# User Centric Policy Management in Online Social Networks

Mohamed Shehab, Gorrell Cheek & Hakim Touati
*Department of Software & Information Systems*
*College of Computing and Informatics*
*University of North Carolina*
*Charlotte, NC*
{*mshehab, gcheek, htouati*}*@uncc.edu*

Anna C. Squicciarini
*College of Information Sciences & Technology*
*Pennsylvania State University*
*University Park, PA*
*acs20@psu.edu*

Pau-Chen Cheng
*IBM T.J. Watson Research Center*
*Hawthorne, NY*
*pau@us.ibm.com*

***Abstract*—Online social networking sites are experiencing tremendous user growth with hundreds of millions of active users. As a result, there is a tremendous amount of user profile data online, e.g., name, birthdate, etc. Protecting this data is a challenge. The task of access policy composition is a tedious and confusing effort for the average user having hundreds of friends. In this paper, we propose a Policy Manager (PolicyMgr) Framework for social networks. PolicyMgr assists users in composing and managing their access control policies for objects posted to their profiles. Our approach is based on a supervised learning mechanism that leverages user provided example policy settings as training sets to build classifiers that are the basis for auto-generated policies. Furthermore, we provide mechanisms to enable users to fuse policy decisions that are provided by their friends or others in the social network. These policies then regulate access to user profile objects. We implemented our framework and, through experimentation, demonstrate positive emerging results.**

## I. Introduction

Online social networking sites such as Facebook, MySpace and others are experiencing tremendous user growth with hundreds of millions of active users. The average number of friends in a Facebook network is about 130 users [4]. These social networks, along with the applications that run on them, have become a new paradigm of online interaction. As a result, there is a tremendous amount of user profile data online, e.g., name, birthdate, work history, photos and much more.

Current social network architectures adopt a simple user centric policy management approach [5], [9], [3], where a security aware user is able to specify a policy that manages access to their posted profile objects. Managing access to one's personal information for these hundreds of friends is a daunting task. Worst yet, security unaware users usually follow an open and permissive default policy. As a result, more often than not, users are unaware of who has access to what profile object. In addition, the potential for unwanted information leakage is great. We believe that new tools need to be placed in the hands of the average user to assist them in effectively managing access to their profile information.

In this paper, we propose a Policy Manager (PolicyMgr) Framework that assists users in managing access to objects posted to their profiles. Our approach leverages input from the user, metrics from their social graph, and proven supervised learning techniques to effectively identify trusted and non-trusted friends relative to a specific object, e.g., photo. The profile owner provides example policy settings as training sets to build classifiers that can precisely generate policies for other users in their friend's list - identifying them as trusted and non-trusted friends. Furthermore, we explore the fusion of policy decisions generated by their neighboring friends to enhance the accuracy of the supervised learning approach. We implemented our framework on the Last.FM social network. Through experimentation, we demonstrated the feasibility of our framework.

## II. Preliminaries

### A. Social Networks

Users and relationships between users are the core components of social networks. Each user maintains a user profile and is connected to a set of friends. We assume friendship is mutual; where if $u_i$ is a friend of $u_j$, this implies that $u_j$ is also a friend of $u_i$. Each user $u_i \in V$ maintains a profile $P_i$ which is composed of $N$ profile attributes, $A_i = \{a_1^i, \ldots, a_N^i\}$. For example, a Facebook user profile includes attributes such as birthdate, location, gender, religion, etc.

A social network can be modeled as an undirected graph $G(V, E)$ where the set of vertices $V$ is the set of users and the set of edges $E$ is the set of friendship relationships between users. The edge $(u_i, u_j) \in E$ implies that users $u_i$ and $u_j$ are friends. Using this model for social networks, we leverage the nodal network structural properties to provide additional user attributes. These attributes include several small world network metrics such as degree, betweenness, closeness, etc. [11], [2]. For a user $u_i$, we are able to compute $M$ network metrics $B_i = \{b_1^i, \ldots, b_M^i\}$.

