Report of Task 3: Your Phone Understands You*

Kaixiang Mo†, Ben Tan†, Erheng Zhong† and Qiang Yang†
†Hong Kong University of Science and Technology, Hong Kong
{kxmo, btan, ezhong, qyang}@cse.ust.hk

ABSTRACT

Demographic prediction is important for many applications, such as recommendation, personalization and behavior targeting. The task of the challenge [8] is to predict users' demographics based on the data collected by users' mobile phone. To accomplish the task, we propose a supervised learning framework to perform feature construction from the raw data, data clearing, feature extraction and selection, model building and ensemble, and prediction revision. The experimental results show that the framework can achieve prediction accuracies on "gender", "job" and "marital" as high as 96%, 83% and 86% respectively, and achieve RMSE on "age" and "number of people" as low as 0.69 and 0.66 respectively, under the leave-one-out evaluation.

1. INTRODUCTION

The task is to predict users' demographics, including "gender", "job type", "marital status", "age" and "number of family members", based on users' mobile data. We describe our solution, which is a supervised learning framework, in this paper. The proposed framework has five components: feature construction, which extracts features from the raw data, such as the number of calls in the morning, the number of applications used in the evening, etc. and converts the data of each user into a feature vector; data cleaning, which replaces missing data and normalizes features into a fixed scope; feature selection and extraction, which reduces the huge number of dimensions brought in the feature construction step; model building, which builds a classification or regression model based on the selected features; and prediction revision, which adjusts the prediction results according to the relationship between each demographic. We highlight the novelties of our solution as follows:

1. A unified feature construction framework. We define each feature is a conditional probability of an action by given users and conditions and then estimate this probability using maximum likelihood estimation techniques. In addition, to handle the huge data, we formulate the original data into a entity-relation model, where each kind of information is an entity (corresponding to one csv file) and the relation between each entity is built according to user ID. Then, to construct a feature, we can send a query to this generated relational database to obtain sufficient statistics. Our previous work [3] show that feature construction is an effective way to improve classification performance.
2. Cost-sensitive classification based regression models. As most feature selection and extraction methods are proposed for classification problems, to utilized these state-of-the-art approaches, we convent the regression tasks (number of family members and age) into cost-sensitive classification problems. The experimental results show that cost-sensitive SVM outperforms the support vector regression and linear regression models as much as 0.12 on RMSE. Most of our submissions for two regression tasks are based on such models.
3. Collaborative Prediction Adjustment. One interesting point of this challenge is that all subtasks are related. For example, age is related to job type. For a PhD student, she is unlikely younger than 16 or older than 35. Thus, besides building individual models for each task, we also utilize the task relevant to adjust predictions. In our experiment, this improves about 4% accuracy on classification tasks and reduces about 0.02 RMSE on regression tasks.

Considering the limited number of labeled data, we propose a simple model averaging framework to generate the final submission. Specifically, we first manually select models with best leave-one-out performance and then group them into several levels. Finally, we average the results respect to different levels to obtain different submissions. We analyze that the leave-one-out evaluation is not biased while the model averaging can reduce the variance, and hence the built models have good generalization ability.

2. THE PROPOSED FRAMEWORK

The proposed framework is composed of five components, including feature construction, data cleaning, feature selection and extraction, model building and prediction adjustment. One important point is that, we formulate the whole process into an iterative framework, and hence the evaluation results on built models in each iteration can help us construct more meaningful features in the next iteration. The whole process can be found in Figure 1.

*Kaixiang Mo, Ben Tan and Erheng Zhong contributed equally to this work.

This material was prepared for the Mobile Data Challenge 2012 (by Nokia) Workshop; June 18-19, 2012; Newcastle, UK.

