Telco Churn Prediction with Big Data

Yiqing Huang[1,2], Fangzhou Zhu[1,2], Mingxuan Yuan[3], Ke Deng[4], Yanhua Li[3], Bing Ni[3], Wenyuan Dai[3], Qiang Yang[3,5], Jia Zeng[1,2,3,*]
[1]School of Computer Science and Technology, Soochow University, Suzhou 215006, China
[2]Collaborative Innovation Center of Novel Software Technology and Industrialization
[3]Huawei Noah's Ark Lab, Hong Kong
[4]School of Computer Science and Information Technology, RMIT University, Australia
[5]Department of Computer Science and Engineering, Hong Kong University of Science and Technology
[*]Corresponding Author: zeng.jia@acm.org

ABSTRACT

We show that telco big data can make churn prediction much more easier from the 3V's perspectives: Volume, Variety, Velocity. Experimental results confirm that the prediction performance has been significantly improved by using a large volume of training data, a large variety of features from both business support systems (BSS) and operations support systems (OSS), and a high velocity of processing new coming data. We have deployed this churn prediction system in one of the biggest mobile operators in China. From millions of active customers, this system can provide a list of prepaid customers who are most likely to churn in the next month, having 0.96 precision for the top 50000 predicted churners in the list. Automatic matching retention campaigns with the targeted potential churners significantly boost their recharge rates, leading to a big business value.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; H.2.8 [**Database Management**]: Database Applications—*Data Mining*

Keywords

Telco churn prediction; big data; customer retention

1. INTRODUCTION

Customer churn is perhaps the biggest challenge in telco (telecommunication) industry. A churner quits the service provided by operators and yields no profit any longer. Figure 1 shows the churn rate (defined as the percentage of churners among all customers) in one of the biggest operators during 12 months in China. The prepaid customers has a significantly higher churn rate (on average 9.4%) than the postpaid customers (on average 5.2%), because the prepaid customers are not bound to contract and can quit easily without recharging. Since the cost of acquiring new customers is much higher than that of retaining the existing ones (e.g., around 3 times higher), it is urgent to build churn prediction systems to predict the most likely churners for proper retention campaigns. As far as millions of customers are concerned, reducing 1% churn rate will lead to a significant profit increase. Without loss of generality, this paper aims to develop and deploy an automatic churn prediction and retention system for prepaid customers, which has long been viewed as a more challenging task than churn prediction for postpaid customers [25, 29].

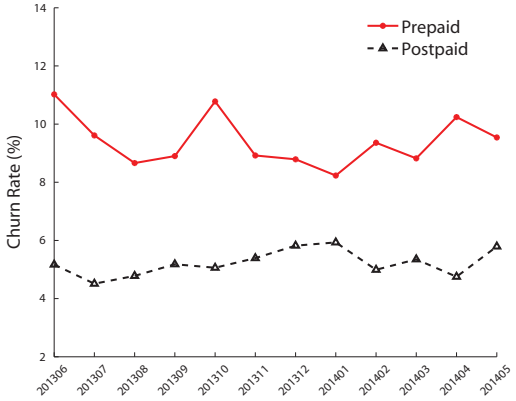

Figure 1: The churn rates in 12 months.

In this paper, we empirically demonstrate that telco big data make churn prediction much easier through 3V's perspectives. Practically, the recharge rate of potential churners has been greatly improved around 50%, achieving a big business value. For years of system building, telco data have very low inconsistencies and noises, so that Veracity is a natural property of telco data. In this sense, this study thereby covers 5V characteristics of telco big data: Volume, Variety, Velocity, Veracity and Value, in the context of churn prediction and retention systems. From extensive experiments, we continue to see at least marginal increases in predictive performance by using a larger Volume of training data, a more Variety of features from both telco business supporting systems (BSS) and operation supporting systems (OSS), and a higher Velocity of processing new coming data. For example, we achieve 0.96 precision on the top predicted 50000 churners (ranking by the churn likelihood) in the next month, which is significantly higher than the previous churn pre-

SIGMOD'15, May 31–June 4, 2015, Melbourne, Victoria, Australia.

http://dx.doi.org/10.1145/2723372.2742794

diction system deployed in this operator (0.68 precision) as well as the current state-of-the-art research work [16, 14, 13, 32, 1].[1] The results are based on the 9-month dataset from around two million prepaid customers in one of the biggest operators in China. This study provides a clear illustration that bigger data indeed can be more valuable assets for improving generalization ability of predictive modeling like churn prediction. Moreover, the results suggest that it is worthwhile for telco operators to gather both more data instances and more possible data features, plus the scalable computing platform to take advantage of them. In this sense, the *big data platform* plays a major role in the next-generation telco operation. As an additional contribution, we introduce how to integrate churn prediction model with retention campaign systems, automatically matching proper retention campaigns with potential churners. In particular, the churn prediction model and the feedback of retention campaign form a closed loop in feature engineering.

To summarize, this paper makes two main contributions. The primary contribution is an empirical demonstration that indeed churn prediction performance can be significantly improved with telco big data by integrating both BSS and OSS data. Although BSS data have been utilized in churn prediction very well in the past decade, we have shown that it is worthwhile collecting, storing and mining OSS data, which takes around 97% size of the entire telco data assets. This churn prediction system is one of the important components for the deployed telco big data platform in one of the biggest operators, China. The second contribution is the integration of churn prediction with retention campaign systems as a closed loop. After each campaign, we know which potential churners accept the retention offers, which can be used as class labels to build a multi-class classifier automatically matching proper offers with churners. This means that we can use a reasonable campaign cost to make the most profit. This paper thereby makes a small but important addition to the cumulative answer to a current open industrial question: How to monetize telco big data?

2. PREVIOUS WORK

In industry, data-driven churn predictive modeling generally includes constructing useful features (aka predictor variables) and building good classifiers (aka predictors) or classifier ensembles with these features [13, 36]. A binary classifier is induced from instances of training data, for which the class labels (churners or non-churners) are known. Using this classifier, we want to predict the class labels of instances of test data in the near future, where training and test data have no overlap in time intervals. Each instance typically is described by a vector of features which will be used for prediction. Following this research line, many classifiers have been adopted for churn prediction, including logistic regression [24, 14, 22, 25, 23], decision trees [34, 22, 3], boosting algorithms (e.g., variants of adaboost [18, 23]), boosted trees (gradient boosted decision trees) or random forest [21, 27], neural networks [9, 16], evolutionary computation (e.g., genetic algorithm and ant colony optimization) [9, 2, 33], ensemble of support vector machines [7, 33, 20], and ensemble of hybrid methods [34, 13]. Most customer behavior features are extracted from BSS, including call detailed records (call number, start time, data usage, etc.), billing records (account balance, payment records, average revenue per user called ARPU, etc.), demographic information (gender, birthday, career, home address, etc.), life cycle (new entry or in-net duration), package/handset (type and upgrade records, close to contract expiration, etc.), social networks (call graphs) [8, 19, 28], purchase history (subscribed services), complaint records [27], and customer levels (VIP or non-VIP). The churn management system becomes one of the key components in business intelligence (BI) systems.

[1]Although we cannot fairly compare our results with previous results due to different datasets, features and classifiers, we enhance the predictive performance over baselines that are adopted by most of previous research work.

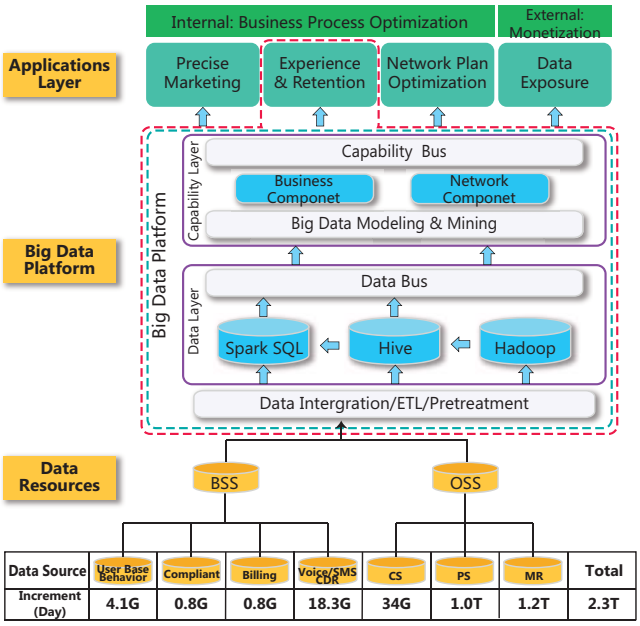

Figure 2: The overview of telco big data platform.