### B. Policies in Social Networks

A user $u_i$ posting an object $O$ on their profile is allowed to setup an access control policy to specify which friends are allowed (denied) access to the posted object. The access control policy is managed and stored by the hosting social network site. Current social networks allow users to categorize friends into groups based on relationship, location, institution, family, work, etc. For example, Orkut has the following default *friend groups* defined: best friend, family, school, and work. Orkut also allows its users to create new *friend groups*. Facebook has a similar capability which they call *friend lists*. Furthermore, users are able to specify policies in terms of theses groups. Users are also able to specify exception lists indicating explicitly which friends should and should not be given access to specific objects. For example, the Family photo album in Orkut can be restricted to just the Family friend group while the Graduation photo album is viewable by everyone.

## III. PolicyMgr Framework

### A. Motivation

With the growing size and adoption of social networks, users are continuously updating their profiles by adding friends and posting new objects. For example, on Facebook alone, over 25 billion pieces of content (web links, news stories, blog posts, notes, photos, etc.) are shared each month [4]. This coupled with the fact that the average user has 130 friends makes it a challenging effort in managing access to user information. In order to guarantee fine-grained protection, a user has to specify a policy every time an object (e.g., photo) is added to their profile or they establish a new friendship. Maintaining an effective user access control policy can be a very laborious and tedious task. As a result, policies are only partially configured and maintained. Or, they may be all together ignored. This leads to user content not being properly protected and potentially unknowingly made available to unintended recipients.

In most social networks, users are allowed to specify simple policies that are based on the principle of allow and deny, where the user has to decide who to allow and deny access or who to trust or not. Instead of asking the focus user to decide for each of her friends who to give access or not, our proposed framework only requires the focus user to choose the access rights (or label) of $\alpha$ carefully selected users from her friend's list. (Note: We refer to the profile owner as the *focus user*.) Our proposed framework uses supervised learning mechanisms to decide on the access control policy settings for the remaining users. The details of our framework are further discussed below.

### B. Supervised Learning Approach

In machine learning literature, a classification learning model is a function $f$ that takes as an input a set of attributes and returns a label or classification, e.g., a function that can take the user's age, gender, credit rating and job status and generate a recommendation to either grant or deny a loan. A supervised learning mechanism uses training data $\Theta$ to learn the function $f$, which we refer to as $f_\Theta$.

Taking a simple user centric approach to address the policy composition problem would require each focus user to manually decide on trust and access control settings for each of his friends. This is a tedious task given that users have on average hundreds of friends [4], [10]. Instead, the approach we adopt is an adapted user centric approach, where the user is required to only provide a small subset of their friends' permission mappings. These example permission mappings are used as a training set $\Theta$ for the supervised learning algorithm. Basically, we attempt to learn the mapping function $f_\Theta : \mathcal{X} \rightarrow \mathcal{Y}$, where:

1) $\mathcal{X}$ is a set of user profile attributes and network metrics $\{A_j, B_j\}$ describing user $u_j$.
2) $\mathcal{Y}$ is a set of labels $\{y_0, \ldots, y_m\}$, in our case it is $\{trusted, non-trusted\}$.
3) $\Theta$ is the training set, which is a set of labeled friends' profiles, provided by the user.

Our goal is to learn the function $f_\Theta$ based on the provided training set $\Theta$. Once $f_\Theta$ is learned, we can automatically decide if a user with a given profile is allowed or denied access. This supervised learning mechanism requires an example data set to train and guide the generation of the mapping function $f_\Theta$. Given a friend $u_j$ with a profile $P_j = \{A_j, B_j\}$, the classifier $f_{\Theta_i}$ for user $u_i$ assigns the label $y_l$ to user $u_j$ provided that this label maximizes the classifier's confidence or the probability measure $P(u_j \rightarrow y_l|\Theta_i)$ based on the training set $\Theta_i$.