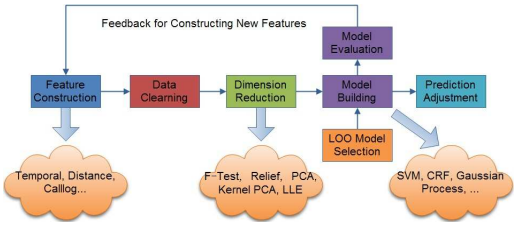


Figure 1: Main Flow

2.1 Feature Construction

As the given data is raw data, which cannot be taken as supervised models' inputs, the first and the most important process is to construct features with highly discriminability and represent each user as a feature vector, i.e., $\mathbf{x} = \{x_i\}_{i=1}^m$, where x_i represents the i-th value of the instance $\mathbf{x}$ and m is the number of features. Without explicit prior knowledge on inferring user profiles, we aim to construct features as many as possible and then perform feature selection or extraction. Since there are infinite ways to perform feature construction, to make the feature construction process tractable, we formulate it into a unified paradigm. We propose to extract two kinds of features: global and local. A global feature is defined as the probability that one given user will do one kind of actions, where each kind is related to a csv file F, and formally, we define it as $Pr(F|u, \Theta)$. This kind of features captures users' global active status. In addition, we define a local feature in each csv file as the conditional probability of an action given users and conditions. Formally, for a given user u and conditions Θ, the conditional probability of an action is defined as $Pr(a|u, \Theta)$. Then, the task of feature construction can be divided into three subtasks:

- Enumerating possible actions as many as possible
- Defining appropriate conditions
- Computing the conditional probabilities

Firstly, we categorize the actions in the raw data into three cases: continue variables, such as the time length of calling (in calllog.csv) and the avdelt value (in accel.csv); independent discrete variables which are different for different users, such as the number of places visited by users (in visit_sequence_10min.csv); and common discrete variables which are shared by every users, such as the call type, e.g, short message and phone call (in calllog.csv). For the continue variables, we extract their mean and variance. For example, we extract the mean and variance of the call length for each user. For the common discrete variables, we record the frequency of each discrete value for each user. For example, we extract the number of calls, number of messages, etc. For the independent variables, we retain ratio of the top-5 values as features. For example, we extract the running time ratio of top-5 frequent applications for each individual user. The third level is specific features, which are designed for specific task, such as the average number of accelerometer records for gender and the number of in-calls for job type. We summarize the extracted actions or statistics in each csv file as follows:

- accel: the average acceleration and the Std of average acceleration
- application: the general operation to a phone, the three most frequently used applications
- bluetooth: the three most frequently used bluetooth prefix
- calendar: private appointment
- calllog: the most frequent call regions, the out-going calls and in-coming calls, the message and voice calls, the duration of calls, and the missing calls
- contacts: the phone prefix
- gsm: the time staying at home
- media: the average size of the media and the std of the size of the media
- mediaplay: listening to songs from a particular artist and listening to a particular song
- process: using "phonebook.exe", using "accontrol.exe", using "sniffer.exe" and "WELCOM.exe"
- distance: moving length and maximum moving speed
- sys: space left in the drive D and time being in silent mode
- wlan: most frequently used brand
- visit: most frequently visited places

Secondly, we introduce two conditions based on the temporal information. The first one is based on whether the happening time is weekend or not and the second one is based on four user-defined timeslots: morning, afternoon, evening and midnight. Then, for a given user, his/her features can be constructed based on the above actions and conditions. For example, $Pr(a = calling|u = 001, morning)$ represents the probability that the user 001 makes a call in the morning. Motivated by "bag of words" [9], we estimate this probability based on counting. In other words, we compute this probability by calculating the number of given actions for each user under some conditions. For example, we aim to calculate the number of calls, the number of applications, the number of processes, etc. In practice, since the number of records in different csv files can be very different, e.g, tens in application.csv and thousands in accel.csv, in order to avoid the imbalance problem, we estimate $Pr(a|u, \Theta)$ according to the file F that the action a belongs to. Hence, we denote it as $Pr(a|u, \Theta, F)$. Using the Bayesian rule, we obtain the following computing equations:

$$Pr(F|u, \Theta) = \frac{Pr(F, \Theta, u)}{Pr(\Theta, u)} \propto \frac{n_{F,u,\Theta}}{n_{u,\Theta}} \quad (1)$$