However, previous work has two limiting factors. First, sophisticated techniques have been applied in churn prediction for years, and it is getting harder to make a breakthrough to further lift the accuracy through the advances of prediction techniques. This motivates us to seek opportunities of telco big data from the 3V's perspectives: Volume, Variety, Velocity. An important, open question is: to what extent do larger volume, richer variety and higher speed of data actually lead to better predictive models? For example, although OSS data occupy around 97% size of the entire telco data assets, a variety of features from OSS have been rarely studied before, which reflect the quality of service (call drop rate and mobile internet speed), mobile search queries, and trajectory information [17]. Second, matching proper retention campaigns with potential churners has not been investigated to maximize the overall profit, causing a limited business value of existing churn prediction models [35, 32].

3. TELCO BIG DATA PLATFORM

The teleco operators create enormous amounts of data every day. BSS is the major IT components that a teleco operator uses to run its business operations towards customers. It supports four processes: product management, order management, revenue management and customer management. OSS is computer systems used by teleco service

providers to manage their networks (e.g., mobile networks). It supports management functions such as network inventory, service provisioning, network configuration and fault management. Note that both BSS and OSS often work separately, and have their own data and service responsibilities, With the rapid expansion, the telco data storage has moved to PB age, which requires a scalable big data computing platform for monetization.

Figure 2 overviews the functional architecture of the telco big data platform, where data sources are from both BSS and OSS. Generally, the data from BSS are composed of User Base/Behavior, Compliant, Billing and Voice/Message call detailed records (CDR) tables, which cover users' demographic information, package, call times, call duration, messages, recharge history, account balance, calling number, cell tower ID, and complaint records. The data volume in BSS is around 24GB per day. The data from OSS can be broadly categorized into three parts: circuit switch (CS), packet switch (PS) and measurement report (MR). CS data describe the call connection quality, e.g., call drop rate and call connection success rate. PS data are often called mobile broadband (MBB) data, including data gathered by probes using Deep Packet Inspection (DPI). PS data describe users' mobile web behaviors, which is related to web speed, connection success rate, and mobile search queries. MR data are from radio network controller (RNC), which can be used to estimate users' approximate trajectories [17]. The data volume in OSS is around 2.2TB per day, occupying over 97% data volume of the entire dataset. Also, we can use web crawler to obtain some Internet data (e.g., map information and social networks).

More specifically, BSS data are from the traditional BI systems widely used in telco operators, which consist of around 140 tables. OSS data are imported by the Huawei integrated solution called SmartCare, which collects the data from probes and interpret them to x-Detail Record (xDR) tables, such as User Flow Detail Record (UFDR), Transaction Detail Record (TDR), and Statistics Detail Record (SDR). All these tables have the key value called International Mobile Subscriber Identification Number (IMSI) or International Mobile Equipment Identity (IMEI), which can be used for joint operation and feature engineering. We use the multi-vendor data adaption module to change tables to the standard format, which are imported to big data platform through extract-transform-load (ETL) tools for data integration. We store these raw tables in Hadoop distributed file systems (HDFS), which communicates with Hive/Spark SQL for feature engineering purposes. These compose the data layer connected with the capability layer through the data bus in the big telco data platform. The main workload of data layer is to collect and update all tables from BSS and OSS periodically. In the capability layer, we build two types of models, business and network components, based on both unlabeled and labeled instances of features using data mining and machine learning algorithms. Using capability bus, these models can support application layer including internal (e.g., precise marketing, experience & retention and network plan optimization) and external applications (e.g., data exposure for trading). The main workloads of application layer include customer insight (e.g., churn, complaint, recommendation products or services) and network insight (e.g., network planning and optimization). The churn prediction system is supported by the telco big data platform, and is located at the red dashed lines in Figure 2.

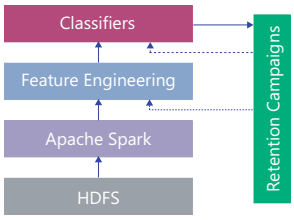

Figure 3: Churn prediction and retention systems.

Efficient management of telco big data for predictive modeling poses a great challenge to the computing platform [11]. For example, there are around 2.3TB new coming data per day from both BSS and OSS sources, where more than 97% volume of data is from OSS. The empirical results are based on a scalable churn prediction and retention system using Apache Hadoop [31] and Spark [37] distributed architectures including three key components: 1) data gathering/integration, 2) feature engineering/classifiers, and 3) retention campaign systems. In the data gathering and integration, we move data tables from different sources and integrate some tables by ETL tools. Most data are stored in regular tables (e.g., billing and demographic information) or sparse matrix (e.g., unstructured information like textual complaints, mobile search queries and trajectories) in HDFS. The feature engineering and classification components are hand coded in Spark, which is based on Hive/Spark SQL and some widely-used unsupervised/supervised learning algorithms, including PageRank [26], label propagation [40], topic models (latent Dirichlet allocation or LDA) [4, 39], LIBLINEAR (L-2 regularized logistic regression) [12], LIBFM (factorization machines) [30], gradient boosted decision tree (GBDT) and random forest (RF) [5]. This churn management platform has been deployed in the real-world operator's systems, which can scale up efficiently to massive data from both BSS and OSS.

4. PREDICTION AND RETENTION

The structure of the churn prediction system is illustrated in Figure 3. The HDFS and Apache Spark support the distributed storage and management of raw data. Feature engineering techniques are applied to select and extract relevant features for churn model training and prediction. In the business period (monthly), the churn classifier provides a list of likely churners for proper retention campaigns. In particular, the churn prediction model and the feedback of retention campaign form a closed-loop.

4.1 Feature Engineering

We make use of Hive and Spark SQL to quickly sanitize the data and extract the huge amount of useful features. The raw data are stored on HDFS as Hive tables. Because Hive runs much slower than Spark SQL, we use Spark SQL to manipulate the tables. Spark SQL is a new SQL engine designed from ground-up for Spark.[2] It brings native support for SQL to Spark and streamlines the process of querying data stored both in RDD (Resilient Distributed Dataset) and Hive [37]. The Hive tables are directly imported into

[2]`https://spark.apache.org/sql/`

Features	Description	Features	Description
localbase_outer_call_dur	duration of local-base outer call	ld_call_dur	duration of long-distance call
roam_call_dur	duration of roaming call	localbase_called_dur	duration of local-base called
ld_called_dur	duration of long-distance called	roam_called_dur	duration of roaming called
cm_dur	duration of calling China Mobile	ct_dur	duration of calling China Telecom
busy_call_dur	duration of calling in busy time	fest_call_dur	duration of calling in festival
sms_p2p_inner_mo_cnt	count of inner-net MO SMS	sms_p2p_other_mo_cnt	count of other MO SMS
sms_p2p_cm_mo_cnt	count of MO SMS with China Mobile	sms_p2p_ct_mo_cnt	count of MO SMS with China Telecom
sms_info_mo_cnt	count of information-on-demand MO SMS	sms_p2p_roam_int_mo_cnt	count of roaming international MO SMS
mms_p2p_inner_mo_cnt	count of inner-net Mo MMS	mms_p2p_other_mo_cnt	count of other MO MMS
mms_p2p_cm_mo_cnt	count of MO MMS with China Mobile	mms_p2p_ct_mo_cnt	count of MO MMS with China Telecom
mms_p2p_roam_int_mo_cnt	count of roaming international MO MMS	all_call_cnt	count of all call
voice_cnt	count of voice call	local_base_call_cnt	count of local-base call
ld_call_cnt	count of long-distance call	roam_call_cnt	count of roaming call
caller_cnt	count of caller call	voice_dur	duration of voice call
caller_dur	duration of caller call	localbase_inner_call_dur	duration of local-base inner-net call
free_call_dur	duration of free call	call_10010_cnt	count of call to 10010 (service number)
call_10010_manual_cnt	count of call to manual 10010	sms_bill_cnt	count of billing SMS
sms_p2p_mt_cnt	count of MT SMS	mms_cnt	count of all MMS
mms_p2p_mt_cnt	count of MT MMS	gprs_all_flux	all GPRS flux
gender	gender of user	age	age of user
credit_value	user credit value	innet_dura	duration of innet
total_charge	the total cost in a month	gprs_flux	the total GPRS flux
gprs_charge	the total cost of GPRS	local_call_minutes	minutes of local call
toll_call_minutes	minutes of long-distance call	roam_call_minutes	minutes of roaming call
voice_call_minutes	minutes of voice call	p2p_sms_mo_cnt	count of MO SMS
p2p_sms_mo_charge	cost of MO SMS	pspt_type	credentials type
is_shanghai	if user is shanghai resident or not	town_id	user town id
sale_id	selling area id	pagerank	user effect calculated with pagerank
product_id	product id	product_price	product price
product_knd	kind of product	gift_voice_call_dur	duration of gift voice call
gift_sms_mo_cnt	count of gift MO SMS	gift_flux_value	gift flux
distinct_serve_count	count of service SMS	serve_sms_count	count of service provider through SMS
innet_dura	duration in net	balance	account balance
balance_rate	recharge over account balance	total_charge	recharge value
PAGESIZE	page size	PAGE_SUCCEED_FLAG	page display success flag
L4_UL_THROUGHPUT	data upload speed	L4_DW_THROUGHPUT	data download speed
TCP_CONN_STATES	TCP connection status	TCP_RTT	TCP return time
STREAMING_FILESIZE	streaming file size	STREAMING_DW_PACKETS	streaming download packets
ALERTTIME	alert time	ENDTIME	alert end time
LAC	local area code	CI	cell ID