There are five steps involved in the learning based policy management process. In step 1, for each focus user's friends the profile attributes $A_j$ are collected and the network attributes $B_j$ are computed based on the generated social graph information. In step 2, the collected attributes $\{A_j, B_j\}$ are used to cluster the focus user's friends into $K$ non-overlapping clusters. The clustering is performed to ensure that the focus user labels representative members of each of the computed clusters. We use the k-means clustering algorithm [8]. From each of the clusters, we randomly select $\alpha$ friends. In step 3, The focus user is then asked to classify each of the $\alpha$ friends, that is to indicate which of the friends are trusted to access a specific object and which are not trusted. This implies the classification labels are selected from the set $\mathcal{Y} = \{y_0, y_1\}$, where $y_0$ and $y_1$ are trusted and non-trusted classes respectively. The labeled $\alpha$ friends are added to the training set $\Theta$. In step 4, the training set $\Theta$ can be used directly to train a classifier. However, there are several classifier algorithms and it is crucial to select the classifier that is most suited for this specific user instance. So, the mechanism we adopt is to train and tune several classifiers and then compare their performance based on standard cross validation methods such as n-fold cross validation [15]. Given $m$ classifiers $\{f^1_{\Theta_i}, \ldots, f^m_{\Theta_i}\}$, the classifier with the highest accuracy is selected, which is denoted as $f^*_{\Theta_i}$.

In step 5, the knowledge accumulated by other users in the social network can be utilized to further enhance the classifier accuracy. It is important in this step to seek classification advice from other friends who classify users similar to the focus user. This is referred to as the *selection process* where $\beta$ other user classifiers are selected based on their accuracy in labeling the focus user's training set $\Theta_i$. In the *fusion process*, the decisions of the selected $\beta$ classifiers are combined with the decisions of the focus user's classifier to classify the remaining focus user's friends. The details of this approach are discussed in the following section.

### C. Classifier Selection and Fusion

The inherent advantage of social networks is the ease of sharing of news, photos, videos and several other data objects among users. We extend this sharing to include the accommodation of user experiences by leveraging their trained classifiers, where user $u_j$ is able to share their mapping function $f_{\Theta_j}$ with other users. Assume a user $u_i$ would like to leverage the experience of other users in the social network to improve their mapping function $f_{\Theta_i}$. We use $f_{\Theta_k}$ to refer to the best classifier $f^*_{\Theta_k}$ for user $u_k$.

Given a user $u_i$ and a set of users $S = \{u_1, \ldots, u_n\}$, the set $S$ can be chosen from the neighboring trusted friends or from other experienced users in the social network. Each user $u_k$ in the set $S$ is willing to share their mapping function $f_{\Theta_k}$ to improve the mapping function of user $u_i$. This translates into two sub-steps: (1) The selection of $\beta$ users from the set $S$ that are best fit to help user $u_i$ in computing an improved mapping function, (2) The fusion of the different $f_{\Theta_k}$ functions provided by the $\beta$ users with the focus user's function $f_{\Theta_i}$.

*Definition 1:* (Selection) Given a user $u_i$, a set of user trained classifier functions $f_S = \{f_{\Theta_1}, \ldots, f_{\Theta_n}\}$, the training set $\Theta_i$ for user $u_i$, and a classifier fitness function $\Phi : f_{\Theta_k} \times \Theta_i \rightarrow \Re$, select the best $\beta$ classifiers based on the fitness function.

The selection process is based on the fitness function as defined in Def. 1. The fitness function is a mechanism to rank the classifiers in $f_S$ based on their similarity to the decisions taken by the classifier of user $u_i$. This mechanism attempts to locate the $\beta$ classifiers that match the focus user's perspective. The fitness function tests each classifier $f_{\Theta_k}$ by labeling the tuples in the training set $\Theta_i$ and computing the vector $[TP, TN, FP, FN]^T$, which represents the true positive, true negative, false positive and false negative respectively. The fitness of $f_{\Theta_k}$ is based on the classifier accuracy [12], [1], computed as $\frac{TP+TN}{TP+TN+FP+TN}$. The $\beta$ classifiers with the highest fitness are selected and are denoted by the set $S_\beta = \{f_{\Theta_1}, \ldots, f_{\Theta_\beta}\}$.