$$Pr(a|u, \Theta, F) = \frac{Pr(a, \Theta, u, F)}{Pr(\Theta, u, F)} \propto \frac{n_{a,u,\Theta,F}}{n_{F,u,\Theta}} \quad (2)$$

where $n_{F,u,\Theta}$ is the number of records under conditions Θ in the file F, $n_{a,u,\Theta,F}$ denotes the number of action a that the user u takes under conditions Θ in the file F, $n_{u,\Theta}$ denotes the number of all actions that the user u takes under Θ, and $n_{F,u,\Theta}$ denotes the number of all actions that the user u takes under Θ in the file F. These statistics can be calculated directly from the raw data in csv files directly. Based on the conclusion in [9], this estimation is also the result of maximum likelihood estimation. In addition, to make the computation process efficient, we formulate the original data into a entity-relation model, where each raw information is an entity (corresponding to one csv file) and the relation between each entity is built according to user ID. Then, each statistic is related to one query. For example, $n_{a=calling,u=001,morning}$ is considered as a query, "select count(*) from calllog where user=001, action=calling and time=morning". Let n denote the number of users, m denote the number of features and c denote the number of tasks. The generated dataset can be defined as $T = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^n$, where $\mathbf{y}_i = \{y_j^i\}_{j=1}^c$ and $y_j^i \in Y_j$. Y_j is the label space of the j-th subtask. For example, for the gender prediction, $Y = \{female, male\}$ or $Y = \{1, 2\}$. In this challenge, $n = 80$ and $c = 5$.

2.2 Data Cleaning

Each feature may have different value scopes and some features may contain missing values. In addition, some of the constructed features may have non-zero values in only one or two users, which has no discriminability. Thus, in this step, we perform feature filtering and normalization. Firstly, we set a threshold for each feature, if there are only two non-zero values of this feature, it is removed. Then, for the rest features, we normalize them into $[-1, 1]$ using

$$x_i^t = 2\frac{x_i - x_{min}}{x_{max} - x_{min}} - 1 \tag{3}$$

where x_{max} and x_{min} represent the maximal and minimal values of this feature.

2.3 Dimension Reduction

The number of initial features constructed from raw data is huge, which is more than tens of thousands. Compared to the number of instances, which is less than 100, the huge number of features will cause models overfitting and misguide the prediction. Thus, we perform dimension reduction before model building. We adopt two kinds of methods: one is supervised feature selection, including F-test [13] and Relief [5] and the other is unsupervised feature extraction, including PCA [7], kernel PCA [14] and LLE [12], which stand for three kinds of feature extraction methods, i.e., linear, kernel-base, and manifold-base respectively. We discuss these adopted methods briefly as follows.

F-Test F-Test is any statistical test in which the test statistic has an F-distribution under the null hypothesis. It is widely used when comparing statistical models in order to select the model that best fit your data distribution.

Relief Relief feature selection evaluates each attribute by comparing an sample of instances with a set of nearest neighbor instances, the attribute which makes instances with the same label close to each other while keeping instances with different labels away from each other has high relevancy.

PCA Principal component analysis generates a set of linearly uncorrelated variables in all the given possibly correlated variables.

kernel PCA Kernel principal component analysis is an extension of principal component analysis using techniques of kernel methods. Using a kernel, the originally linear operations of PCA are done in a high dimension space with non-linear mapping.

LLE Locally Linear Embedding attempts to discover nonlinear structure in high dimensional data by exploiting the local symmetries of linear reconstructions.