Figure 4: Some basic features in churn prediction.

Spark SQL for basic queries, including join queries and aggregation queries. For example, we need to join the local call table and the roam call table to combine the local call and roam call features. Also, we need to aggregate local call tables of different days to summarize a customer's call information in a time period (e.g., a month). All the intermediate results are stored as Hive tables, which can be reused by other tasks since the feature engineering may be repeated many times. Finally, a unified wide table is generated, where each tuple in the table represents a customer's feature vector. The wide table is exported into Hive to build classifiers. An example of some basic features in the wide table and explanations can be found in Figure 4.

These basic features can be broadly categorized into three parts: 1) baseline features, 2) CS features, and 3) PS features. The baseline features are extracted from BSS and are used in most previous research work, e.g., account balance, call frequency, call duration, complaint frequency, data usage, recharge amount, and so on. We use these baseline features to illustrate the differences between our solution and previous work. These features compose a vector, $\mathbf{x}_m = [x_1, \ldots, x_i, \ldots, x_j, \ldots, x_N]$, for each customer m.

Besides these basic features, we also use some unsupervised, semi-supervised and supervised learning algorithms to extract the following complicated features: 1) Graph-based Features, 2) Topic Features, and 3) Second-order Features.

4.1.1 CS and PS Features

Both CS and PS features are from OSS. We use several key performance indicators (KPI) and key quality indicators (KQI) in OSS as well as some statistical features, such as the average data upload/download speed and the most frequent connection locations from MR data. Due to the page limitation and the focus of this paper, we only give a brief introduction of the KPI/KQI features extracted from CS/PS data. Interested users can check Huawei's industrial OSS handbook for details.[3]

CS features represent the service quality of voice. The KPI/KQI features used in CS data includes average Perceived Call Success Rate, average E2E Call Connection Delay, average Perceived Call Drop Rate and average Voice Quality. Perceived Call Success Rate indicates the call success rate perceived by a calling customer. It is calculated from the formula, [Mobile Originated Call Alerting]/([Mobile Originated Call Attempt] + [Immediate Assignment Failures]/2). Each item in the formula is computed from a group of PIs (performance indicators). E2E Call Connection Delay indicates the amount of time a calling customer waits before hearing the ringback tone. It is defined as [E2E Call Connection Total Delay]/[Mobile Originated Call Alerting] = [Sum(Mobile Originated Connection Time - Mobile Originated Call Time)]/[Mobile Originated Call Alerting]. Perceived Call Drop Rate indicates the call drop rate perceived by a customer. It is obtained from [Radio Call Drop After Answer]/[Call Answer]. Voice quality include Uplink MOS (Mean Opinion Score), Downlink MOS, IP MOS, One-way Audio Count, Noise Count, and Echo Count. They can be used to assess the voice quality in the operator's network

[3]HUAWEI SEQ Analyst KQI&KPI Definition

and the customers' experience of voice services. Totally, each customer has a total of 9 CS KPI/KQI features.

PS KPI/KQI features reflect the quality of data service, which includes web, streaming and email. For example, we use the web features such as average Page Response Success Rate, average Page Response Delay, average Page Browsing Success Rate, average Page Browsing Delay, and average Page Download Throughput. Page Response Success Rate indicates the rate at which website access requests are successfully responded to after the customer types a Uniform Resource Locator (URL) in the address bar of a web browser. It is calculated from [Page Response Successes]/[Page Requests], where [Page Response Successes] = [First GET Successes] and [Page Requests] = [First GET Requests]. Page Response Delay indicates the amount of time a customer waits before the desired web page information starts to display in the title bar of a browser after the customer types a URL in the address bar. It is obtained from [Total Page Response Success Delay]/[Page Response Successes], where [Page Response Success Delay] = [First GET Response Success Delay] + [First TCP Connect Success Delay]. Similar KPI/KQI features are used for both streaming and email services. For each customer, we also select top 5 most frequent locations or stay areas during a period (e.g., a month) represented by latitude and longitude from MR data. Therefore, each customer has 15 PS KPI/KQI features plus 10 most frequent location features from PS data.

4.1.2 Graph-based Features

Graph-based features are extracted from the call graph, message graph and co-occurrence graph, based on CDR and MR data. All are undirected graphs with nodes for customers. The edge weight of call and message graphs are the accumulated mutual calling time and the total number of messages between two customers in a fixed period (e.g., a month) from CDR data. The edge weight of co-occurrence graph is the number of co-occurrences of two customers within a certain spatiotemporal cube (e.g., within 20 minute and 100×100 meter cube) from MR data. Churners will propagate their information through these graphs.

We use Hive/Spark SQL to generate the above undirected graphs represented by the edge-based sparse matrix, $\mathcal{E} = \{w_{m,n} \neq 0\}$, where $w_{m,n} \neq 0$ is the edge weight for mth and nth vertices (customers), and there are a total of $1 \leq m, n \leq N$ customers. Based on the undirected graphs, we use PageRank [26] and label propagation [40] algorithms to produce two features for each graph. The weighted PageRank feature x_m on undirected graphs is calculated as follows,

$$x_m = \frac{(1-d)}{N} + d \sum_{n \in \mathcal{N}(m)} \frac{x_n w_{m,n}}{\sum_{n \in \mathcal{N}(m)} w_{m,n}}, \qquad (1)$$

where $\mathcal{N}(m)$ is the set of neighboring customers having edges with m. The damping factor d is set to 0.85 practically. The initial value of $x_m = 1$. After several iterations of Eq. (1), x_m will converge to a fixed point due to random walk nature of the PageRank algorithm. The higher value of x_m corresponds to the higher importance in the graph. For example, in the call graph, the higher x_m of a customer means that more customers call (or called) this customer. Intuitively, customers with higher x_m have a low likelihood to churn. Each customer has a total of 3 PageRank features on 3 undirected graphs.

In label propagation, the basic idea is that we start with a group of customers that we have known that they are churners. Following the edge weights in the graph, we propagate the churner probabilities from the customer seed vertex (the ones we have churner label information about) into the customers without churner labels. After several iterations, the label propagation algorithm converges and the output is churner labels for the non-churner customers. Given the edge weight matrix $W_{M \times M}$, and a churner label probability matrix $Y_{M \times 2}$, the label propagation algorithm follows the 3 iterative steps:

1. $Y \leftarrow WY$;
2. Row-normalize Y to sum to 1;
3. Fix the churner labels in Y, Repeat from Step 1 until Y converges.

After label propagation, each customer (non-churner) is associated with a churn label probability y_m in matrix Y propagated from the churners in the previous month. The higher value means that this non-churner has a higher likelihood to be the churner in the graph. Since we have 3 undirected graphs, we obtain 3 churner label propagation features y_m for each customer m. As a result, we have 6 graph-based features for each customer plus 3 PageRank features, which are different and independent from label propagation features because PageRank features do not involve churner labels.