Given the $\beta$ classifiers, the next step involves fusing the decisions of these classifiers and the decisions generated by the focus user's classifier ($f_{\Theta_i}$) to improve the classification result. We adopt the following classifier fusion algorithms [6]: *group voting*, *group confidence product* and *most confident*.

After the $\beta$ classifiers with the highest fitness are selected, an appropriate fusion algorithm (of the three listed above) is chosen to fuse the results of the $f_{\Theta_i}$ and the $f_{\Theta_k}$ functions producing a predicted label, i.e., trusted or non-trusted. This final classifier function is designated as $f^\beta_\Theta$. Note that when the focus user adds new friends, the generation of the their access rights can be automated based on the available function $f^\beta_\Theta$. Furthermore, as the focus user adds similar objects to their profile, the new objects can also adopt the same classifier function.

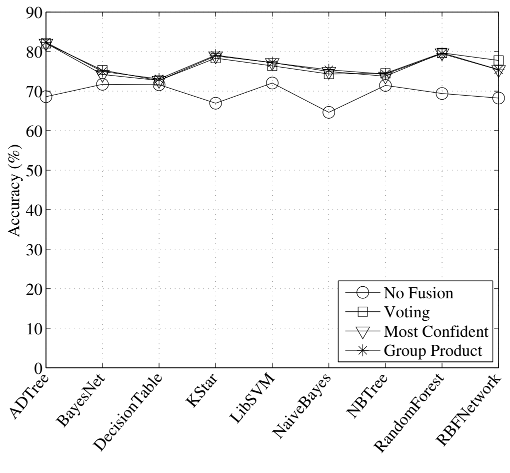
## IV. Experimental Results

Last.FM [7] is one of the web's most popular recommendation and web radio services providing a social networking platform where friendships have an impact on what people listen to. Using Last.FM's public developer API, we implemented a crawler that was able to collect user profile attributes and friendship relationships. The crawler was written in Java 1.6; the crawler loads the focus user's profile information, their friends list, and stores all the loaded data into a MySQL database. We collected about 1.6 million user profiles and about 13 million friendship links.

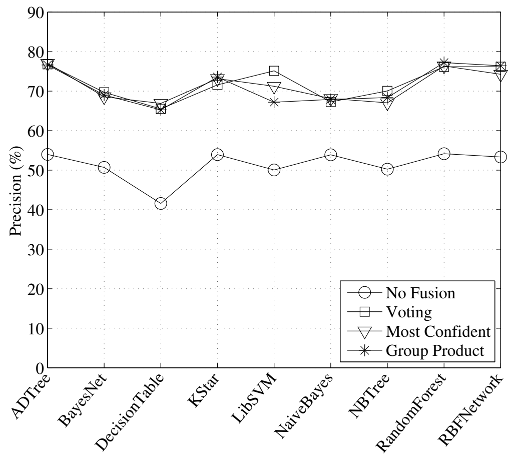
A series of experiments were conducted using a randomly selected subset of the Last.FM data set, approximately 200 focus users. For each focus user, the following profile attributes were obtained: Age, Gender, and Home Country. We also collected Shouts, which are posts made by users on their friends' profiles. In addition, each focus user's social graph was built and a series of network metrics were computed on their respective social graph, which included degree centrality, betweenness centrality, closeness centrality, common friends, HITS, and Eigenvector centrality.

The network metrics for different users were computed using the open source Java Universal Network Framework (JUNG) [13]. Following the collection of user data, a training set $\Theta$ was compiled out of a subset of the focus user's friend set. The overall friend set was clustered into $K$ clusters leveraging the k-means clustering algorithm. (Note: We fixed $K$ to two; further experiments are necessary for other values of $K$.) The training set $\Theta$ was comprised of a percentage $\alpha$ of friends randomly selected from each cluster. Each user in the training set $\Theta$ was labeled as trusted or non-trusted. The trusted and non-trusted label for each friend in the training set would normally be applied by the focus user, instead the number of Shouts generated was used as an indication of trust ($Shouts > 5$). Further user studies are planned to capture actual trust labels from the user.