2.4 Model Building

After obtaining features with high discriminability, we perform model building following two ways. One is to build models for each task individually, where we build a serial of models based on state-of-the-art algorithms, including C4.5 [10], gradient boosted tree [4], random forest [16], SVM [2] and logistic regression for classification tasks, and RepTree [1], support vector regression [15], Gaussian process [11], linear regression and lasso [17] for regression tasks. These kinds of algorithms can be divided into three categories: tree-based algorithms, linear algorithms, and kernel-based algorithms. For both linear and kernel-based algorithms, each of them aims to build a model f with the following objective:

$$\min_f \ell(f, T) + \lambda R(f) \tag{4}$$

where $\ell(f, T)$ is the loss function, λ is the trade-off parameter, and $R(f)$ is the regularization term to control the model complexity. For example, SVM uses the hinge loss and linear regression uses square loss. In addition, considering the huge number of features and small number of instances, beside the traditional L2-norm for R, we also build models with L1-norm, i.e., $R(f) = |f|_1$. For the regression tasks, we also convert them into cost-sensitive classification tasks. We implement it based on hinge loss as follows. Let $s(\mathbf{x}_i, y)$ be a discriminative function for both the label and the instance, e.g, $s(\mathbf{x}_i, y) = \mathbf{w}_y \mathbf{x}_i^T$ for linear SVM, where $\mathbf{w}_y$ is the coefficients. The hinge loss for classification based on s can be represented as

$$\sum_{i=1}^{n} \sum_{y} \max\Big(0, 1 - \big(s(\mathbf{x}_i, y_i) - s(\mathbf{x}_i, y)\big)\Big) \tag{5}$$

By incorporating the cost information, the objective function for regression tasks is defined as

$$\min_s \sum_{i}^{n} \sum_{y} M_{y_i,y} \max\Big(0, 1 - \big(s(\mathbf{x}_i, y_i) - s(\mathbf{x}_i, y)\big)\Big) \tag{6}$$

where $M_{i,j}$ is the cost for misclassifying the instance into j-th class, of which true class is i. Take the "age" task for example, there are 6 classes, $[0, 1, 2, 3, 4, 5]$. Then, the M is a 6×6 matrix, where $M_{i,j} = (i - j)^2$. The experimental results show that this convention can improve the prediction RMSE by 0.12 on the "number of families" task.

2.5 Prediction Adjustment

For the first kind of model building, each subtask is independent. Based on several common senses: 1. users in different groups may have different probabilities to take different jobs, e.g, users have few opportunity to get married and become a PhD student if they are younger than 16; 2. users in different marital status may have different numbers of family members, e.g, married people unlikely have only one family member; 3. job type is also biased toward to users with different genders, e.g, there are more male PhD students than female students, we found that there are dependency between each subtask. To utilize their correlations, we revise the independent predictions using their concurrence from two aspects. The first one is to produce the final prediction for each task as the combination of the original prediction plus the weight summation of the predictions from other tasks. Formally, we define $c \times c$ matrices to measure the label relation between two tasks, i.e, $W_{i,j}$ for the i-th and j-th tasks. Take the "gender" and "marital" tasks as an example, where "gender" has two classes and "marital" has three classes, and then their relation matrix is 2×3. Each entities $W_{i,j}(m, n)$ in each relation matrix represent the normalized label concurrences between the label m in the i-th and the label n in the j-th tasks. For example, there are 20 users who are male and married, and then the corespondent entry $W_{gender,marital}(male, married) = 20/80 = 0.25$. We define the output of $f(\mathbf{x}_i)$ as a vector of scores for each candidate label. For example, for the "gender" task, the output of $f(\mathbf{x}_i)$ is $[s(\mathbf{x}_i, y = 0), s(\mathbf{x}_i, y = 1)]$. Let g_j denote the final

Table 1: Gender (Accuracy)

Feature Set	J48	Random Forest	GBDT	SVM	Logistic Regression	Linear SVM
General (50)	0.7948	0.7820	0.7948	0.8589	0.8333	0.8333
Temporal General (50)	0.8589	0.8846	0.8846	0.8974	0.8974	0.8846
Call Log Related (50)	0.7692	0.7179	0.7435	0.7564	0.6923	0.6923
Cross User Features (50)	0.8205	0.9102	0.8589	0.9102	0.9230	0.9230
Speed/Distance Related (20)	0.8333	0.7564	0.7948	0.7820	0.7820	0.7820
Count by 10min (20)	0.8717	0.8717	0.8333	0.8717	0.8589	0.8589
Best Combination (20)	0.8589	0.8846	0.8846	0.8974	0.8205	0.7820