4.1.3 Topic Features

There are lots of text logs of customer complaints or search queries. In a certain period (e.g., a month), each customer can be represented as a document containing a bag of words of their complaint or search text. After removing less frequent words, we form 2408 and 15974 vocabulary words from complaint and search text. Because the word vector space is too sparse and high-dimensional to build a RF classifier, we choose the probabilistic topic modeling algorithm called LDA [4] to obtain compact and lower-dimensional topic features. We use a sparse matrix $\mathbf{x}_{W \times M}$ to represent the bag-of-word representation of all customers, where there are $1 \leq m \leq M$ customers and $1 \leq w \leq W$ vocabulary words. LDA allocates a set of thematic topic labels, $\mathbf{z} = \{z^k_{w,m}\}$, to explain non-zero elements in the customer-word co-occurrence matrix $\mathbf{x}_{W \times M} = \{x_{w,m}\}$, where $1 \leq w \leq W$ denotes the word index in the vocabulary, $1 \leq m \leq M$ denotes the document index, and $1 \leq k \leq K$ denotes the topic index. Usually, the number of topics K is provided by users. The nonzero element $x_{w,m} \neq 0$ denotes the number of word counts at the index $\{w, m\}$. The objective of LDA inference algorithms is to infer posterior probability from the full joint probability $p(\mathbf{x}, \mathbf{z}, \boldsymbol{\theta}, \boldsymbol{\phi}|\alpha, \beta)$, where $\mathbf{z}$ is the topic labeling configuration, $\boldsymbol{\theta}_{K \times M}$ and $\boldsymbol{\phi}_{K \times W}$ are two non-negative matrices of multinomial parameters for document-topic and topic-word distributions, satisfying $\sum_k \theta_m(k) = 1$ and $\sum_w \phi_w(k) = 1$. Both multinomial matrices are generated by two Dirichlet distributions with hyperparameters α and β. For simplicity, we consider the smoothed LDA with fixed symmetric hyperparameters.

We use a coordinate descent (CD) method called belief propagation (BP) [38, 39] to maximize the posterior probability of LDA,

$$p(\boldsymbol{\theta}, \boldsymbol{\phi}|\mathbf{x}, \alpha, \beta) = \frac{p(\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}|\alpha, \beta)}{p(\mathbf{x}|\alpha, \beta)} \propto p(\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}|\alpha, \beta). \qquad (2)$$

The output of LDA contains two matrices $\{\boldsymbol{\theta}, \boldsymbol{\phi}\}$. In practice, the number of topics K in LDA is set to 10. The matrix $\boldsymbol{\theta}_{K\times M}$ is the low-dimensional topic features for all customers, when compared with the original sparse high-dimensional matrix $\mathbf{x}_{W\times M}$ because $K \ll D$. For both complaint and search texts, we generate $K = 10$ dimensional topic features of each customer, respectively. So, there are a total of 20 topic features for each customer.

4.1.4 Second-order Features

In the feature vector, $\mathbf{x}_m = [x_1, \ldots, x_i, \ldots, x_j, \ldots, x_N]$, second-order features are defined as the product of any two feature values $x_i x_j$, which may help us find the hidden relationship between a specific pair of features. Suppose that there are N features, the total number of second-order features could be $(N+1)N/2$, which is often a burden to build a classifier. So, we use LIBFM [30] regression model to select the most useful second-order features. The regression model can be used for classification problems by setting the response of regression as class labels. The LIBFM optimizes the following objective function by the stochastic gradient descent algorithm,

$$y = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle \mathbf{v}_i, \mathbf{v}_j \rangle x_i x_j, \tag{3}$$

where w_0 and w_i are weight parameters, and $\mathbf{v}_i$ is a K-length vector for ith feature value. The inner product $\langle \mathbf{v}_i, \mathbf{v}_j \rangle$ denotes the weight for the second-order feature $x_i x_j$. The larger weight means that the second-order feature is more important [30]. So, we rank the learned weight $\langle \mathbf{v}_i, \mathbf{v}_j \rangle$ after optimizing (3), and select 20 second-order features with the top largest weights for churn prediction.

4.2 Classifiers

For experiments we report, we choose the Random Forest (RF) classifier [5] to make predictions. The major reason of this choice is that RF yields the best predictive performance among several widely-used classifiers (See comparisons in Subsection 5.8). RF is a supervised learning method that uses the bootstrap aggregating (bagging) technique for an ensemble of decision trees. Given a set of training instances, $\mathbf{x}_m = [x_1, \ldots, x_i, \ldots, x_j, \ldots, x_N]$, with class labels $y_m = \{\text{non-churner} = 0, \text{churner} = 1\}$, RF fits a set of decision trees $f_t, 1 \le t \le T$ to the random subspace of features. The label prediction for test instance $\mathbf{x}$ is the average of the predictions from all individual trees,

$$y = \frac{1}{T} \sum_{t=1}^{T} f_t(\mathbf{x}), \tag{4}$$

where y is the likelihood of being a churner. We rank this likelihood in descending order for predictive performance evaluation as well as for retention campaigns. Generally, customers with higher likelihood have lower recharge rates in our experiments.

For each decision tree, it randomly selects a subset of $\sqrt{N}$ from N features, and builds a splitting node by iterating these features. The Gini improvement $I(\cdot)$ is used to determine the best splitting point given ith column of features from $\mathbf{x}_{M\times N}$,

$$I(\mathbf{x}_{1:M,i}) = G(\mathbf{x}_{1:M,i}) - q \times G(\mathbf{x}_{1:P,i}) - (1-q) \times G(\mathbf{x}_{P+1:M,i}), \tag{5}$$

$$G(\cdot) = 1 - \sum_{c=1}^{2} p_c^2, \tag{6}$$

where P is the splitting point that partition $1:M$ instances into two parts $1:P$ and $P+1:M$, and q is the fraction of instances going left to the splitting point P. The function $G(\cdot)$ is the Gini index for a group of customer instances, and p_1 is the probability of churners and p_2 is the probability of non-churners in this group. For each column of features, we visit all splitting points P and find the maximum Gini improvement I. Then, we find the maximum I among all features, and use the feature with the maximum I as the splitting point in the node of a decision tree. Other parameters of RF are given as follows: 1) the number of trees is 500 and 2) the minimum samples in leaf nodes is 100. We fix the minimum samples in leaf nodes to 100 to avoid over-fitting. The split process of each decision tree will stop when the number of samples in an individual node is less than 100.

After training RF, we obtain each feature's importance value by summing the Gini improvements of all nodes,

$$\text{Importance}_i = \sum_{t=1}^{T} \sum_{P_i \in \text{tree}_t} I(\mathbf{x}_{1:M,i}). \tag{7}$$

The feature importance ranking could help us evaluate which features are major factors in churn prediction.

4.3 Retention Systems

The retention systems for potential churners make a closed loop with churn prediction systems as shown in Figure 3. Telco operators are not only concerned with the potential churners, but also want to carry out effective retention campaigns to retain those potential churners for further profits. Generally, if a customer accept a retention offer, he/she will keep on using the operator's service for the next 5 months to get the 1/5 offer per month. Nevertheless, there are multiple retention offers and operators do not know which offers match a specific group of potential churners. For example, some potential churners will not accept any offer, some want more cashback, some want more flux, and some want more free voice minutes. Before the automatic retention system is deployed, operators often match offers with potential churners by domain knowledge, and the campaign results are not satisfactory. So, it is necessary to build an automatic retention systems for matching offers with churners.

We define this task as a multi-category classification problem. A potential churner $\mathbf{x}_m$ is classified into multiple categories $y_m = \{0, 1, 2, \ldots, C-1\}$, where $y_m = 0$ denotes that the potential churner will not accept any offer, and other values means different types of retention offers. The class labels (retention results) are accumulated after each retention campaign. We train a RF retention classifier in Subsection 4.2 to do multi-category classification as shown in Figure 3. The retention classifier is updated if retention campaign results are available, similar to the churn classifier. Moreover, we use the label propagation algorithm to propagate the campaign result label y_m on three undirected graphs in Subsection 4.1.2. These $3 \times C$ new features are added to the original churn prediction features for training

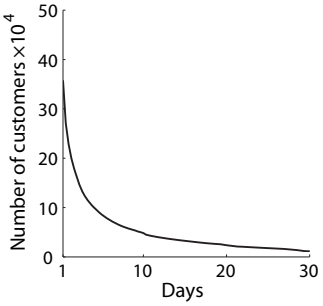

Figure 5: Distribution of the number of recharged customers in the recharge period.

