Recommendation System for Stack Exchange

Anish Kelkar
apk2129@columbia.edu

Divi Khanna
dk2745@columbia.edu

Pranav Bhalla
pb2538@columbia.edu

Richa Singh
rs3375@columbia.edu

Abstract

Stack Exchange Website is prime representative of the online Q&A websites' community. Such services are extremely popular and they attract tens of millions of users daily. Due to such large scale participation, it is a challenge to ensure that all the questions get answered in a timely manner. As a result, it could be very efficient to route a question to a potential answerer as soon as it is posted. Our objective in this report is to develop a personalized recommendation system that identies questions which a particular user might be interested in answering at a given point in time. We propose a recommendation system for routing a question to the most desirable group of expert users who would be willing to provide useful answers to that question.

Contents

1 Introduction

Question-answering sites such as StackOverflow or Yahoo Answers provide users with a wealth of questions that they can answer. However, some questions might be more relevant to the user than others - namely, they cover topics that users have indicated that they are interested in, by answering other questions on that topic. Our objective is to develop a personalized recommendation system that identifies questions which a particular user might be interested in answering at a given point in time. Online forums contain huge amounts of valuable user-generated content. In current forum systems, users have to passively wait for other users to visit the forum systems and read/answer their questions. The user experience for question answering suffers from this arrangement. We look to address the problem of “pushing” the right questions to the right persons, with objective to obtain quick, high-quality answers, thus improving user satisfaction and improving the quality of the content. We propose a framework for the efficient and effective routing of a given question to the top potential experts (users) in a forum, by utilizing both the content and structures of the forum system.

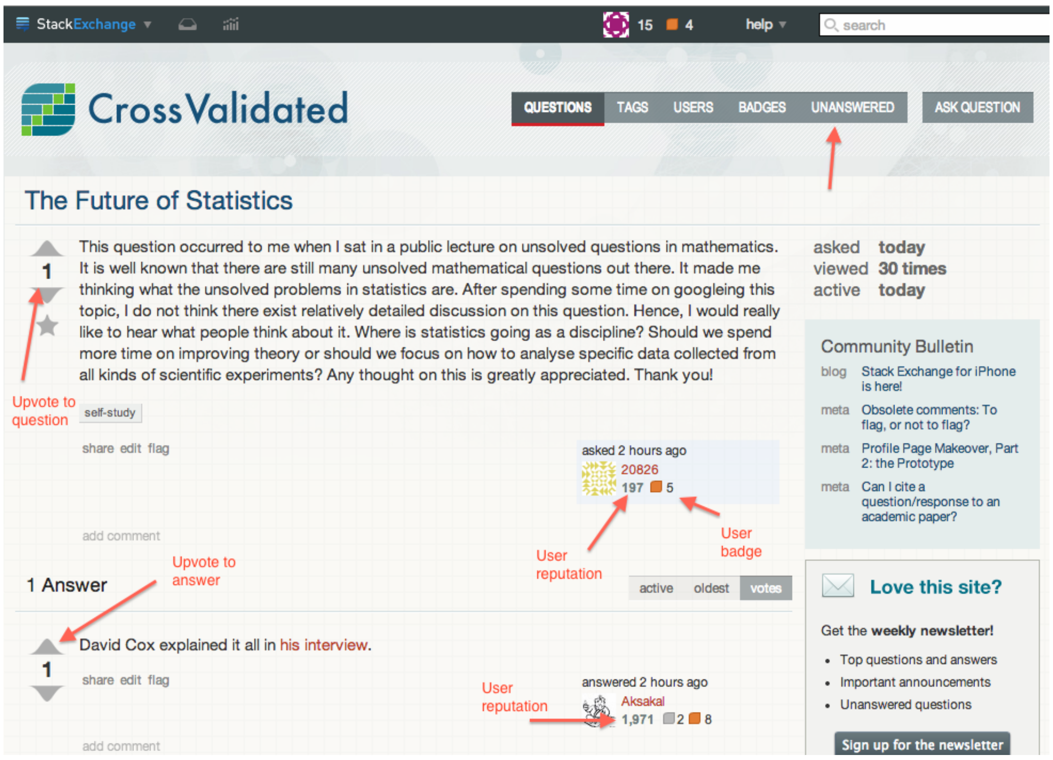
1.1 Stack Exchange

Stack Exchange is a network of question and answer websites on diverse topics in many different fields. Each site covers questions and answers on a specific topic. The sites are modeled after Stack Overflow, a forum for computer programming questions that was the original site in this network. Today there are about 120 network sites, and the community encourages building more sites in the network. As of April 2014, Stack Overflow has over 2,700,000 registered users and more than 7,100,000 questions.

Figure 1: **Top Sites in Stack Exchange Network**

The primary purpose of each Stack Exchange site is to enable users to post questions and answer them. Stack Exchange is a community oriented *cooperative and collaborative learning* platform that allows users to communicate online and contribute to their area of expertise. It is *self-moderated* by the community of users who participate in asking and answering questions. Along with posting questions and answers, users can add comments to posts and can even edit text written by others. Each Stack Exchange site has a *”meta”* section where users can settle disputes, because the self-moderation system can lead to significant arguments.

Users are subject to a reputation award process. which aids in the process of self moderation. [1] Users can vote on both answers and questions, and through this process users earn reputation points, a form of gamification[1]. Users receive privileges by collecting reputation points, ranging from the ability to vote and comment on questions and answers to the ability to moderate many aspects of the site. All user generated content (questions, comments and answers) posted on the Stack Exchange Network is licensed under a Creative Commons license Attribution Share Alike, and this has helped in creating a knowledge resource free for the entire world to use.



The Future of Statistics

1

This question occurred to me when I sat in a public lecture on unsolved questions in mathematics. It is well known that there are still many unsolved mathematical questions out there. It made me thinking what the unsolved problems in statistics are. After spending some time on googleing this topic, I do not think there exist relatively detailed discussion on this question. Hence, I would really like to hear what people think about it. Where is statistics going as a discipline? Should we spend more time on improving theory or should we focus on how to analyse specific data collected from all kinds of scientific experiments? Any thought on this is greatly appreciated. Thank you!

self-study

asked 2 hours ago
20826
197 5

asked today
viewed 30 times
active today

Community Bulletin

blog	Stack Exchange for iPhone is here!
meta	Obsolete comments: To flag, or not to flag?
meta	Profile Page Makeover, Part 2: the Prototype
meta	Can I cite a question/response to an academic paper?