Table 2: Job (Accuracy)

Feature Set	J48	Random Forest	GBDT	SVM	Logistic Regression	Linear SVM
General (20)	0.5616	0.5342	0.6164	0.5753	0.5753	0.5753
Temporal General (50)	0.6027	0.7260	0.6849	0.7808	0.7123	0.7123
Call Log Related (20)	0.4931	0.3972	0.4520	0.4931	0.4246	0.4657
Cross User Features (50)	0.5479	0.6301	0.5616	0.6164	0.5890	0.5753
Speed/Distance Related (20)	0.5616	0.5068	0.5205	0.5068	0.4794	0.4794
Count by 10min (50)	0.6027	0.6575	0.6712	0.6575	0.6712	0.6712
Best Combination (50)	0.5890	0.6575	0.6712	0.7808	0.6986	0.6986

prediction for the j-th task and we have

$$g_j(\mathbf{x}_i) = \beta f_j(\mathbf{x}_i) + (1-\beta)\sum_{k\neq j} f_k(\mathbf{x}_i)W^T_{j,k} \tag{7}$$

In practice, β can be determined by cross-validation techniques. We found that the optimal value is 0.9 under the Leave-One-Out evaluation in the experiment. Another strategy is considering the initial predictions from other tasks as features for the current task and then build a new model,

$$g_j(\mathbf{x}_i) = f_j(\{\mathbf{x}_i, [f_k(\mathbf{x}_i)]_{j\neq k}\}) \tag{8}$$

where $\{\mathbf{x}_i, [f_k(\mathbf{x}_i)]_{j\neq k}\}$ indicates the new feature vector is combined by the original features and predicted labels in other tasks. We used these two techniques in all of our prediction results. Due to the limited number of labeled data, we perform model averaging to avoid overfitting. Formally, the final model $h(\mathbf{x})$ is defined as

$$h(\mathbf{x}) = \frac{1}{M}\sum_f f(\mathbf{x}) \tag{9}$$

where M is the number of models for averaging. We analyze that the averaged model can reduce the prediction variance as follows. Let f^* denote the ideal model and $d(f, f^*)$ denote the difference between f and f^*. We obtain

$$\begin{aligned} d(f,f^*) &= \mathbf{E}_f\big(f(\mathbf{x}) - f^*(\mathbf{x})\big)^2 \\ &= \mathbf{E}_f\big(f(\mathbf{x})^2 - 2f(\mathbf{x})f^*(\mathbf{x}) + f^*(\mathbf{x})^2\big) \end{aligned}$$

On the other hand, the difference $d(h, f^*)$ is

$$\begin{aligned} & d(h,f^*) = \mathbf{E}\big(\mathbf{E}_f(f(\mathbf{x}) - f^*(\mathbf{x}))\big)^2 \\ = \;& \mathbf{E}\Big(\big(\mathbf{E}_f f(\mathbf{x})\big)^2 - 2\big(\mathbf{E}_f f(\mathbf{x})\big)f^*(\mathbf{x}) + f^*(\mathbf{x})^2\Big) \\ \leq \;& d(f,f^*) \qquad as \qquad \mathbf{E}[f]^2 \leq \mathbf{E}[f^2] \end{aligned}$$

In addition, according to the conclusion in [6], leave-one-out evaluation is not biased for prediction and hence the strategy we adopted, model averaging + leave-one-out, makes the built models have good generalization abilities.