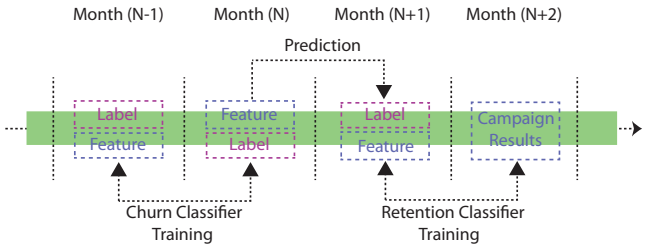

Figure 6: Experimental setting based on a sliding window of four months.

and classification purposes. The major advantage of the label propagation features in retention are that customers with close relationship tend to have similar retention offers. The campaign results feedback to the feature engineering layer as a closed loop, and the features are updated after each retention campaign as shown in Figure 3.

5. RESULTS

To build the churn prediction system, we need labeled data for training and evaluation purposes. Domain experts of telco operators guide us to label churners. If a prepaid customer in the recharge period does not recharge within 15 days, this customer is considered to be a churner. Because the customer can only receive calls in the recharge period (but cannot make any call out), it is convenient to carry out retention campaigns through calls and messages. This labeling rule helps predict potential churners at an early stage. In this sense, churn prediction is equivalent to predicting customers who will not recharge for more than 15 days in the recharge period in the next month. Figure 5 shows the distribution of the number of recharged customers in the recharge period in the past 9 months, where the x-axis denotes the number of days in recharge period and y-axis denotes the number of recharged customers. From customers in the recharge period, we see that less than 5% customers recharge beyond 15 days (in the next month), which confirms that customers have low likelihoods to recharge beyond 15 days (labeled churners).

Table 1 shows the statistics of the dataset collected from the past 9 consecutive months from 2013 to 2014. It shows the total number of customers, churners and non-churners using the above labeling rule. On average, the number of churners takes around 9.2% of the total number of customers in the dataset. It is interesting to see that though there are lots of churners, the total number of prepaid customers remains in almost the same level. This indicates that each month the number of acquired new prepaid customers is almost equal to the number of churners, forming a dynamic balance. Because acquiring new customers usually consumes three times higher cost than that of retaining potential churners, there is a big business value of automatic churn prediction and retention systems.

Figure 6 shows our experimental settings in a four month sliding window. First, we use Month N to label features in Month $N-1$, and use labeled features in Month $N-1$ to build a churn classifier. Second, we input the features with unknown labels in Month N into the classifier, and predict a list of potential churners (labels) in Month $N+1$ ranking by likelihood in descending order. Third, we will carry out retention campaigns on the list of potential churners in Month $N+1$ using A/B test, where the list is partitioned randomly into two parts, one for retention campaigns and the other for nothing. Fourth, we evaluate the churn prediction performance as well as retention performance in Month $N+2$, because we already know the labels in Month $N+1$ and retention campaign results in Month $N+2$. Finally, we use campaign results as labels to train the retention classifier to classify potential churners into multiple retentions strategies. This retention classifier will be used for matching churners with retention strategies in the next sliding window (not shown). The entire process repeats after the sliding window moves to the next month.

5.1 Performance Metrics

The churn prediction system will output a list of top U customers (non-churners in the current month) that have the higher likelihood to be churner in the next month. We use recall and precision metrics on top U users to evaluate the prediction results. Generally, increasing U will increase recall but decrease precision. Fixing a certain U, the higher recall and precision correspond to the better prediction performance. The definition of recall@U is

$$\text{R@U} = \frac{\text{The number of true churners in top } U}{\text{The total number of true churners}}. \quad (8)$$

Similarly, the definition of precision@U is

$$\text{P@U} = \frac{\text{The number of true churners in top } U}{U}. \quad (9)$$

We also use the area under the ROC curve (AUC) [13] evaluated on the test data, which is the standard scientific accuracy indicator,

$$\text{AUC} = \frac{\sum_{n \in \text{true churners}} Rank_n - \frac{P \times (P+1)}{2}}{P \times N}, \quad (10)$$

where P is the number true churners and N the number of true non-churners. Sorting the churner likelihood in descending order, we assign the highest likelihood customer with the rank n, the second highest likelihood customer with the rank $n-1$, and so on. The higher AUC indicates the better prediction performance. Because the number of positive (churner) and negative (non-churner) instances are quite imbalanced, the area under the precision-recall curve (PR-AUC) is a better metric than AUC for the overall predictive performance [10]. We use AUC, PR-AUC, R@U and P@U to evaluate the overall predictive performance in terms of a large volume of training data, a large variety of customer features, and a high velocity of processing new coming data.

Table 1: Statistics of Dataset (9 Months).

	Month 1	Month 2	Month 3	Month 4	Month 5	Month 6	Month 7	Month 8	Month 9
Churner	185779	173576	196984	184728	216010	201374	200492	199456	202873
No-Churner	1927748	1935496	1907548	1909698	1893469	1909472	1918349	1983917	1949832
Total	2113527	2109072	2104532	2094426	2109479	2110846	2118841	2183373	2152705

Table 2: Variety performance ($U = 2 \times 10^5$).

Features	AUC	PR-AUC	R@U	P@U	ΔPR-AUC
F1	0.87468	0.54123	0.41251	0.48133	0.000%
F2	0.91498	0.60879	0.50718	0.57511	12.483%
F3	0.91842	0.62172	0.51071	0.58718	14.871%
F4	0.89451	0.57691	0.46198	0.54991	6.592%
F5	0.87864	0.54681	0.42100	0.48543	1.031%
F6	0.90472	0.58874	0.48791	0.55782	8.778%
F7	0.88659	0.55181	0.42819	0.49765	1.955%
F8	0.89619	0.57093	0.46019	0.54177	5.488%
F9	0.89027	0.56799	0.45786	0.53181	4.944%

Table 3: The overall predictive performance.

Top U	Recall	Precision	AUC	PR-AUC
50000	0.22775	0.95928	0.93261	0.71553
100000	0.41198	0.86764		
150000	0.53930	0.75719		
200000	0.62884	0.66218		
250000	0.69362	0.58431		
300000	0.74550	0.52334		
350000	0.78498	0.47234		
400000	0.81621	0.42974		

Table 4: Importance ranking of some features.

Rank	Features	Category	Importance
1	balance	F1	0.163336
2	page_download_throughput	F3	0.159772
3	localbase_call_dur	F1	0.083948
6	page_response_delay	F3	0.051910
9	voice_quality	F2	0.036732
11	search_txt_topic	F8	0.015014
17	innet_dura× total_charge	F9	0.008867
41	labelpropaagtion_cooccurrence	F6	0.003507
54	pagerank_voice	F4	0.000703
68	pagerank_cooccurence	F5	0.000518

5.2 Volume

Figure 7 examines the question: when using baseline features in Section 4.1, do we indeed see increasing predictive performance as the training dataset continues to grow to massive size? This experiment aims to predict the potential churners in Month 7 in Table 1 by accumulating the labeled instances of feature vectors from Month 6 to Month 1. Such a experiment repeats 3 times to predict churners in Months 7, 8 and 9 by the sliding window in Figure 6, and the average results are reported (variance is too small to be shown). As Figure 7 shows (x-axis denotes volume of training data measured by the number of months and y is the predictive performance metrics AUC, PR-AUC, R@U and P@U), the performance keeps improving even when we add more than 4 times larger training data. We show results on $U = \{5 \times 10^4, 1 \times 10^5, 2 \times 10^5\}$, which denotes the number of predicted churners with top largest likelihoods. As far as R@5×10^4 is concerned, the larger training data achieves around 15% improvement, confirming that the Volume of big telco data indeed matters. This result suggests that telco operators need to store at least 4 months' data for a better churn prediction performance.