1 Answer

1

David Cox explained it all in his interview.

answered 2 hours ago
Aksakal
1,971 2 8

Get the **weekly newsletter!**

- Top questions and answers
- Important announcements
- Unanswered questions

Figure 2: **An Sample Question Screen**

2 Proposed Recommendation System

2.1 Objective

Stack Exchange provides a user (U) wealth of questions that he can answer. But some questions might be more relevant to the this user than others. A personalized recommendation system matches an incoming questions (Q) with a list of users (L) who might be more interested in answering it.

This system is built with the objective to ensure that a new questions is recommended to somebody who has the capability and willingness to answer that question. In terms of Bayes formulation, a recommendation system will calculate:

$$\Pr(Q|U) = \frac{\Pr(U|Q)\Pr(Q)}{\Pr(Q|U)\Pr(U) + \Pr(Q|\neg U)\Pr(\neg U)} \tag{1}$$

$\Pr(Q|U)$: Probability a question is answered by a user

$\Pr(U|Q)$: Probability a user wants to answer a given question (Measure of interest by user)

$\Pr(Q)$: Probability a question comes to user (Matched to user based on previous expertise)

$P(U)$: Probability (complex) that a user is selected by the recommendation system (Selection made on basis of *Rank* given to user based on the features that can contribute in making U a likely choice to answer the question).

[1]Gamification is the use of game thinking and game mechanics in non-game contexts to engage users in solving problems. Gamification has been studied and applied in several domains, such as to improve user engagement, physical exercise, return on investment, data quality, timeliness, and learning.

However, there are certain areas of concern before developing a simplistic approach towards mathematical manipulation of conditional probabilities.

1. We assume that there are topics in which user U shows more interest historically in by answering other questions in related topics. Then the actual history of answers given by U serve as a proxy for gauging interest in the incoming question Q.
2. User (U) may be a seasoned user of Stack Exchange and has now become an expert in his area of interest, but not be completely willing to answer questions at all times.
3. There may be few users who are novice in their areas of interest but actively involved in answering questions, making them good prospects even when their expertise may be low.

2.2 Problem Solving Approach

To tackle the above issues in machine learning, we have used a basic recommendor system that follows a hybrid approach combining collaborative filtering and content-based filtering techniques to improve upon our objective.[2]

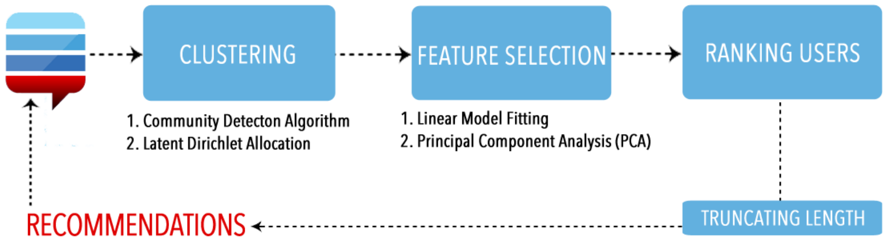

CLUSTERING
1. Community Detecton Algorithm
2. Latent Dirichlet Allocation
FEATURE SELECTION
1. Linear Model Fitting
2. Principal Component Analysis (PCA)
RANKING USERS
TRUNCATING LENGTH
RECOMMENDATIONS

Figure 3: **Recommendation System**

2.2.1 Clustering

1. *Collaborative filtering:* We use a set of clustering algorithms to cluster the tags in the system, based on the similarity measure of co-occurance of tags in dataset. Clustering based on how tags co-occur has been extensively researched [5] [6], and used in systems like *Twitter* and *Flickr*. Tags that frequently co-occur together in questions are believed to be related and can be clustered together. It is important to highlight here that on Stack Exchange every new question must be assigned atleast one tag before it can be posted to the online community. A class of algorithms that cluster objects based on some similarity measure are known as community detection algorithms, and we have used these for creating clusters of tags that co-occur together.

2. *Content-based filtering:* Our methodology finds users who have previously answered questions on some given tags. As we already have a cluster of tags from the previous step, we use a normal matching approach to map tags cluster with users and create user communities who have the expertise in answering those tags. By the use of additional tags that belong to a cluster, we expand our list of probable users who may be good matches for an incoming question with tags in the tag cluster. Our recommendation system expands the scope of the question so that we may find users with relevant knowledge and known history, who would be able and interested to answer the question earlier and better.

   Currently we use community detection algorithms present in *igraph* package to cluster users. Tags assigned by users are clustered, making it a supervised clustering method. In future, we plan to use Latent Dirichlet Allocation (LDA) to topic model questions based on body of posts. This will create an unsupervised list of tags for assigning to questions and a community detection algorithm can be used to cluster these new tags as well.

[2]Infering people's choice and decisions by learning from datasets is an active area of research and no one approach is better. Every datasets has features that need to be understood before applying machine learning methods. As beginners, we may have overlooked some features or made mistakes while applying others.

2.2.2 Feature Selection

Feature selection is a name given to the method in which we try to predict the prominence of features associated with users that can be used to rank them for the recommendation system. Simply put, if we have a list of 100 users who can probably answer a question, we will recommend to only a small list of those users such that extra-recommendation does not hamper the system. We use two methods to select and rank users based on the features in the dataset.

- *Linear Model Fitting:* First is a linear model fitted over features that computes a score for each user based on regression weights. Users are then ranked based on the score. We compute a score for each of the user and rank them.Relying only on the tags extracted from the question, we would be stuck with a small group of users. But when we have more tags from the clustering we are able to evaluate a larger community of users whom we can rank for this specific question and then push this question to the top few.
- *PCA:* We assume that rank of users is a linear combination of features such as users' activeness, reputation, received upvotes and other scores. We create a feature matrix and run a principle component analysis to select the best features based on the fundamental of maximum variance. We assume here that the principal component showing maximum variance contains more information and hence is a better predictor of user behavior.

