Alternate *ACM* SIG Proceedings Paper in LaTeX

TestName A

ABSTRACT

TO BE DONE

1. INTRODUCTION

TO BE DONE

2. DATASET ANALYSIS

2.1 Data Collection

Zhubajie [1](ZBJ) is a famous online freelancing marketplace platform in China. Employers publish different kinds of human intelligent tasks(HITs) on ZBJ for freelancers to work on. There are three kinds of HITs on this platform: tender, pitch and piece-work. Tender is a kind of task that after a employer publish a task with content and salary, freelancers submit description of themselves and their ability to accomplish the task, then the employer make a choice to cooperate with one of the candidates. While a pitch task is a task that freelancers directly submit their fulfilled works for employer to choose, and one candidate or several of the candidates are rewarded. In a piece-work task, employer need different freelancers to fulfill a lot of micro-tasks. We crawled a dataset of the fulfilled tasks and their corresponding freelancer submissions from ZBJ platform. The dataset spans a time period of 13 months between January 2015 through January 2016 and it contains 121 thousand HITs posted by 93 thousand employers with 2.7M works submitted by 54 thousand freelancers. We select eight categories of the tender and pitch HITs, then tag them with five(need to be alternated) price bins according to the salary of the task.

2.2 Data Features

TO DO

3. CHOSEN WORK PREDICTION

[1]www.zbj.com

DOI: notknown

In this section, we will formulate the chosen work prediction problem in a tender or pitch HIT. We will then describe features of the freelancers, employers and submitted works.

3.1 Problem Formulation

Considering employer $e \in E$ (Employer set) publishes a HIT, $task_e^m$, which is the m-th HIT from the whole HITs set T. And a freelancer $u \in U$ (Freelancer set) submit the n-th work w_{ue}^{mn} to $task_e^m$. Our work is to predict the $top-k$ works which are going to be rewarded by e in $task_e^m$ and belongs to category $cate_i$ and price bin $price_j$. If a work w_{ue}^{mn} is chosen by e , then the result $r(w_{ue}^{mn}) = 1$, otherwise $r(w_{ue}^{mn}) = 0$.

3.2 Data features for prediction

Freelancer features.

Freelancer features describe the characters of the freelancer u to enable us to acquire, in order to judge the ability of u to fulfill a HIT.

Number of categories: We use $cateNum(u)$ to represent the number of categories that a freelancer u has worked on.

Number of works in related category: We use function $workNum(u, cate_i)$ to describe the category-related interest of freelancer u. $workNum(u, cate_i)$ is the number of works submitted by freelancer u to HITs in $cate_i$.

Fraction of works in related category: Fraction of works in related category is also used to describe the category-related interest of the freelancer u. Fraction of work in category $cate_i$ is
$workNumRate(u, cate_i) = \frac{workNum(u,cate_i)}{\sum_i workNum(u,cate_i)}$.

Number of works in related price bin: We use function $priceNum(u, price_j)$ to represent the price-related interest of the freelancer u. Similarly to the feature $workNum(u, cate_i)$, $priceNum(u, price_j)$ represents the number of works that a freelancer u has submitted to HITs belongs to price bin $price_j$.

Fraction of work sin related price bin: We use function $priceRate(u, price_j)$ to describe the price-related interest of the freelancer u. Fraction of work in price bin $price_j$ is
$priceRate(u, price_j) = \frac{priceNum(u,price_j)}{\sum_i priceNum(u,price_j)}$.

System reputation level: ZBJ platform gives a freelancer u a system reputation level $sysLevel(u)$ when u has fulfilled certain amount of HITs. The more works u submits and gets reward from employers, the higher level $sysLevel(u)$ u will get from the platform.