One should note, however, that the AUC, PR-AUC, R@U and P@U curves do seem to show some diminishing returns to scale. This is typical for the relationship between the amount of data and predictive performance, especially for linear models like logistic regression [36]. The marginal increase in generalization accuracy decreases with more training data for two reasons. First, the classifier may be too simple to capture the statistical variance of larger dataset. There is a maximum possible predictive performance due to the fact that accuracy can never be better than perfect. Second, more importantly, accumulating more earlier training data in past months may be not helpful in predicting potential churners in recent months. For example, the churner behaviors in Month 1 may be quite different from those in Month 7. Adding training data in Month 1 can provide little information to predict churners in Month 7. This follows the temporal first-order Markov property: the present state depends only on the most closest previous state.

5.3 Variety

The experiments reported above confirm that Volume (the number of training instances) can improve the predictive modeling accuracy. An alternative perspective on increasing data size is to consider a fixed number of training instances and increase the number of different possible features (Variety) that are collected, stored or utilized for each instance. This Variety perspective of big data motivates us to explore extensively OSS features plus BSS features. We believe that predictive modeling may benefit from a large variety of features if most of the features provide a small but non-zero amount of additional information about target prediction. To this end, we add separately the following categories of features to the baseline features (F1 for BSS features) one by one in Table 2: F2 for CS features, F3 for PS features, F4 for Call graph-based features, F5 for Message graph-based features, F6 for Co-occurence graph-based features, F7 for Topic features (complaints), F8 for Topic features (search queries), F9 for Second-order features. There are 150 features (F1: 70, F2: 9, F3: 25, F4: 2, F5: 2, F6: 2, F7: 10, F8: 10, F9: 20), and detailed descriptions on these features can be found in Section 4.1. On our dataset in Table 1, we repeat this experiment 7 times to predict churners in Months $3 \sim 9$, using one month labeled features for training and predict the potential churners in the next month as shown in Figure 6. The average results are reported (variance is too small to be shown).

Table 2 summarizes the contribution of each category of features to the baseline features. CS and PS features (F2 and F3) contribute significantly 12.483% and 14.871% improvements on PR-AUC, respectively. This result is promising because churners are quite related with KPI/KQI features from both voice and data services. We can use a customer-centric network optimization solution to improve KPI/KQI experiences of potential churners. PS features are more effective than CS features because customers use more and

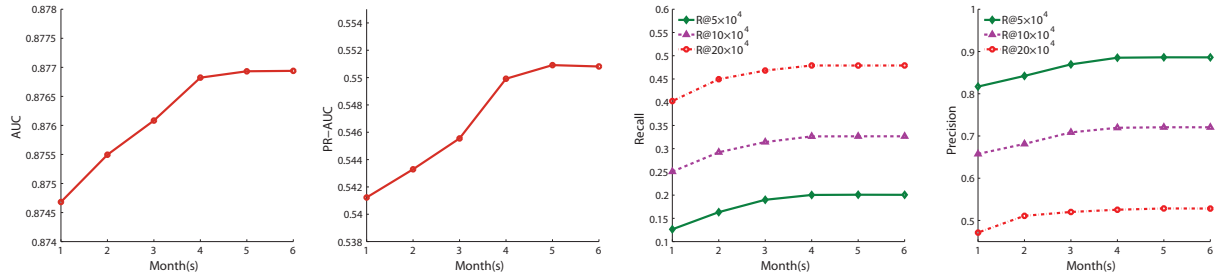

Figure 7: Adding more training data (x-axis) leads to continued improvement of predictive power (y-axis).

more data service in China, and the voice service becomes less important than before. Surprisingly, message graph-based features from complaint text (F4) has the least contribution 1.031% improvement in terms of PR-AUC, because almost all customers use the over-the-top (OTT) applications like Wechat, and seldom use the short message service (SMS). In contrast, call graph (F4) and co-occurrence graph-based (F6) features enhance the predictive performance more than message graph-based features. The importance of co-occurrence graph indicates that customers in the same spatiotemporal cube tend to churn with similar likelihoods. For example, university students have strong edge weights in co-occurrence graph, and often churn simultaneously. Topic features of complaint text (F7) make a small 1.955% improvement, implying that complaint is not a major early signal for churning. Although a majority of churners have bad experience, they still do not complain before churning. Similar results have also been observed in [27]. In contrast, topic features of search queries (F8) are more informative. Potential churners may access to other operators' portal, search other operators' hotline, search new handset, and so on. Finally, second-order features (F9) are beneficial in practice, which has been verified in many KDD Cup competitions [30]. These results support that features from OSS can improve the churn prediction accuracy. It is worthwhile integrating both BSS and OSS data for predictive modeling.

Table 4 shows importance ranking of some features among 150 features by RF classifier in each category. The most important feature is "balance", because most churners with different causes often show low balance than non-churners. The second important feature is "page_download_throughput", which becomes much smaller since churners often become inactive in data usage. Some KPI/KQI features are ranked at relatively higher places. Also, graph-based features play important roles in differentiating churners from non-churners. Indeed, churners influence non-churners through call, message and co-occurrence graphs, especially using label propagation on co-occurrence graphs. The second-order feature like "total_charge × innet_dura" means the product of the total charge and the in net duration of each customer, which is also ranked at a higher place. Topic features from search queries are informative with the rank 11.

Table 3 summarizes the final results using all 150 features and accumulating 4 month training data. The precision P@5 $\times 10^4$ is as high as 0.96, which means 96% of the top 5×10^4 predicted churners will really churn in the next month. Moreover, we gain 6.62% AUC improvement, 32.20% PR-AUC improvement, 52.44% R@2×10^5 improvement, and 37.57% P@2×10^5 improvement over the baseline in Table 2. The improvement of AUC is not as high as PR-AUC, R@U and P@U because AUC is not a good performance metric for the imbalance of positive and negative instances [10]. Although each category of features can improve PR-AUC, the integration of these features has the shared information so that the overall improvement is not a simple sum of their separate improvements in Table 5. One should note that the baseline is a widely-used solution for churn prediction in previous research work [16, 14, 13, 32, 1]. The only difference lies in that we use a larger volume of training instances and a higher variety of features from OSS data. In conclusion, Volume and Variety of telco big data indeed make churn prediction much easier than previous research work.

Table 5: Velocity performance ($U = 2 \times 10^5$).

Velocity	AUC	PR-AUC	R@U	P@U	ΔPR-AUC
30 days	0.87512	0.54081	0.40211	0.47093	0.000%
20 days	0.87770	0.54268	0.40994	0.47989	0.345%
10 days	0.87972	0.55116	0.41357	0.48201	0.576%
5 days	0.88073	0.55445	0.41392	0.48399	0.692%

5.4 Velocity

Table 5 addresses the velocity question: how fast do we need to update classifiers using new coming data (labeled instances of features) to capture the recent behaviors of prepaid customers? To validate the effectiveness of velocity, we move the sliding window every 30, 20, 10 and 5 days in Figure 6, and predict the churners in the next 30 days using baseline features. By moving the sliding window, we repeat experiments 5 times and report the average result (variance is too small to be shown). We see that feature engineering and classifier updating with every 5 days gives a slightly higher accuracy (less than 0.7% improvement with regard to PR-AUC) than that with every 30 days. The improvement steadily increases with the velocity increase of processing new coming data. These results confirm that a high velocity of processing new coming data is beneficial. However, we should note that the improvement is limited by consuming more computing resources in feature engineering and classifier updating. In practice, increasing velocity of updating features and classifiers by every 5 days will predict around 2500 more true churners in the top $U = 2 \times 10^5$ list. Considering only a subset of these churners will be retained, the reward is not that high when compared with computing costs. To make a good balance between computing efficiency and prediction accuracy, we still choose to update classifiers every month in practice. Another reason is that some big tables for feature engineering are summarized automatically by BSS monthly, which will save lots of feature engineering

Table 6: Business value of churn prediction.

Month	Group A					
	Top 5×10^4			Top $5 \times 10^4 \sim 1 \times 10^5$		
	Total	Recharge	Rate	Total	Recharge	Rate
8	7994	134	1.68%	8032	808	10.06%
9	8024	84	1.04%	8050	798	9.91%
Month	Group B					
	Top 5×10^4			Top $5 \times 10^4 \sim 1 \times 10^5$		
	Total	Recharge	Rate	Total	Recharge	Rate
8	7972	1474	18.49%	8026	2280	28.41%
9	7988	2458	30.77%	8042	3194	39.72%

costs since we do not need to re-generate these big tables manually by every 5 days. So, Table 3 summarizes the final predictive performance of our deployed churn prediction system (Volume = 4 months, Variety = 140 features, Velocity = 1 month).

