Bahaa, Shawn, Priyanka and Yang

Online:

1) Introduction
2) Understanding Data
3) Data cleaning
4) Evaluation criteria: mAP@7 (mean Avg Precision)
5) Model building
    a) RandomForest
    b) LTSM
6) Results and future work
7) Q and As

Introductions to this problem:

1) Ongoing Kaggle competition
2) Santander bank wants us to predict which additional products will be purchased by their existing customers in the next month (June 2016) based on their past behavior (Jan 2015 till May 2016).
3) Understanding your data:
    a) Translate
    b) Rename attributes
    c) Understand schema

    https://docs.google.com/spreadsheets/d/1dgGBX0PICEc0PayOUWN6ZjGeTUERbeYfo6S3tczFfCM/edit#gid=0

Understanding your data:

Original:

Time indexed data:

2015-6-28, customer-id,~~~

2015-7-28, customer-id,~~~

heterogenous data:

a) Some customer start at the first, second,~~~ month
b) Different data type, string, int, float, timestamp, etc
c) Missing values: NA vs 0
d) Unbalanced labels for training

Data for model training:

Customer-id indexed data:

Customer-id-1,~~17 month data~~

Customer-id-2,~~17 month data~~

well-structured data

a) Fill -1 for missing customer month data, same size data for each customer
b) Unify data type
c) Fill missing data(N/A) as -1
d) Instead resampling, assign a LARGE value for our loss function for rare classes
e) Data cleaning part(next slide)

Data cleaning and Feature Engineering:

1) Remove irrelevant attributes:
   A) Province Name, since we have province-code; B) Time-Stamp (fecha_dato):2015-6-28
   C) indfall→ dead_customer (you do not want mess up with them! so R.I.P.)

2) Modify data:
   A) membership-end-data: 2015-11-30-> keep month number from 1970-1-1 (no days or second)
   a) not needed b) large numbers skew the distribution

   B) take a log of renta (income) ; after you take a log the incomes match a normal distribution which is make more sense to use)

Data Summary

Number of engineered features = 21 + 1(Id)

Number of products = 24 (each taking value 0/1 - sparse attributes) of 18th month

Number of month data available = 17 (not all customers)

Evaluation criteria: mAP@7 (mean Avg Precision)

1) Avg Precision: ap@n = sum (P(k)/min(m,n))   : m is the really value number
2) Example: (for one customer)

Real [1,2,3,0,0,0,0]	m=3, n=7 so min(m,n) = 3
pred1 [2,4,3,0,0,0,0]	ap@7=⅓ (1/1 + 0/2 + 2/3 + 0 + 0 + 0 + 0)=0.556
pred2 [2,3,4,0,0,0,0]	ap@7=⅓ (1/1 + 2/2 + 0 + 0 + 0 + 0 + 0)=0.667 (order matters)
pred3 [2,3,1,0,0,0,0]	ap@7=⅓ (1/1 + 2/2 + 3/3 + 0 + 0 + 0 +0) = 1 (max value)
pred4 [2,4,0,0,0,0,3]	ap@7=⅓(1/1 + 0 + 0 + 0 + 0 + 0 + 2/7) = 0.429 (again order matters)

3) Mean AP: add ap@7 for all customer then divided by the total number
4) Python code example:

https://github.com/benhamner/Metrics/blob/master/Python/ml_metrics/average_precision.py

mAP@7 for this project (continues)

1) Mean AP:

   The mean of AP@7 for each customer = (0 + 0 + 0.76+~~~)/1_Million

2) Assumed best result:

   ~ 0.0315 (only 3.15% customers out of ~1M buy new products in average over the 17 months)

3) Python code for calculating mAP:

https://github.com/benhamner/Metrics/blob/master/Python/ml_metrics/average_precision.py

Model building -introduction:

1) the problem is a Multiple-Classification problem with potential temporal correlated data
2) Possible models: classifiers :

    2.1) Non-temporal model: Randomforest, NN, etc

    2.2) Temporal model: LSTM

Naive approach - RandomForest pipeline

Trained on 5 mths data (from 2015-1-28 to 2015-5-28) (21+24)* 5 features

Kept the last 6th mth features into test set and predicted for the 18th month

Built 24 Random forests for each product (took probabilities /confidence for each product being recommended)

Aggregated the results

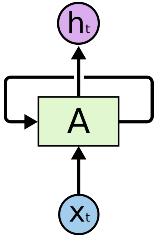
Reinitialized last month purchases as -1 for the 17th month to prevent recommending products that the customer already has.

