Report

Donggun Lee

1 Introduction and Related Work

Many companies need to predict and analyze customer churn in business. For customers who are expected to deviate, they will be rewarded and turned to remainder, and reducing the rewards for remaining customers will be the company's ideal investment plan. The project likewise intends to model customer churn prediction in games. The data form contains 28 days of user and character activity, and we use 28 days of information to predict whether the user will quit the game before 64 days or remain and pay for the time played.

2 Approach

I used various techniques to modeling churn prediction. In complex dataset, difficult relation would be expected between lots of features. Also existing neural network could be specialized in solving complex relationships. It is important that the problem is not just regression problem on day and price, the model should maximize the expectation profit. So I used score function which given by competition site. The main reason that I use score function is the process of optimizing for predict exact day and price is not exactly same as maximizing expected profit.

2.1 Method

2.1.1 Preprocessing

Basically, I aggregated most features on common keys, user id and day. Especially some feature can't be aggregated immediately. First, trade data don't have single user id. So I splited table by user id that sell item and buy item. And aggregate each table by user id and day respectively. And I add features called average item price and item amount which are easily calculated by existing data. And I use special skill that I soon to explain called numbering unique technique that can numbering multiple characteristic data to number. Finally I merge all features on key of (userid, day) and also add more features that is mean, max, mean, sum of each features. Then I get 28×164 size of samples that 164 features for 28 days. Numbering unique technique can transform the characterized data to number. First make ordered list for whole characterized data. And get make binary data expressed the existence of each character. For example '110100' means that user have a,b,d when list is [a,b,c,d,e,f]. And make list using binary data like [110100, 011011, 000001]. And sort the list by size like [000001, 011011, 110100]. And get index of the data in list. As mentioned before, the user that have class a,b,d get 2 because of binary data is '110100' and index is 2. Then it can transform all multiple

2.1.2 Network

I use neural networks called ResNet and EfficientNet. I used various version along to number of layers. 18, 34, 50 layers for ResNet and b1, b4 for EfficientNet. As below figures show the architecture of each network.

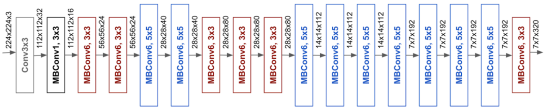

FIGURE 1 – The architecture of EfficientNet.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
		3×3 max pool, stride 2				
conv2_x	56×56	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

ures for ImageNet. Building blocks are shown in brackets (see also Fig. 5), with the numbers of block

FIGURE 2 – The architecture of ResNet.

2.1.3 Loss

I use both smooth L1 loss and CrossEntropy loss for first trial and both focal loss and CrossEntropy loss for second trial and score loss for final trial. I modify all zero to small values like 10^{-7}. Because when score is 0 the loss is 0 that caused problem of training in some samples. And I delete the first condition that actual prediction day equal to 64.

$$\begin{aligned} L_{1;smooth} &= \mid x \mid \quad if \quad \mid x \mid > \alpha \\ &= \frac{1}{\mid \alpha \mid} x^2 \quad if \quad \mid x \mid \leq \alpha \end{aligned}$$

Equation 1 -The smooth L1 loss. α is a hyper-parameter here and is usually taken as 1.

$$H(p,q) = -\sum_{x \in \mathcal{X}} p(x) \log q(x)$$

Equation 2 -The cross entropy loss.

Expected Revenue = Residual value × conversion rate(γ) - cost(C)
Residual value = Extra survival time(T) × Average daily payment(R)

$$\begin{aligned} T &= 0 \quad if \ \hat{t} = 64 \ or \ t = 64 \\ &= 30 \times \exp -\frac{(t-\hat{t})^2}{2 \times 15^2} \quad , otherwise \end{aligned}$$

$\hat{t}$: the predict value of survival time, t : the actual value of survival time

$$\begin{aligned} Cost(C) &= 0 \quad if \ \hat{t} = 64 \ or \ R = 0 \\ &= 0.01 \times 30 \times \hat{R} \quad , otherwise \end{aligned}$$

$\hat{R}$: the predict value of average daily payment

$$\begin{aligned} Conversion\ rate(\gamma) &= 0 \quad if \ \hat{C} < \frac{C_{opt}}{10} \ or \ C_{opt} = 0 \\ &= 1 \quad if \ \hat{C} \geq C_{opt} \\ &= \frac{10}{9}(\frac{\hat{C}}{C_{opt}} - 0.1) \quad , otherwise \end{aligned}$$

$\hat{C}$: the predict value of cost, C_{opt} : the proper cost (R=$\hat{R}$)

Equation 3 -The whole process of score loss(Expected Revenue is score).

2.2 Experimental Setting

I split 40000 training samples to 35000 and 5000 for each training set and validation set. The SGD optimizer is used for optimizing model with momentum 0.9 and weight decay for 5×10^{-4}. And scheduling learning rate for each 12, 25, 35, 45 epochs with multiplying by 0.2. And get validation score for each step and save models only in best score with early stopping to prevent overfitting. And I used random crop, random horizontal flip and random rotation.

3 Experimental Results

As below table, I evaluate each networks using different losses using score function in validation sets. Intuitively, the purpose of loss function is maximizing score that is expected revenue of each user. So using score function gave great improvement in validation score in each network. the best score is 10520 when using resnet34 with modified score loss.

Validation score	Smooth-L1 + CrossEntropy Loss	Smooth-L1 + focal loss	score loss	modified score loss
$ResNet18$	6152	7122	8621	9423
$ResNet34$	6232	7543	9275	**10520**
$ResNet50$	5864	7884	8765	9533
$EfficientNet-b1$	6866	6921	7751	8665
$EfficientNet-b4$	7981	7648	8355	8853

4 Conclusion

I used neural networks to maximize expected revenue of each user by predicting day and price. Preprocessing step is so important to make efficient and valuable dataset for training. I also proposed my special numbering skill called numbering unique technique. Also loss function is important that decide purpose of network that is well for what. Our purpose is maximizing expected revenue also given by score function. So I use score function and modified version too. It gave improvement in validation score. Also network called ResNet34 is best performed among various neural networks.