2.2.3 Model Validation

Stack Exchange datadump is hosted on Internet Archives and is updated periodically to merge new Q&A data. At the beginning of this project in March we downloaded the entire datadump to make our model. The current data has been updated substantially in two months and we simply subset the new dataset against the old dataset to generate our test data. This traverses the need for us to divide the data into training and testing. The March dataset has been completely used to train the recommendation system.

3 Data

In this project, we are focused on Sustainable Living dataset of Stack Exchange. All domains under the Stack Exchange umbrella, such as Stack Overflow, Mathematics, AskUbuntu etc. have their data set as a set of XML documents. Each dataset consists of the following files:

XML Files	**Important Fields used in this Project**
Posts.xml	Id, PostTypeId (question, answer, comment), AcceptedAnswerId, OwnerUserId, Title, Tags, Body
Users.xml	Id, Reputation
Comments.xml	Id, PostId, Score, Text
Badges.xml	Id, UserId, Name
PostHistory.xml	Id, PostHistoryTypeId, PostId
PostLinks.xml	Id, LinkTypeId, PostId
PostLinks.xml	Id, LinkTypeId, PostId
Votes.xml	Id, VoteTypeId (UpMod, DownMod), PostId

Table 1: Simplified Data Schema of Stack Exchange

Among these files, we have mainly made use of the Posts.xml and Users.xml since they contain most of the information we needed. Data was munged in different ways to create lists, hashes, clusters that are used when running the clustering, feature selection and validation procedures.

Posts.xml: The "Id" field uniquely identifies a post in the data set, and the "PostTypeId" identifies the type(Type-1 is Question while Type-2 an answer). "AcceptedAnswerId" is valid only in a question type Post, in which it is "Id" of the answer (another Post) that has been "accepted" by the questioner. "OwnerUserId" is the id of the user who made the post. "Title" and "Body" contain the actual content of a post. "Tags" contain the tags that a user has marked a question with.

Users.xml: "Id" is the unique identifier for a user and used as "OwnerUserId" in the Posts.xml file. "Reputation" is the score that a user has based on the activities on the website, reputation can be increased by asking questions, giving answers and other activities that contribute to maintaining and enhancing the content on the exchange.

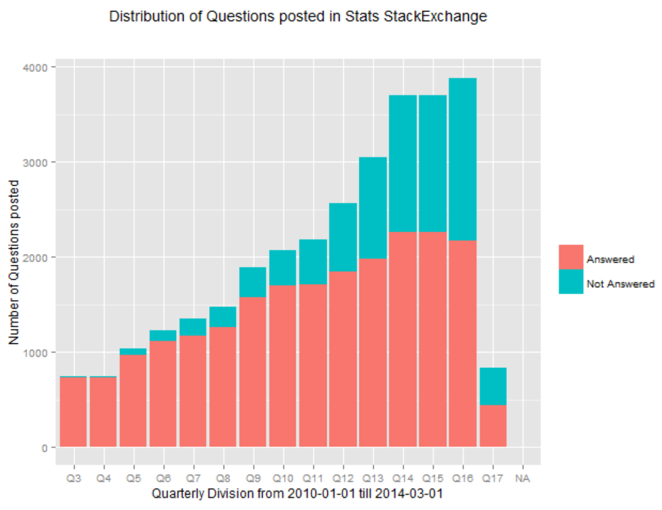


Figure 4: **Distribution of unanswered questions in Stats Stack Exchange over last three years**

4 Clustering

4.1 Tag Clustering

Studying relationships between keyword tags on social web-sites has become a popular topic of research, both to improve tag suggestion systems and to discover connections between the concepts that the tags represent. Existing approaches have mostly relied on tag-occurance. [4]

```
Tags by pairwise co-occurrence (displaying random 10)
            [,1]                      [,2]
   [1,]  "<husbandry>"             "<food>"
   [2,]  "<husbandry>"             "<insects>"
   [3,]  "<food>"                  "<insects>"
   [4,]  "<food>"                  "<permaculture>"
   [5,]  "<food>"                  "<animals>"
   [6,]  "<food>"                  "<husbandry>"
   [7,]  "<food>"                  "<fodder>"
   [8,]  "<permaculture>"          "<animals>"
   [9,]  "<permaculture>"          "<husbandry>"
  [10,]  "<permaculture>"          "<fodder>"
```

A tag co-occurance matrix is best visualized as a network graph, with each tag placed on the node and edges weighted by the co-occurrance of tags in the dataset.