5.5 Value

The business value is a major concern of the deployed churn prediction and retention system. Every month, the system provides a list of top $U = 1 \times 10^5$ potential churners ranking by classification likelihoods (4), which covers around 40% true churners as shown in Table 3. We select a subset of potential churners[4] (some from top 5×10^4 and others from top $5 \times 10^4 \sim 1 \times 10^5$) for the following prepaid mobile recharge offers: 1) Get 100 cashback on recharge of 100, 2) Get 50 cashback on recharge of 100, 3) Get 500MB flux on recharge of 50, and 4) Get 200-minute voice call on recharge of 50, when they enter the recharge period. In the A/B test, it is desirable to see a very low recharge rate of the predicted potential churners without recharge offers (Group A), and a relatively higher recharge rate of potential churners with recharge offers (Group B). We carry out retention campaigns in two consecutive months 8 and 9. In Month 8, we do not train retention classifiers in Figure 6 to match churners with recharge offers, but send offers by short message service to churners according to domain knowledge guided by operator experts. In Month 9, we use the campaign results as labels (customers who accept different offers are defined as different class labels in Subsection 4.3) to build a retention classifier, matching proper offers with churners automatically. In this way, we can gradually improve the retention efficiency to retain more potential churners.

Table 6 shows the recharge rates of A/B test in Months 8 and 9. In group A, the recharge rate is very low in both Months 8 and 9. In top 5×10^4 predicted churners, the recharge rate is less than 2%, which means 98% predicted churners in recharge period will not recharge within 15 days. In top $5 \times 10^4 \sim 1 \times 10^5$ predicted churners, the recharge rate is around 10%. All these results confirm that the high accuracy of the churn prediction system. In Month 8, randomly assigning recharge offers in Group B can significantly enhance the recharge rate, e.g., from 1.68% to 18.48% and from 10.06% to 28.41%, when compared with Group A. Furthermore, in Month 9, after matching recharge offers with churners in Group B, we see a further significant increase of recharge rate, e.g., from 1.04% to 30.77% and from 9.91% to 39.72%, when compared with Group A. Indeed, the strategy of matching offers with churners in Month 9 can retain more potential churners, which leads to around 50% more profit than Month 8. This provides one of the clearest illustration that the integration of churn prediction and retention system is indeed more valuable. From this perspective, our prediction target changes to those potential churners with a higher likelihood to be retained by a specific retention offer.

[4]This is a small-scale test due to the limitation of retention resources at that time.

Table 7: Comparison of methods for data imbalance.

Methods	AUC	PR-AUC	R@U	P@U
Not Balanced	0.83712	0.49092	0.38969	0.43239
Up Sampling	0.84140	0.51882	0.39180	0.44187
Down Sampling	0.84791	0.51967	0.39757	0.45101
Weighted Instance	0.87468	0.54123	0.41251	0.48133

5.6 Early Signals

Telco operators wish to predict churners as early as possible to provide proper retention strategies in a timely and precise manner. In this setting, we change the sliding window in Figure 6. We use labels in Month $N+2$ and features in Month $N-1$ to train the classifier, and use features in Month N and the classifier to predict the potential churners in Month $N+3$. This setting can predict churners 3 months earlier. Similarly, we can change the experimental setting to predict churners from $1 \sim 4$ months earlier. Using baseline features, we show the predictive performance of earlier features in Figure 8. Obviously, the earlier features provide the worse predictive performance. The accuracy significantly decreases with the increase of time interval between the observed features and the predicted labels in Figure 6. For example, the PR-AUC drops around 20% using features from 1 month to 2 month earlier for prediction. The results imply that the prepaid customers often churn abruptly without providing enough early signals. Most churners show their abnormal behaviors just before a month. These observations are consistent with Figure 7, where adding more earlier instances of features provides little additional information to improve predictive performance.

5.7 Data Imbalance

Data imbalance has long been discussed in building classifiers [6, 15]. There are four widely-used methods to handle data imbalance: 1) Not Balanced, 2) Up Sampling, 3) Down Sampling and 4) Weighted Instance. The first method directly train classifier using imbalanced churner and non-churner instances. The second method randomly copies the churner instances to the same number of non-churner instances. The third method randomly samples a subset of non-churner instances to the same number of churner instances. The fourth method assigns a proportion weight to each instance, where higher weights are assigned to churners and lower weights to non-churners. We use baseline features to evaluate different methods for data imbalance. Table 7 shows the average predictive performance using different methods (variance is too small to be shown). Clearly, the Weighted Instance method outperforms other methods, having around 10% improvement over the Not Balanced method in terms of PR-AUC. So, we advocate the Weighted Instance method to handle data imbalance in practice.

5.8 Classifiers

We compare some widely used classifiers in data mining: RF, LIBFM, LIBLINEAR (L2-regularized logistic regression) and GBDT. Some classifiers have won KDD CUP

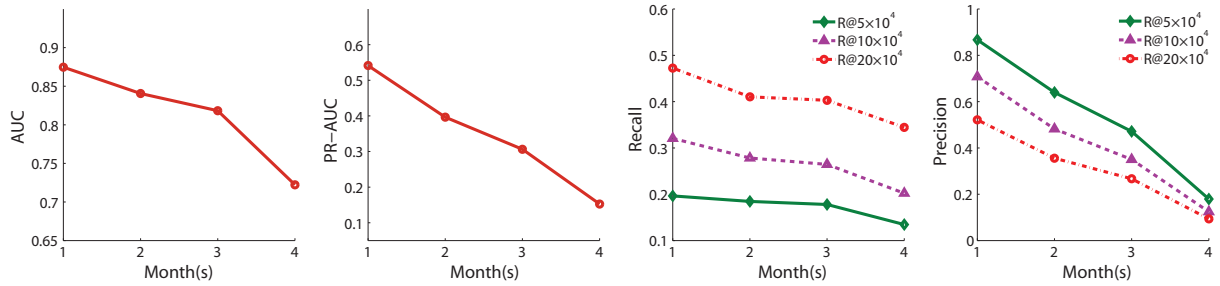

Figure 8: The earlier features produce worse predictive performance.

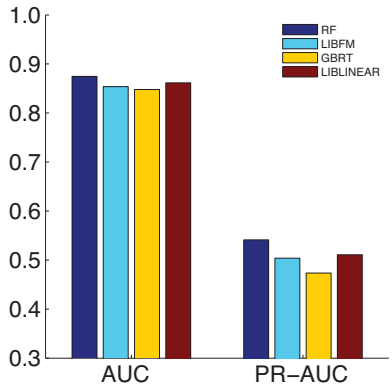

Figure 9: Comparison of different classifiers.

champions based on complicated feature engineering techniques [13, 36, 30]. RF and GBDT fix 500 decision trees with default parameter settings. For example, the learning rate of GBRT, LIBFM and LIBLINEAR is fixed as 0.1. LIBFM and LIBLINEAR use discrete binary features by preprocessing the original continuous feature values, because linear models are more suitable for sparse binary features. Using the same baseline features, we find that RF performs slightly better (less than 3%) than other classifiers in Figure 9 in terms of AUC and PR-AUC. The results are consistent with several previous research works advocating RF classifier for churn prediction [21, 7, 27]. Nevertheless, the classifiers are not as important as features. Given a variety of features, most scalable classifiers can achieve almost the same accuracy. The result indicates that adding a good feature may enhance the predictive modeling more significantly than changing to a better classifier, which have important implications for telco operators faced with competition. Telco operators dealing with larger data assets, more important than prediction techniques, potentially can obtain substantial competitive advantages over telco operators without access to so much data.

6. CONCLUSIONS

This paper re-explores an old but classic research topic in telco industry—churn prediction—from the 3V's perspectives of big data. Releasing the power of telco big data can indeed enhance the performance of predictive analytics. Extensive results demonstrate that Variety plays a more important role in telco churn prediction, when compared with Volume and Velocity. This suggests that telco operators should consider gathering, storing, and utilizing big OSS data to enrich Variety, which occupy more than 90% size of the entire telco data assets. Integration of BSS and OSS data leads to a better automatic churn prediction and retention system with a big business value. Extension work includes inferring root causes of churners for actionable and suitable retention strategies.