3. EXPERIMENTAL RESULTS

The main task in the experiment is to perform model selection, which aims to detect the best combination of feature sets, learning models and model parameters. Most model implementations are based on Weka [1] and libsvm [2]. We evaluate the effectiveness of each combination using Leave-One-Out (LOO) method. For classification tasks (i.e. gender, marital, job), the performance is measured by classification accuracy. Performance in regression tasks (i.e. number of people, age) is tested by Root-Mean-Square Error(RMSE). The best performance of each model using different feature sets are shown in Table 1∼4 corresponding to each subtask respectively. We observe that, in most cases, SVM and Random Forest achieve the best performance in three classification tasks. Thus, in our final submissions, which are based on model averaging, most base results are contributed by these two models. On the other hand, cost-sensitive SVM outperforms other approaches. This, from the empirical aspect, provides evidences that the transformation from regression to classification is effective, as shown in Section 2.4. In addition, we report some results in feature construction so as to indicate the improvement progress. For each subtask, we list 3 valuable features according to the relief feature selection algorithm in Table 5.

As described in subsection 2.1, a large number of features are extracted from the raw data. These features are constructed step by step in our experiment, followed by strict selection process in order to provide solid foundation for subsequent model building. Firstly, probability of each record is computed based on temporal conditions. Secondly, we calculate general statistical features, namely, mean and covariance for continues records, number of distinct values and top-5 ratios for discrete ones. Thirdly, general statistical and temporal conditions are considered together to generate plenty of sophisticated feature sets, which potentially have power of exploring more detailed information. Finally, specific features are extracted for each task. To test the discrimination ability of constructed feature sets in each step, first of all, we build a base line method as follow 1) the majority class in training data is employed to make the prediction for classification tasks; 2) prediction value of regression tasks is the average output value of training data. Secondly, we train different models on each feature set and compare the results against baseline method. The feature set can be retained for future use if its best result among models outperforms the baseline, otherwise, discarded.

[1] `http://www.cs.waikato.ac.nz/~ml/weka/`
[2] `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

Table 3: Marital (Accuracy)

Feature Set	J48	Random Forest	GBDT	SVM	Logistic Regression	Linear SVM
General (20)	0.6455	0.6455	0.6329	0.6835	0.6582	0.7341
Temporal General (50)	0.7341	0.6455	0.7088	0.7594	0.7341	0.7215
Call Log Related (50)	0.5822	0.5569	0.6075	0.5696	0.5822	0.5696
Cross User Features (50)	0.6455	0.6962	0.6075	0.7721	0.7468	0.7468
Speed/Distance Related (20)	0.6582	0.5316	0.5569	0.5822	0.5316	0.5316
Count by 10min (50)	0.6202	0.7341	0.7341	0.7594	0.7594	0.7594
Best Combination (50)	0.6329	0.7088	0.7215	0.7848	0.7721	0.7721

Table 4: Regression Tasks (RMSE)

Feature	Gaussian Process	SVR	SVM	LinearSVM	Gaussian Process	SVR	SVM	LinearSVM
	Age				Num_People			
General (50)	1.052	1.007	-	-	1.098	1.001	-	-
Temporal General (50)	0.932	0.911	0.880	0.887	0.964	0.885	0.765	0.824
Call Log Related (50)	0.969	0.975	1.024	1.066	1.038	1.024	1.083	1.089
Cross User Features (50)	1.010	0.914	0.851	0.887	0.865	0.845	0.765	0.800
Speed/Distance Related (20)	1.004	1.000	0.987	0.993	1.065	1.033	1.089	1.058
Count by 10min (50)	1.081	0.957	0.880	-	0.981	0.874	0.721	-
Best Combination (50)	1.018	0.987	0.864	0.948	0.819	0.760	0.721	1.013

The best performance in each round is shown in Fig.(2), where "Round 0" means base line method, "Round 1∼4" indicate the four feature construction steps. From the result, we can see that feature set with good discrimination ability is well constructed in each step. Specially, feature sets in round 3 always outperform that in the previous two rounds. Later on, we strengthen the feature sets with several strategies. For instances, in round 5, a new feature set is raised by selecting a group of subsets with high discrimination. From the results, we can see that improvement always happens. Furthermore, in round 6, feature selection and extraction are employed, by which we reduce each feature set into lower dimension (i.e. dimension 10, 20,50,100). It brings in attractive performance improvement. Finally, we carry out ensemble method on feature sets and models with top performance, which makes the prediction results even better. Hence, we select it as the final submitted method, the details of which is described at the end of this section.