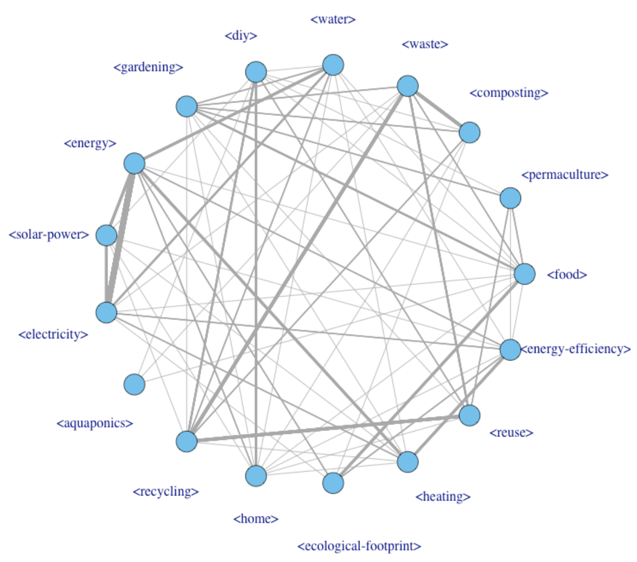


Figure 5: **Top 10% tags with Edges Weighted by Frequency of Co-occurance**

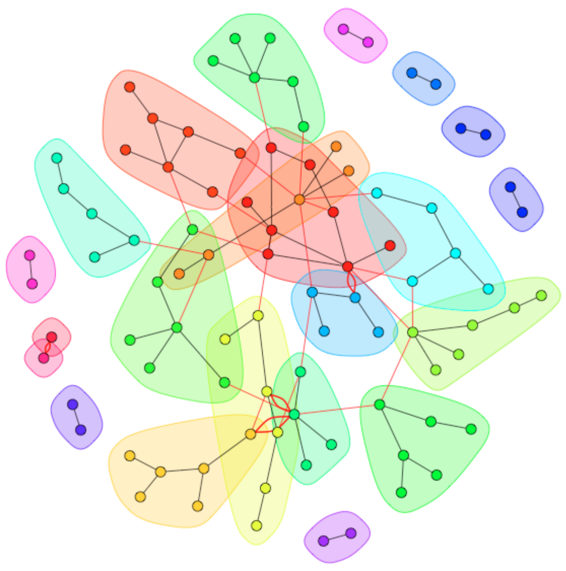


Figure 6: **Community Detection for 100 Random Tags**

4.2 Community Detection Algorithm

We use the state-of-art network analysis package *igraph*, which has 8 community detection algorithms[3]. A detailed analysis and comparison of these algorithms is also given extensively in this paper [2]. After trying out a bunch of them such as *fast.gredy, eigenvector, edge.betweeness*,we chose *infomap.community* originally developed by M. Rosvall and C. T. Bergstrom [3], as the best algorithm for community detection suited to the needs of Stack Exchange dataset.

The core of the algorithm follows closely the Louvain method. The Louvain method is a simple, efficient and easy-to-implement method for identifying communities in large networks. The method has been used with success for networks of many different type (see references below) and for sizes up to 100 million nodes and billions of links. The analysis of a typical network of 2 million nodes takes 2 minutes on a standard PC. The method unveils hierarchies of communities and allows to zoom within communities to discover sub-communities, sub-sub-communities, etc. It is today one of the most widely used method for detecting communities in large networks. With this algorithm, a fairly good clustering of the network can be found in a very short time. A network graph showing tags communities of 100 random tags is visualized in Figure 5. [4]

4.2.1 Communities of Tags & Users

We have created hash tables with tag clusters, mapping them to users who have answered questions on tags present in the cluster. This matching of tags with users create communities of users who we believe will be experts to answer questions in that community. For example- community "4" has the following given list of "tags" and "users" in it.

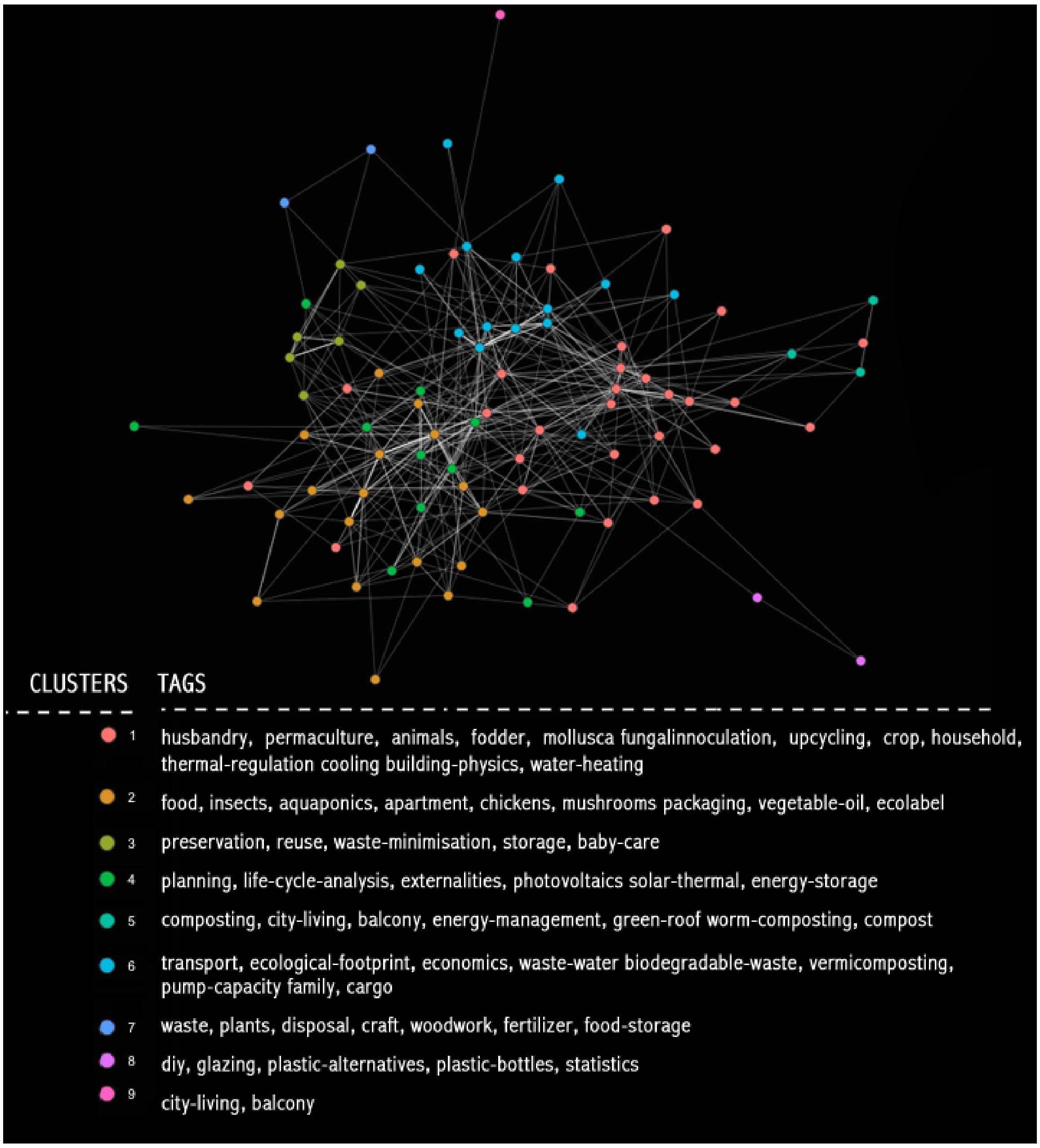
Community No.	**Tags**
4	"husbandry","food","insects","animals","fodder","mollusca"
33	"housing", "warming", "survival"
17	"permaculture", "fungal-innoculation", "pets", "pest-management"

Table 2: Community No. and Tags present in them

[3]The analysis and results of these 8 community detection algorithms are given in log for reference
[4]It is difficult to visualize for complete network graph, but the result is still present in log.