Efficiency: Efficiency is the number of works that u submits in a limited time period. We use three functions $effi7d(u)$, $effi14d(u)$, $effi30d(u)$ to represent the efficiency of u corresponding to three different time periods which are 7 days, 14 days and 30 days. Efficiency is a character that show us the work ability and passion of a freelancer.
Category-related efficiency and Price-related efficiency: Similarly to efficiency, we calculate category-related efficiency and price-related efficiency in different categories or different price bins for a freelancer u as $cateEffi7d(u, cate_i)$, $cateEffi14d(u, cate_i)$, $cateEffi30d(u, cate_i)$, $priceEffi7d(u, price_j)$, $priceEffi14d(u, price_j)$, $priceEffi30d(u, price_j)$.
Quality: Only if freelancers could get benefit from the platfrom, they will keep on working on the platfrom and the lifetime of the platfrom will last long. The higher quality freelancers submit works with, the more they will get benefit. In order to reflect the quality of a freelancer u, we calculate the number of works rewarded, the number of works costs and a success rate of a freelancer. $n_{suc}(u)$ denotes the number of works worked by u that get rewarded, while $n_{fail}(u)$ denote the number of works worked by u that do not get rewarded even when the task is terminated. So the number of works rewarded is $rewardedWorkNum(u) = n_{suc}(u)$; the number of works costs is $workCostNum(u) = n_{suc}(u) - n_{fail}(u)$; success rate is $sucRate(u) = n_{suc}(u)/(n_{suc}(u) + n_{fail}(u))$.
Category-related Quality and Price-related Quality Similarly to **Quality**, we calculate category-related Quality and price-related Quality in different categories or different price bins for a freelancer u : $cateN_{suc}(u, cate_i)$, $cateRewardedWorkCost(u, cate_i)$, $cateSucRate(u, cate_i)$, $priceN_{suc}(u, price_j)$, $priceRewardedWorkCost(u, price_j)$, $priceSucRate(u, price_j)$.

Submitted Work Features.

Relative reputation: The system reputation level of a work is the the system reputation level of the freelancer who submits the work. We use $rank(w_{ue}^{mn}, task_e^m)$ denote the system reputation level rank of work $repRank(w_{ue}^{mn})$ compared with other works of the task $task_e^m$. $repRank(w_{ue}^{mn}, task_e^m)$ is the number of works whose system reputation level is bigger then the system reputation level of w_{ue}^{mn}. And we use $num(task_e^m)$ denote the total number of works submitted to the task $task_e^m$. Relative reputation is then denoted as $relativeRep(w_{ue}^{mn}) = \frac{repRank(w_{ue}^{mn}, task_e^m)}{num(task_e^m)}$.
Response time: Response time $responseDate(w_{ue}^{mn})$ is the count of days after a task $task_e^m$ is published.
Relative response time: Similar to relative reputation, we use $responseRank(w_{ue}^{mn})$ denote the time rank of a work w_{ue}^{mn} submiteed to $task_e^m$. The earlier a work is submitted the lower its ranker is. And the relative response time is $relativeResponseDate(w_{ue}^{mn}) = \frac{responseRank(w_{ue}^{mn}, task_e^m)}{num(task_e^m)}$.

Employer Features.

Employer features describe the preference of a employer e to make a choice.
Reputation preference: We calculate the median and mean system reputation level of the works selected by the employer e: $medianRep(e)$, $meanRep(e)$.
Relative reputation preference: We calculate the median and mean relative system reputation level of the works selected by the employer e: $medianRelativeRep(e)$, $meanRelativeRep(e)$.
Response time preference: We calculate the median and mean system response time works selected by the employer e: $medianResponseDate(e)$, $meanResponseDate(e)$.

Graph-based Feature.