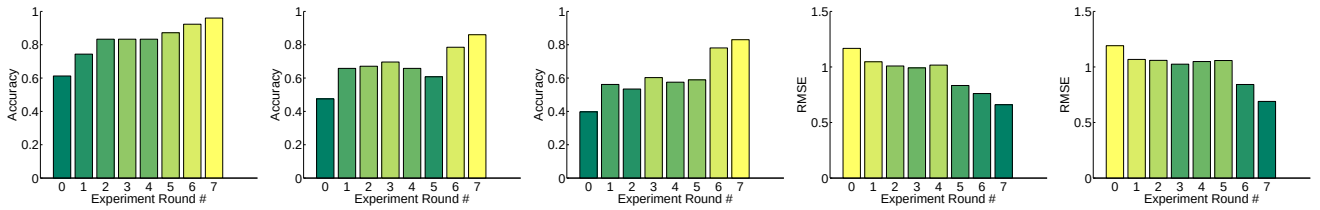
A serial of state-of-the-art algorithms are selected to perform model building. We show their results in Table 1∼4, where the marks in the table (#) indicates the number of features in the set. From the results, we can find that some features achieve equally high performance with all models, such as "Temporal General" in Table 1, "Cross User Features" in Table 4. Nonetheless, most feature sets gain high prediction results with some models but low values with the rest ones, since the models capture data information from different aspects. Consequently, feature sets with discrimination information will never be missed due to our carefully selected models. In addition, feature selection and extraction also play an important role in the experiment. It not only yields performance improvement, but also picks out some explainable features. From Table 5, it is not difficult to come up with some connections between features and task classes. For example, feature-1 in gender is a informative attribute to distinguish female from male. As we known, cell phone accelerometer could measure the speed of movement of the object it is attached to. Therefore, one possible reason is that female is quieter than male, this factor is reflected by the variance of acceleration on cell phone.

Finally, we describe the submitted results as follows. We first adjust all predictions using the strategies presented in Section 2.5. Consequently, the adjusted and unadjusted predictions are pooled together to perform model averaging to generate the final submissions. For classification tasks, the final submission is the model averaging of single results. Specifically, each single model will produce a prediction of class probabilities, or confidences for each class. Then, we aggregate all probabilities together and select the class with highest probabilities as the final prediction. For regression tasks, we average the predictions from each single model directly. Our consideration is that, combinations of different models and feature spaces can describe the final predictions from different perspectives, and feature spaces, and hence achieve more robust results.

- Top-3 predictions with lowest leave-one-out errors, all of which are generated by SVM (Cost-sensitive SVM for regression tasks).
- Predictions which have lower leave-one-out errors than a threshold, 0.9 in accuracy for "gender", 0.8 in accuracy for "job" and "marital", and 0.75 in RMSE for "age" and "number of people". Each task contains about 15 base predictions. Among them, predictions of three classification tasks are based on SVM and Random Forest while predictions of two regression tasks are based on Gaussian Process and Cost-sensitive SVM.
- Predictions which have lower leave-one-out errors than another threshold, 0.85 in accuracy for "gender", 0.75 in accuracy for "job" and "marital", and 0.8 in RMSE for "age" and "number of people". Each task contains about 30 base predictions.
- Predictions in 4 specific feature sets filtered by Reflief selector: accel, wlan, calllog and visit under temporal conditions with all models.
- Predictions of SVM (Cost-sensitive SVM for regression tasks) under the feature space constructed in round 6.

4. CONCLUSION

We presented a supervised learning framework to predict users' demographics based on their log data recorded by their mobile phones. Importantly, we proposed a unified feature construction process to build features from provided raw data for each user. We formulate each feature as a conditional probability of an action given users and conditions. Consequently, these conditional probabilities can be computed through action counts in each csv file. We processed data to convert each user as a feature vector and clear noises. The products of the previous steps were taken as the input for dimension reduction, in order to extract dis-



(a) Gender (b) Marital (c) Job (d) Number of People (e) Age

Figure 2: Best Performance using Different Features (Round 0: baseline method; Round 1: global features with temporal conditions; Round 2: local features without conditions; Round 3: local features with temporal conditions; Round 4: task-specific local features with temporal conditions; Round 5: combination of high discrimination features from round 3 and 4; Round 6: features generated by dimension reduction and feature selection; Round 7: final ensemble results.)