Community No.	Users
4	“130”, “85”, “185”, “130”, “66”, “10”, “1077”, “1073”, “1065”
33	“462”,”180”,”15”, “48”, “86”, “10”, “404”, “400”, “48”, “86”, “48”, “86”
17	“624”, “85”, “888”, “624”, “778”, “794”, “751”, “749”, “514”, “15”, “130”, “488”

Table 3: Community No. and Users present in them



CLUSTERS		TAGS
●	1	husbandry, permaculture, animals, fodder, mollusca fungalinnoculation, upcycling, crop, household, thermal-regulation cooling building-physics, water-heating
●	2	food, insects, aquaponics, apartment, chickens, mushrooms packaging, vegetable-oil, ecolabel
●	3	preservation, reuse, waste-minimisation, storage, baby-care
●	4	planning, life-cycle-analysis, externalities, photovoltaics solar-thermal, energy-storage
●	5	composting, city-living, balcony, energy-management, green-roof worm-composting, compost
●	6	transport, ecological-footprint, economics, waste-water biodegradable-waste, vermicomposting, pump-capacity family, cargo
●	7	waste, plants, disposal, craft, woodwork, fertilizer, food-storage
●	8	diy, glazing, plastic-alternatives, plastic-bottles, statistics
●	9	city-living, balcony

Figure 7: **Network Graph with 9 Large Clusters**

In the above figure we are able to see the clusters for the Sustainiblity data set using the *infomap.community*. They are not spatially arranged together since we used co-occurence as a measure for similarity, and it is not representible spatially.

5 Feature Subset Selection

Assumption: Rank of users is a linear combination of features such as users' activeness, reputation, received upvotes, scores etc. We use linear model fit and PCA analysis to make decision about how best to capture these features while creating a ranked list of users.

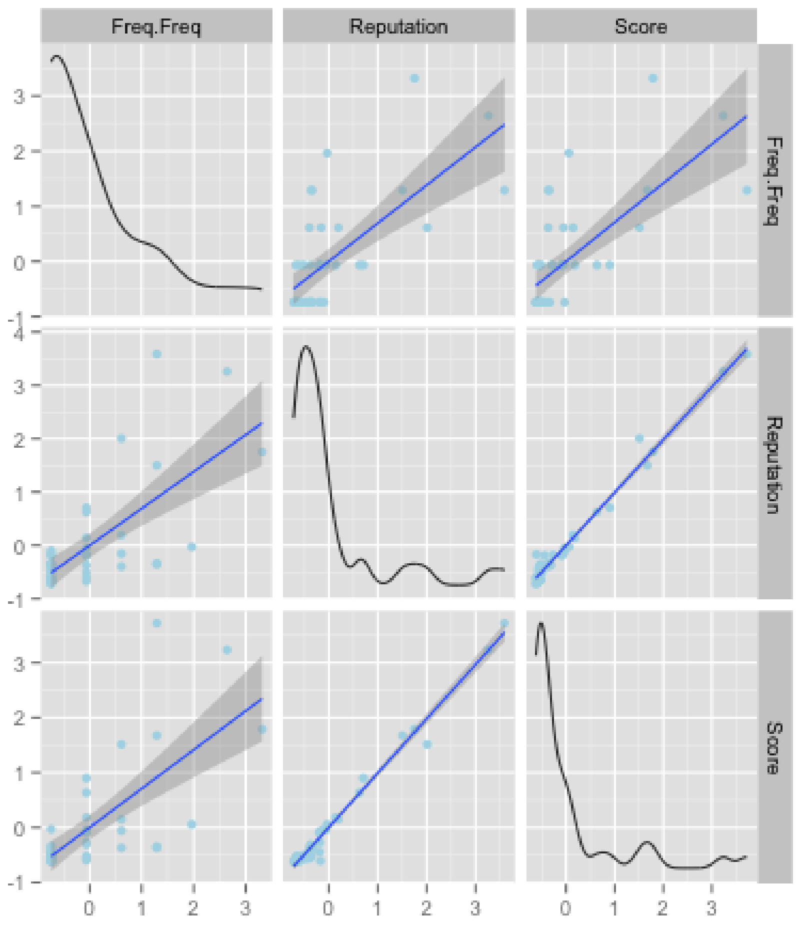


Figure 8: **Scatterplot for Features**

5.0.2 Method 1: Linear Model Fitting

- To incorporate both reputation and tag score to rank a user, we needed to decide whether these features were relevant in finding good users to recommend new questions to. We used linear regression to come up with weights for both of these. The output(y) for this regression is a binary 1/0 value indicating whether a given user has answered a question, this value is mostly 0 since very few users from the whole set answer a particular question. The linear fitting here more closely approximates a classification problem in which we are trying to classify some user to possible answeree (1) or not relevant (0).

```
lm(formula = output ~ reputation + score, data = modelDF)
Coefficients:
(Intercept)   reputation        score
 -4.620e-03    9.634e-06    2.794e-02
```

We obtained the above weights for the Sustainibility data set, which indicated that tag score was a much better predictor than reputation. But even with the weights available we had no way of limiting ranked users, since in the real world we want to limit the number of users to whom we push a new question. This is described in detail in the next section "Ranking".

- *Advantages:* The model is simple since the number of features is less and the output is well defined. It presents an accurate picture that is consistent with our expectations. Since reputation is a static measure that doesn't change across questions, we do not expect it to be as significant as tag score, but more of a tie breaker when tag score for two users are the same.