Sort the recommended products by probability

Calculate MAP

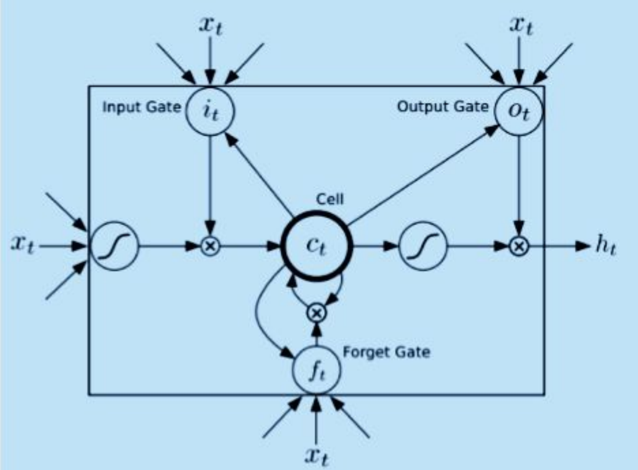
Better Approach: LSTM approach' building block

1) The NN has memory→ Recurrent neural networks(RNN):



2) A improved RNN→Long Short Term Memory nn:

LSTM can discuss how long, and what to remember during training



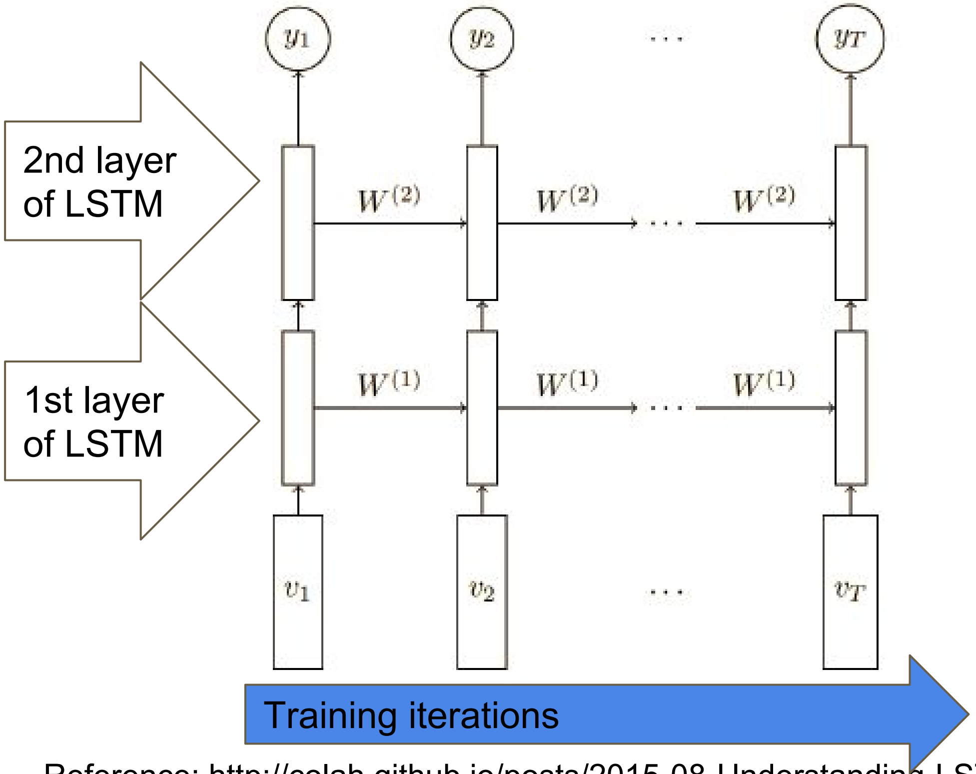
$$
\begin{aligned}
\mathbf{i}_t &= \sigma\left(W_{xi}\mathbf{x}_t + W_{hi}\mathbf{h}_{t-1} + W_{ci}\mathbf{c}_{t-1} + \mathbf{b}_i\right), \\
\mathbf{f}_t &= \sigma\left(W_{xf}\mathbf{x}_t + W_{hf}\mathbf{h}_{t-1} + W_{cf}\mathbf{c}_{t-1} + \mathbf{b}_f\right), \\
\mathbf{c}_t &= \mathbf{f}_t\mathbf{c}_{t-1} + \mathbf{i}_t \tanh\left(W_{xc}\mathbf{x}_t + W_{hc}\mathbf{h}_{t-1} + \mathbf{b}_c\right), \\
\mathbf{o}_t &= \sigma\left(W_{xo}\mathbf{x}_t + W_{ho}\mathbf{h}_{t-1} + W_{co}\mathbf{c}_t + \mathbf{b}_o\right), \\
\mathbf{h}_t &= \mathbf{o}_t \tanh(\mathbf{c}_t).
\end{aligned}
$$

Output h is related to 3 gates
(input, output, and forget Gate)

Reference: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM approach (continues): pipeline

1) Architecture and training dynamics:



1) vi-> inputs
   - v1: 1st mth data
   - v2: 2nd mth data
   - v3: 3rd mth data
2) yi-> outputs
   y1: predicted products for 2st mth
   y2: predicted products for 3nd mth

3) penalize more heavily towards the end of the sequence; network has seen the full profile

Reference: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Results:

Currently: we have both pipeline working, the best result we have now is 0.0209 by a simple network.

Comparing to the benchmark 0.00412, not bad. But we need to do better to lead in the leading board (1st place is 0.0309, 2nd is 0.0307)

Feature Selection

Recursive Feature elimination (RFE)