In order to reflect the confidence of worker's work ability and employer's evaluation ability, we define $authority_u$ for workers and $authority_e$ for employers. We run iterative computations on $authority_u$ and $authority_e$. In each iteration, $authority_u = \frac{\sum\sum 1(r(w_{ue}^{mn})=1)\cdot authority_e}{\sum\sum 1(r(w_{ue}^{mn})=1)}$ and $authority_e = \frac{\sum\sum 1(r(w_{ue}^{mn})=1)\cdot authority_u}{\sum\sum 1(r(w_{ue}^{mn})=1)}$. However, due to the data sparse problem, we cluster the employer into employer set according to the number of tasks they have published. So e' is the cluster where e belongs to, and then $authority_u = \frac{\sum\sum 1(r(w_{ue'}^{mn})=1)\cdot authority_{e'}}{\sum\sum 1(r(w_{ue'}^{mn})=1)}$ and $authority_{e'} = \frac{\sum\sum 1(r(w_{ue'}^{mn})=1)\cdot authority_u}{\sum\sum 1(r(w_{ue'}^{mn})=1)}$. Till iterations to converge, we will get the authority score as a feature for each worker and employer.

4. METHODOLOGY

In this section we will firstly describe how we use the features to combine a final input feature matrix. And then we show the baselines we used in this paper to compare with our work. Finally, we will explain our method and the evaluations.

4.1 Feature matrix

Suppose that there is a work w_{ue}^{mn}, which is submitted to e's task $task_e^m$ by u. And $task_e^m$'s category is $cate_i$ and it belongs to price bin $price_j$. Then we get the three type of features: work features $\vec{w}$, freelancer features $\vec{u}$ and employer features $\vec{e}$ separately. For $\vec{u}$ and $\vec{w}$, we directly use the values of the features we showed in section 3.2 as follows:

$$\vec{w} = \begin{vmatrix} relativeRep(w_{ue}^{mn}) \\ responseDate(w_{ue}^{mn}) \\ responseRank(w_{ue}^{mn}) \end{vmatrix}^T$$

$$\vec{u} = \begin{vmatrix} cateNum(u) \\ workNum(u, cate_i) \\ workNumRate(u, cate_i) \\ priceNum(u, price_j) \\ priceRate(u, price_j) \\ sysLevel(u) \\ effi7d(u) \\ effi14d(u) \\ effi30d(u) \\ cateEffi7d(u, cate_i) \\ cateEffi14d(u, cate_i) \\ cateEffi30d(u, cate_i) \\ priceEffi7d(u, price_j) \\ priceEffi14d(u, price_j) \\ priceEffi30d(u, price_j) \\ rewardedWorkNum(u) \\ rewardedWorkCost(u) \\ sucRate(u) \\ cateN_{suc}(u, cate_i) \\ cateRewardedWorkCost(u, cate_i) \\ cateSucRate(u, cate_i) \\ priceN_{suc}(u, price_j) \\ priceRewardedWorkCost(u, price_j) \\ priceSucRate(u, price_j) \\ authority_u \end{vmatrix}^T$$

While the employer features represent the incline of an employer e, so we should use these features with the attributions of the candidate work. We calculate the difference and absoluate difference between feature of an employer and the corresponding feature of one submitted work. For example, we will use $medianRep(e) - sysLevel(u)$, $meanRep(e) - sysLevel(u)$ and $abs(medianRep(e) - sysLevel(u))$, $abs(meanRep(e) - sysLevel(u))$ to represent the variance between the employer incline and the current work.

$$\vec{e} = \begin{vmatrix} medianRep(e) - sysLevel(u) \\ meanRep(e) - sysLevel(u) \\ medianRelativeRep(e) - relativeRep(w_{ue}^{mn}) \\ meanRelativeRep(e) - relativeRep(w_{ue}^{mn}) \\ medianResposeDate(e) - responseDate(w_{ue}^{mn}) \\ meanResposeDate(e) - responseDate(w_{ue}^{mn}) \\ abs(medianRep(e) - sysLevel(u)) \\ abs(meanRep(e) - sysLevel(u)) \\ abs(medianRelativeRep(e) - relativeRep(w_{ue}^{mn})) \\ abs(meanRelativeRep(e) - relativeRep(w_{ue}^{mn})) \\ abs(medianResposeDate(e) - responseDate(w_{ue}^{mn})) \\ abs(meanResposeDate(e) - responseDate(w_{ue}^{mn})) \\ authority_e \end{vmatrix}^T$$

The Feature vector of work w_{ue}^{mn} is consist of the three vectors:
$vectorW(w_{ue}^{mn}) = [\vec{w} \quad \vec{u} \quad \vec{e}]$.
And with certain count of works, we get the final feature matrix : $featureMatrix(W) = \begin{vmatrix} vectorW(w_{uem1}) \\ vectorW(w_{uem2}) \\ \vdots \\ vectorW(w_{ue}^{mn}) \end{vmatrix}$.