- *Disadvantages:* This is a binary classification problem that is suited more to methods like LDA (Linear Discriminant Analysis), Naive Bayes which usually perform better at classification tasks.

5.0.3 Method 2: Principal Component Analysis

- The general form for the formula to compute scores on a components created using PCA is:

$$c_1 = \beta_{11}x_1 + \beta_{11}x_2 + ... + \beta_{1p}x_p$$

c_1 =the subject's score on principal component 1 (the first component extracted)
β_{1p} = the regression coefficient (or weight) for observed variable p, as used in creating principal component 1
x_p = the subject's score on observed variable p

```
Code Result: Principal Components Analysis (Random User Community)
Standardized loadings (pattern matrix) based upon correlation matrix

            PC1    PC2    PC3 h2       u2
Freq.Freq   0.68   0.73   0.00  1 0.0e+00
Reputation  0.96  -0.25  -0.09  1 2.2e-16
Score       0.96  -0.27   0.09  1 1.1e-16

                       PC1  PC2  PC3
SS loadings            2.31 0.67 0.02
Proportion Var         0.77 0.22 0.01
Cumulative Var         0.77 0.99 1.00
Proportion Explained   0.77 0.22 0.01
Cumulative Proportion  0.77 0.99 1.00
```

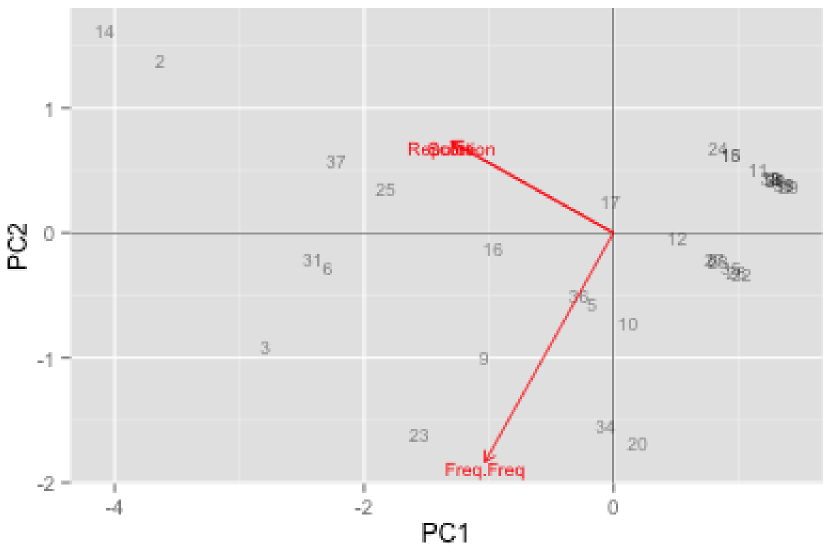


Figure 9: **Biplot for PCA**

- One of the most commonly used criteria for determining the number of factors or components to include is the latent root criterion, also known as the eigenvalue-one criterion or the Kaiser criterion. With this approach, you retain and interpret any component that has an eigenvalue greater than 1.0. With this in mind, we keep only the first principle component PC1 in our analysis. In short, we retain components that account for at least 77% of the total variance.

- The regression weights (loadings) are determined using a type of equation called an eigenequation. The eigenvalue is a numeric estimation of how much of the variation each component explains.

```
Standardized loadings (pattern matrix) based upon correlation matrix

            PC1   h2    u2
Freq.Freq   0.68 0.46 0.539
Reputation  0.96 0.93 0.070
Score       0.96 0.92 0.081

                PC1
SS loadings     2.31
Proportion Var 0.77

Test of the hypothesis that 1 component is sufficient.
The degrees of freedom for the null model are  3  and
objective function was  3.72
The degrees of freedom for the model are 0  and the objective function was
0.93
The total number of observations was  39  with MLE Chi Square =  33.07
with prob <  NA

Fit based upon off diagonal values = 0.95
```

- We create factor scores by creating a rotated solution. Factor scores represent the linear composite of weighted observed variables, and we have used the factor scores to rank users within a community.

```
Code Result: Factor Scores

          PC1
1   0.09979495
2  −0.53193338
3  −0.11569072
4   3.28292671
5  −0.62701537
6   1.51187505
7  −0.56132168
8  −0.68266328
9  −0.67431464
10 −0.47890094
```

- Users are ranked based on features, such as Frequency of answering questions, Reputation and Scores. A rank calculated by PCA Factor Scores is then used to rank users. We are selecting only those for recommendation who have a positive rank (factor score).

```
Code Result: Ranking Users Based on Features using PCA Factor Scores

   OwnerUserId Freq.Freq Reputation Score       Rank
4          130         6       4064   308 3.28292671
17          48         4       4398   347 3.14724566
37          85         7       2527   193 2.40194348
28          66         4       2272   184 1.60531347
6           15         3       2787   171 1.51187505
11         400         5        712    55 0.65016960
18         486         2       1461   122 0.57712768
24          60         2       1381   101 0.44927329
41          96         3        941    62 0.32583526
```

- *Advantages:* Principal Component Analysis is an unsupervised learning method which is adapted to our situation of using feature selection methods to rank users. Compared to other linear model fitting solutions, PCA has an advantage of being unsupervised. Given our motive to use topic modeling for clustering users, PCA provides a very good fit for an unsupervised recommendation system.
- *Disadvantages:* We understand that unsupervised problems may not be easy to solve, and hence our efficiency will reduce dramatically by using topic models for clustering and PCA scores for ranking.

6 Ranking and Recommendation

A simple feature selection generates a huge list of users, which would overcrowd the system of recommendation and will be counter-productive to our objectives. So not only does our system need to come up with a list of potential answerees, it needs to keep the list to minimal. For our purpose, we have implemented separate algorithms to find a good list length of potential users who can answer a question.