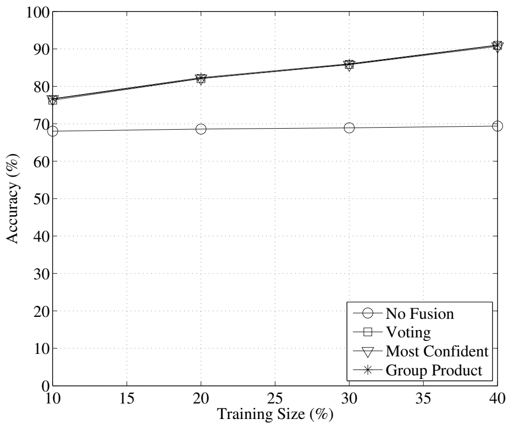
The training set $\Theta$ includes, for each friend, a trusted and non-trusted label. The remaining friend set, called the test set, and the training set are inputs to a variety of different classifiers. In our experiments, nine different classifiers were trained which included the following: Naive Bayes,



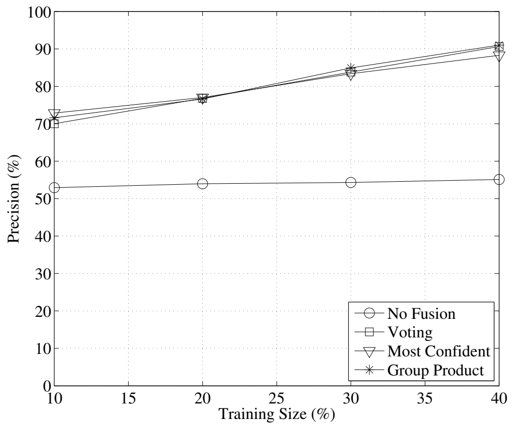
(a) Classifier Type vs. Accuracy



(b) Classifier Type vs. Precision



(c) Training Set vs. Accuracy



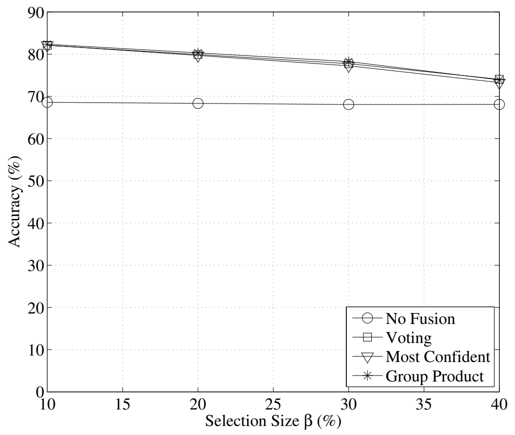
(d) Training Set vs. Precision

Figure 1. Experimental results

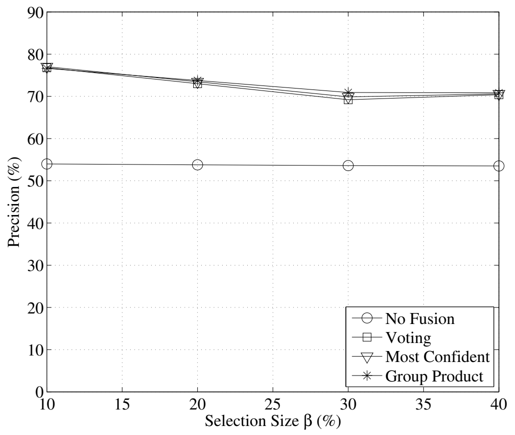
BayesNet, Radial Basis Function Network, K Star, AD Tree, Support Vector Machine, Naive-Bayes Tree, Random Forest and Decision Table. The classifiers were generated using the open source Java WEKA 3.6 library [14]. The classifiers were tested by labeling each friend in the test set as either trusted or non-trusted based on the users' profile attributes and network metrics. The true positive, true negative, false positive, and false negatives for each classifier were recorded. The classifier with the highest accuracy is selected, which is denoted as $f^*_{\Theta_i}$.