4.2 Baseline

We use five different means as baselines to compare with our model. They are random selection, reputation $sysLevel(u)$ in ascending order and descending, submitted rank $responseRank(w_{ue}^{mn})$ in ascending order and descending. We generate a rank list from all the candidate works submitted to one task $task_e^m$ by the five different means separately. And then we choose the top-s works to be the selected works for the task $task_e^m$, where s is demanded number of works corresponding to $task_e^m$.

4.3 Method and Evaluation

Firstly we use the feature matrix we present in section 3 as features and actually situation that whether a works w_{ue}^{mn} is chosen ($r(w_{ue}^{mn}) = 1$) or not ($r(w_{ue}^{mn}) = 0$) as label data, to train a regression model by three different methods, decision tree, random forest and logistic regression. Then similar to how we deal with the baseline, we use the regression of the model we trained with the feature matrix of test data as input to generate a rank list for the submitted works of each task separately and choose the top-s works according to each rank list.

And to evaluation the result of our feature model and the baseline, we induce NDCG@k(Normalized Discounted Cumulative Gain) and MAP@k(Mean average precision) from information retrieval. And in order to implement the evaluation, we alternate the top-m works to top-$(s + k)$.

5. EXPERIMENTS

5.1 Feature selection

We choose two categories to test all the features we proposed in Section 3.2. Then we implement experiments of pointwise ranking and pairwise ranking by decision tree, linear regression, logistic regression and random forest on these two categories with different combinations of features groups.
First of all, we implement experiments on each of the feature group separately and evaluate the result by MAP@k and NDCG@k. We present the k=4's result in the table below.$sucR$, eff, ep, $repu$, $time$, $graph$ represent sucRate, efficiency, employer preference, reputation, submit time, graph-based separately.

group	method	pointwise		pairwise	
	evaluation	MAP@4	NDCG@4	MAP@4	NDCG@4
$sucR$					
eff					
ep					
$repu$					
$time$					
$graph$					

After experiments on feature group separately, we rank the feature group by MAP@k and NDCG@k descendantly. Then we combine top 1,2,...,all groups from the ordered feature groups as features for experiments and evaluate the effectiveness of feature groups, and try to find a trade-off point of the features we use in our model.

group	method	pointwise		pairwise	
	evaluation	MAP@4	NDCG@4	MAP@4	NDCG@4
TOP-1					
TOP-2					
TOP-3					
TOP-4					
TOP-5					
all					

We find TOP-X(TO BE DONE) features perform best, and

we will use these features in the experiments below.

5.2 Evaluate the query depth effectiveness

Considering that MAP@k and NDCG@k have correlation with the lenght of the query result, we need to add experiments to compare our methods with baseline, expecially the random ranking baseline. Then we will find out how our method will perform as more works are submited. We set five groups of comparison experiments by the number of works pertask, which are 1-5, 6-10, 11-20, 21-30, 31-50, over 50 works per task separately.

num of works/task	method	pointwise		pointwise	
	evaluation	MAP@4	NDCG@4	MAP@4	NDCG@4
1-5					
6-10					
11-20					
21-30					
31-50					
over 50					

5.3 Exception analysis

Some features such as $relativeRep(w_{ue}^{mn})$ dose not perform well in the feature selection. We will compare best features with $relativeRep(w_{ue}^{mn})$ or $rank(w_{ue}^{mn}, task_e^m)$, which intuitively should not perform better then $relativeRep(w_{ue}^{mn})$, in different category and different learning methods.