Previous churn prediction research is based only on BSS data such as call/billing records, purchase history, payment records, and CDRs, because it is much easier to acquire BSS data. In contrast, OSS data are much less consistent and much harder to acquire, because telco operators need domain knowledge from OSS/hardware vendors' systems they have. In OSS, the data quality and format will vary greatly among telco operators since each has their own software and hardware vendors. The cost of acquiring OSS data is often high, because of the large volume of data storage as well as license fees charged by OSS/hardware vendors. Therefore, the key challenge of the telco big data platform is the OSS data cleaning and integration layer as shown in Figure 2. However, OSS data do have big business value for accurate location information, mobile search queries and customer-centric network quality measurements. If OSS data analysis is done with appropriate technique, it can reward for the telco operators, some of which have claimed that they have been generating millions of revenue by selling OSS data analysis results. In this paper, our results on churn prediction also confirm that OSS data have cost-cutting and revenue-increasing benefits when combined with BSS data.

Acknowledgments

This work was supported by National Grant Fundamental Research (973 Program) of China under Grant 2014CB340304, NSFC (Grant No. 61373092 and 61033013), Hong Kong RGC project 620812, and Natural Science Foundation of the Jiangsu Higher Education Institutions of China (Grant No. 12KJA520004). This work was partially supported by Collaborative Innovation Center of Novel Software Technology and Industrialization.

7. REFERENCES

[1] A. M. Almana, M. S. Aksoy, and R. Alzahrani. A survey on data mining techniques in customer churn analysis for telecom industry. *Journal of Engineering Research and Applications*, 4(5):165–171, 2014.

[2] W. Au, K. Chan, and X. Yao. A novel evolutionary data mining algorithm with applications to churn prediction. *IEEE Trans. on Evolutionary Computation*, 7(6):532–545, 2003.

[3] K. binti Oseman, S. binti Mohd Shukor, N. A. Haris, and F. bin Abu Bakar. Data mining in churn analysis model for telecommunication industry. *Journal of Statistical Modeling and Analytics*, 1:19–27, 2010.

[4] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

[5] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[6] J. Burez and D. Van den Poel. Handling class imbalance in customer churn prediction. *Expert Systems with Applications*, 36(3):4626–4636, 2009.

[7] K. Coussement and D. Van den Poel. Churn prediction in subscription services: An application of support vector machines while comparing two parameter-selection techniques. *Expert systems with applications*, 34(1):313–327, 2008.

[8] K. Dasgupta, R. Singh, B. Viswanathan, D. Chakraborty, S. Mukherjea, A. A. Nanavati, and A. Joshi. Social ties and their relevance to churn in mobile telecom networks. In *EDBT*, pages 668–677, 2008.

[9] P. Datta, B. Masand, D. Mani, and B. Li. Automated cellular modeling and prediction on a large scale. *Artificial Intelligence Review*, 14(6):485–502, 2000.

[10] J. Davis and M. Goadrich. The relationship between Precision-Recall and ROC curves. In *ICML*, pages 233–240, 2006.

[11] E. J. de Fortuny, D. Martens, and F. Provost. Predictive modeling with big data: Is bigger really better? *Big Data*, 1(4):215–226, 2013.

[12] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[13] I. Guyon, V. Lemaire, M. Boullé, G. Dror, and D. Vogel. Analysis of the KDD Cup 2009: Fast scoring on a large orange customer database. 7:1–22, 2009.

[14] J. Hadden, A. Tiwari, R. Roy, and D. Ruta. Computer assisted customer churn management: State-of-the-art and future trends. *Computers & Operations Research*, 34(10):2902–2917, 2007.

[15] H. He and E. Garcia. Learning from imbalanced data. *IEEE Trans. on Knowledge and Data Engineering*, 21(9):1263–1284, 2009.

[16] S. Hung, D. Yen, and H. Wang. Applying data mining to telecom churn management. *Expert Systems with Applications*, 31(3):4515–524, 2006.

[17] S. Jiang, G. A. Fiore, Y. Yang, J. Ferreira Jr, E. Frazzoli, and M. C. González. A review of urban computing for mobile phone traces: current methods, challenges and opportunities. In *KDD Workshop on Urban Computing*, pages 2–9, 2013.

[18] S. Jinbo, L. Xiu, and L. Wenhuang. The application of AdaBoost in customer churn prediction. In *International Conference on Service Systems and Service Management*, pages 1–6, 2007.

[19] M. Karnstedt, M. Rowe, J. Chan, H. Alani, and C. Hayes. The effect of user features on churn in social networks. In *ACM Web Science Conference*, pages 14–17, 2011.

[20] N. Kim, K.-H. Jung, Y. S. Kim, and J. Lee. Uniformly subsampled ensemble (use) for churn management: Theory and implementation. *Expert Systems with Applications*, 39(15):11839–11845, 2012.

[21] A. Lemmens and C. Croux. Bagging and boosting classification trees to predict churn. *Journal of Marketing Research*, 43(2):276–286, 2006.

[22] E. Lima, C. Mues, and B. Baesens. Domain knowledge integration in data mining using decision tables: Case studies in churn prediction. *Journal of the Operational Research Society*, 60(8):1096–1106, 2009.

[23] N. Lu, H. Lin, J. Lu, and G. Zhang. A customer churn prediction model in telecom industry using boosting. *IEEE Trans. on Industrial Informatics*, 10(2):1659–1665, 2014.

[24] S. Neslin, S. Gupta, W. Kamakura, J. Lu, and C. Mason. Detection defection: Measuring and understanding the predictive accuracy of customer churn models. *Journal of Marketing Research*, 43(2):204–211, 2006.

[25] M. Owczarczuk. Churn models for prepaid customers in the cellular telecommunication industry using large data marts. *Expert Systems with Applications*, 37(6):4710–4712, 2010.

[26] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. 1999.

[27] C. Phua, H. Cao, J. B. Gomes, and M. N. Nguyen. Predicting near-future churners and win-backs in the telecommunications industry. *arXiv preprint arXiv:1210.6891*, 2012.

[28] Pushpa and S. Dr.G. An efficient method of building the telecom social network for churn prediction. *International Journal of Data Mining & Knowledge Management Process*, 2:31–39, 2012.

[29] D. Radosavljevik, P. van der Putten, and K. K. Larsen. The impact of experimental setup in prepaid churn prediction for mobile telecommunications: What to predict, for whom and does the customer experience matter? *Transactions on Machine Learning and Data Mining*, 3(2):80–99, 2010.

[30] S. Rendle. Scaling factorization machines to relational data. In *PVLDB*, volume 6, pages 337–348, 2013.

[31] K. Shvachko, H. Kuang, S. Radia, and R. Chansler. The hadoop distributed file system. In *IEEE Symposium on Mass Storage Systems and Technologies (MSST)*, pages 1–10, 2010.

[32] W. Verbeke, K. Dejaeger, D. Martens, J. Hur, and B. Baesens. New insights into churn prediction in the telecommunication sector: A profit driven data mining approach. *European Journal of Operational Research*, 218(1):211–229, 2012.

[33] W. Verbeke, D. Martens, C. Mues, and B. Baesens. Building comprehensible customer churn prediction models with advanced rule induction techniques. *Expert systems with applications*, 38:2354–2364, 2011.

[34] C.-P. Wei and I. Chiu. Turning telecommunications call details to churn prediction: a data mining approach. *Expert systems with applications*, 23(2):103–112, 2002.

[35] E. Xevelonakis. Developing retention strategies based on customer profitability in telecommunications: An empirical study. *Journal of Database Marketing & Customer Strategy Management*, 12:226–242, 2005.

[36] H.-F. Yu, H.-Y. Lo, H.-P. Hsieh, J.-K. Lou, T. G. McKenzie, J.-W. Chou, P.-H. Chung, C.-H. Ho, C.-F. Chang, Y.-H. Wei, et al. Feature engineering and classifier ensemble for kdd cup 2010. In *JMLR W & CP*, pages 1–16, 2010.

[37] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica. Spark: Cluster computing with working sets. In *HotCloud*, 2010.

[38] J. Zeng. A topic modeling toolbox using belief propagation. *J. Mach. Learn. Res.*, 13:2233–2236, 2012.

[39] J. Zeng, W. K. Cheung, and J. Liu. Learning topic models by belief propagation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(5):1121–1134, 2013.

[40] X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, Technical Report CMU-CALD-02-107, Carnegie Mellon University, 2002.