Table 5: Several Useful Features

Task	Feature-1	Feature-2	Feature-3
Gender	variance of acceleration	time spent at working place	fraction of non-local calls at work days
Job	chance of being at home at night	time spent at work place at weekend	chance of calling/answering at night
Marital	number of places visited in the evening	frequently used applications at evening	ratio of calls and messages
Age	The brands of bluetooth devices used	The brands of wlan devices used	The brands of devices used at work
Num_People	time talking to closest mobile-number	time talking to closest telephone	The brands of devices connected to

criminative attributes. Multiple state-of-the-art algorithms were exploited to construct models for prediction, such as SVM for classification and Gaussian Process for regression. Specifically, we proposed a cost-sensitive classification based framework for regression tasks and achieve respectable improvements on RMSE. We also adjust our predictions collaboratively to incorporate correlation knowledge among subtasks. The final submissions are all based on model averaging, as analyzed; this strategy can reduce the prediction variance and has good generalizability.

Acknowledgement

We thank the support of Hong Kong RGC GRF projects 621010 and 621211. We thank Nathan N. Liu and Yin Zhu for discussions.

References

[1] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and Regression Trees*. Chapman and Hall/CRC, 1 edition, Jan. 1984.

[2] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Min. Knowl. Discov.*, 2(2):121–167, June 1998.

[3] W. Fan, E. Zhong, J. Peng, O. Verscheure, K. Zhang, J. Ren, R. Yan, and Q. Yang. Generalized and heuristic-free feature construction for improved accuracy. In *SDM*, pages 629–640. SIAM, 2010.

[4] J. H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2000.

[5] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, Mar. 2003.

[6] T. Hastie, R. Tibshirani, and J. H. Friedman. *The elements of statistical learning: data mining, inference, and prediction: with 200 full-color illustrations*. New York: Springer-Verlag, 2001.

[7] I. T. Jolliffe. *Principal Component Analysis*. Springer, second edition, Oct. 2002.

[8] J. K. Laurila, D. Gatica-Perez, I. Aad, J. Blom, O. Bornet, T.-M.-T. Do, O. Dousse, J. Eberle, and M. Miettinen. The mobile data challenge: Big data for mobile computing research. In *Mobile Data Challenge by Nokia Workshop, in conjunction with International Conference on Pervasive Computing*. Newcastle, 2012.

[9] W. Pan, E. Zhong, and Q. Yang. Transfer learning for text mining. In C. C. Aggarwal and C. Zhai, editors, *Mining Text Data*, pages 223–257. Springer, 2012.

[10] J. R. Quinlan. Bagging, boosting, and c4.5. In *In Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 725–730. AAAI Press, 1996.

[11] C. E. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[12] Roweis, T. Martinetz, K. Schulten, N. Netw, V. Kumar, A. Grama, A. Gupta, and G. Karypis. Nonlinear dimensionality reduction by locally linear embedding, 2000.

[13] S. S. Sawilowsky. Nonparametric Tests of Interaction in Experimental Design. *REVIEW OF EDUCATIONAL RESEARCH*, 60(1):91–126, Jan. 1990.

[14] B. Schlkopf, A. J. Smola, and K. R. Müller. Kernel principal component analysis. *Advances in kernel methods: support vector learning*, pages 327–352, 1999.

[15] A. J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222, Aug. 2004.

[16] L. B. Statistics and L. Breiman. Random forests. In *Machine Learning*, pages 5–32, 2001.

[17] R. Tibshirani. Regression shrinkage and selection via the lasso. *J. Roy. Statist. Soc. Ser. B*, 58(1):267–288, 1996.