6.1 Method 1: Linear Model Fitting

To limit the number of suggested users, we used a truncating length, which is the top number of users we pick from the ranked users. We used the weights from the above regression and again applied our clustering algorithm to find the positions in the ranked order at which we find the user who answered the question.

```
       actualAnsweree         found.at
1           130               3
2            66               3
3            10               5
4            10               4
5            66               2
6            60               2
```

We get a matrix of the above form that indicates the ranked position of the actual answeree of a question. We take the mean of these positions to get a truncating length for the data set. This truncating length is then used on the test data to limit the number of users and hence get a more accurate view of the efficiency of the recommendation system.

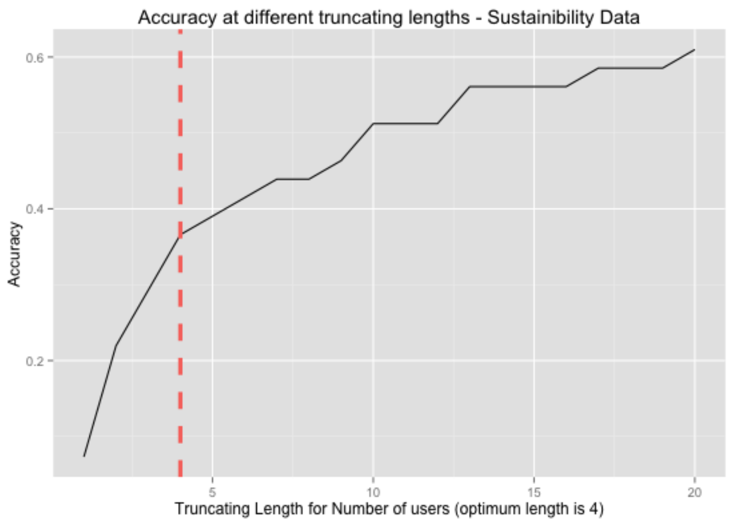


Figure 10: **Accuracy Vs Truncation in Linear Fitting**

In the above figure we are able to see the accuracy at different lengths, along with the accuracy at a mean truncating length of 4, chosen by averaging over the test data.

6.2 Method 2: Principal Component Analysis

We truncated the number of users for recommendation by taking only positive factor scores obtained from PCA. A plot for this gives a better idea of truncation used in this method. Users are truncated below factor scores of zero, ie. only positive factor score users are used for recommendation

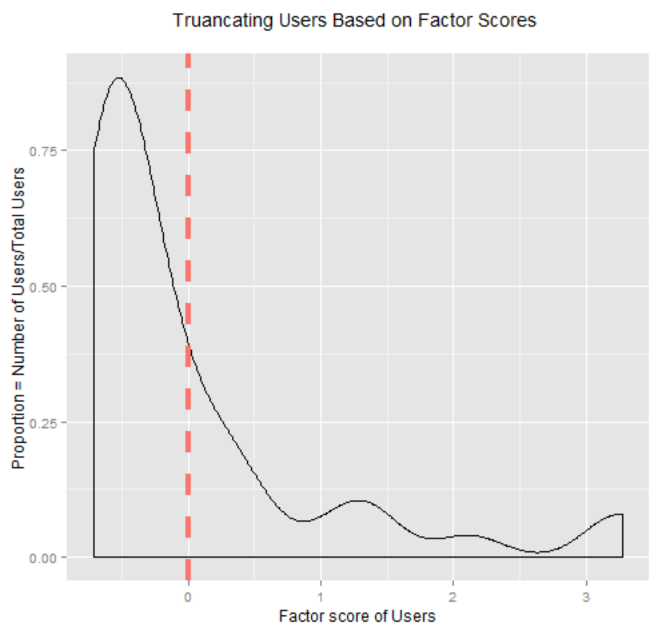
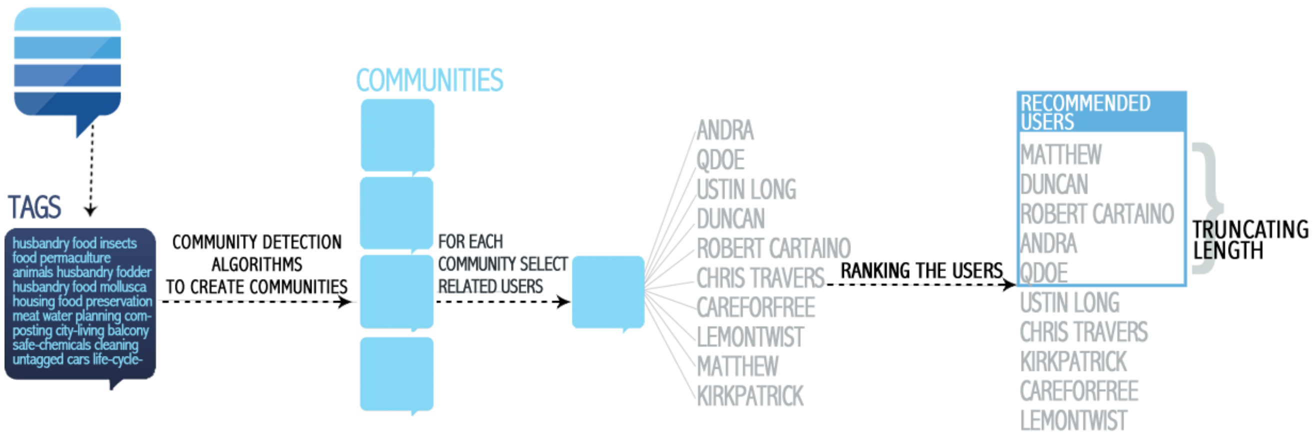


Figure 11: **Truncation in PCA done at factor score of zero**

7 Summary & Results

A process diagram to summarizes working of the algorithm used for this recommendation system:



TAGS
husbandry food insects food permaculture animals husbandry fodder husbandry food mollusca housing food preservation meat water planning composting city-living balcony safe-chemicals cleaning untagged cars life-cycle-
COMMUNITY DETECTION ALGORITHMS TO CREATE COMMUNITIES
COMMUNITIES
FOR EACH COMMUNITY SELECT RELATED USERS
ANDRA
QDOE
USTIN LONG
DUNCAN
ROBERT CARTAINO
CHRIS TRAVERS
CAREFORFREE
LEMONTWIST
MATTHEW
KIRKPATRICK
RANKING THE USERS
RECOMMENDED USERS
MATTHEW
DUNCAN
ROBERT CARTAINO
ANDRA
QDOE
USTIN LONG
CHRIS TRAVERS
KIRKPATRICK
CAREFORFREE
LEMONTWIST
TRUNCATING LENGTH

Figure 12: **Recommendation System Process Diagram**

7.1 Linear Fitting

Validation Method	Accuracy
Users who answer all the tags	2.4%
Users who answer atleast one tag	51.2%
Users who answer atleast one tag from expanded list	63.4%

Table 4: Basic validation methods accuracy on Sustainibility data

In the above table, we see that the naive matching method of finding users who have answered questions on all the tags of a new question has very poor accuracy (2.4%). Again, without using the clustering but picking users who have answered atleast one question with tags matching the new questions tags we get a high accuracy (51.2%). Now using the clustering and expanding the set of users by finding associated tags for a question, over and above the original tags, we get an accuracy of 63.4%. These results seem significant but they have a huge downside that they contain very large number of users to recommend questions to. As has been discussed previously such a practice is discouraged since it clutters the system and doesn't make effective use of the clustering we have performed.

Validation Method	Accuracy
Ranked Users (truncated)	7.3%

Table 5: Ranked validation on Sustainibility data with truncating length set to 10%

Ranking users by their reputation and tag score, adjusted by their weights from the linear fitting, and then truncating them to pick the top 10% gives an accuracy of 7.3%. This value is much lesser than the accuracy for the last methods, but it doesn't clutter the system. But neither does it give us significant results.

Validation Method	Accuracy
Ranked Users (truncated)	36.6%

Table 6: Ranked validation on Sustainibility data with truncating length set to 4

Finally, we have found a good truncating length (mean) on the training data, and then used that to truncate the ranked users. This method gives an accuracy of 36%, which is much better than the previous truncating method. Hence this methodology is the ideal, since it doesn't clutter the system and gives a fairly accuracy.

7.2 Principal Component Analysis

After performing PCA on the data set, we obtained factor scores per user per community, i.e. each user that belonged to a community had a factor score within that community. While ranking users for a new question we go through all the communities for the question tags, and get the individual ranks for users from each of the communities. A final score is obtained and users are ranked based on these. We are able limit users since some users also have negative ranks in the community, and are excluded from being considered.

Validation Method	Accuracy
Ranked Users (truncated to include positive scores)	43%

Table 7: Ranked validation on Sustainibility data with PCA factor score

Both these methods have given weights to the features we have used (reputation, tag score, score). Linear fitting gives one set of weights to all the users, whereas PCA gives different ranks to user in different communities, emphasizing their relative importance in the cluster.

7.3 Models: Which, When and Why?

1. *Supervised:* Methods like Linear Model Fitting give us a lot of freedom in playing around with features and learning more about predictors of human behavior through trial and error. Supervised methods in case of Stack Exchange dataset must be preferred solely because Stack Exchange data has all the features to accomodate a supervised learning algorithm.

2. *Unsupervised:* Unsupervised methods are better in cases in which questions don't come with predefined tags. That is why, unsupervised methods have gained prominence in the study of data sets like Twitter in which some tweets do not have tags. In stack Exchange, we dont have questions without tags but still using unsupervised methods gives us a better understanding of prediction accuracy when compared to supervised methods.

8 Suggested Future Actions

1. *Term Document Matrix for LDA:* We have currently used a basic LDA model to predict topics using the text in the body and title of the question. LDA uses a term document matrix which is then decomposed to predict a single topic for a single question. This approach is an unsupervised learning method for predicting tags of questions in stack exchange database. The next step is our project would be to use topic modeling to predict tags and to cluster them based on community detection algorithms. Thereafter, the system works same as it has been described in this project.

2. *Active-time based clustering:* - Right now, our clustering algorithm considers various features to rank the probable answeree. Main of these features are - how frequently a user answers similar question and his/her reputation. In future, we can incoperate the features such as, at what is the preferrable time for the user to be active on Stack Exchange or what is the time zone of the user. Including time-based features, we can gain a stronger predictive ability to suggest users for the questions posted taken into account depending about the time of the post.

3. *Reverse recommendation:*Presently our algorithm works when a new question is posted on the Stack Exchange web site, we push questions at the time they are posted. Another more interesting way of finding questions for users to answers, is to create an algorithm that runs when a user logs to the system. We have his preferences saved and our algorithm can go through the unanswered questions at this time and match them with the user. This method may be possibly better, since our present method pushes a question to users as soon as they are posted. A user may have many questions recommended to him, and their is no way to find which are more relevant to him without running the algorithm again. Effective testing of such a system is only possible in the real world setting and hence we have not included this in our work.

4. *Scalability:* - This system can be optimised and be scaled over larger domains. To be effective, it would not only require a fast processing speed, but also ability to create compelling recommendations to users.

References

[1] J. Atwood. A Theory of Moderation. `http://blog.stackoverflow.com/2009/05/a-theory-of-moderation/`, 2009. [Online].

[2] A. Lancichinetti and S. Fortunato. Community detection algorithms: a comparative analysis. *Physical review E*, 80(5):056117, 2009.

[3] M. Rosvall and C. T. Bergstrom. Maps of information flow reveal community structure in complex networks. Technical report, 2007.

[4] A. Shepitsen, J. Gemmell, B. Mobasher, and R. Burke. Personalized recommendation in social tagging systems using hierarchical clustering. In *Proceedings of the 2008 ACM conference on Recommender systems*, pages 259–266. ACM, 2008.

[5] B. Sigurbjörnsson and R. Van Zwol. Flickr tag recommendation based on collective knowledge. In *Proceedings of the 17th international conference on World Wide Web*, pages 327–336. ACM, 2008.

[6] C. Wartena, R. Brussee, and M. Wibbels. Using tag co-occurrence for recommendation. In *Intelligent Systems Design and Applications, 2009. ISDA'09. Ninth International Conference on*, pages 273–278. IEEE, 2009.