Following the section of $f^*_{\Theta_i}$, additional classification advice is sought from the focus user's friends. For each focus user, $\beta$ friends (between 10-40) were selected from the focus user's friend set who classify users similarly as the focus user. These $\beta$ additional classifiers are fused with the focus users classifiers to label the remaining friends as either trusted or non-trusted. A future enhancement will include an additional step giving the focus user an opportunity to review, and possibly re-label, each user and thus improve the overall classification results.

Figures 1(a) and 1(b) show the accuracy and precision results generated for the different classifiers using a training set $(\alpha)$ equal to 20% and 10 selected friends $(\beta)$. Accuracy (or correctness) was previously defined in Section III-C and precision (or reproducibility) is defined as $\frac{TP}{TP+FP}$. From these results, we are able to show that without any fusion, the classifier is capable of providing up to $70\%$ accuracy and $55\%$ precision. Using the different fusion mechanisms, the classifier accuracy improved to $83\%$ and the precision increased to $78\%$. Based on these results, it is evident that our fusion based approach improves the classification result with the voting based approach leading the other fusion mechanisms. Furthermore, the AD Tree classifier provides the highest accuracy and precision results. The classifier results could be further improved if the user profile attributes collected were complete, as several user profile attributes were not available. Also, as previously mentioned, a future enhancement will include the presentation of the recommended label to the focus user for validation which would thus improve the overall accuracy. Our approach is a vast improvement over the current state of the norm of users completely ignoring their policy altogether or going through



(a) Selected $\beta$ vs. Accuracy



(b) Selected $\beta$ vs. Precision

Figure 2. Experimental results varying the selected $\beta$

the hassle of labeling all users by hand.

Figures 1(c) and 1(d) depict the accuracy and precision of the fused classifiers (group voting, group confidence product, and most confident) and the best classifier of the focus user (no fusion) holding all parameters constant (AD Tree classifier, $\beta = 10$) except for the size of the training set $(\alpha)$. The training set was varied from 10%-40%. The fused classifiers, performed consistently better than the no fusion case with the accuracy and precision improvements proportionally increasing with higher training set sizes. The fused classifiers, for the most part, performed similarly. There is an improvement going from 10% to 20%, i.e., if a user labels 20%, vice 10%, of his friends as trusted or non-trusted, there is improvement in the accuracy of the classifier labeling the test set. Therefore, it is sufficient to ask the user to label between 10% to 20% of their friends in order for the policy manager to effectively label the remaining users as trusted or non-trusted.

To investigate the size of selected fusion classifiers $(\beta)$, we conducted experiments holding all parameters constant (AD Tree classifier, $\alpha = 20\%$) while varying $\beta$. Figures 2(a) and 2(b) depict the accuracy and precision of the fused classifiers and the best classifier of the focus user (no fusion) for the different $\beta$ values (10-40). Note that as we increase $\beta$, the accuracy and precision drop. This is because as the size of $\beta$ increases, the fusion result is diluted with more users who have classifiers with low confidence. Note that even though the accuracy and precision drop as $\beta$ increases, the fusion based classifier still consistently performs better than the no fusion classifier. If most user mapping functions $f_{\Theta_k}$ have low accuracy rates, a threshold for $\beta$ can be introduced.

## V. Conclusions

In this paper, we presented a Policy Manager Framework that assists users in composing and managing their access control policies in social networks. We proposed an adapted user centric approach, based on supervised learning mechanisms, to decide on trust and access control settings for each user's friends. We incorporated knowledge from others in the social network to enhance the supervised learning results. Moreover, we demonstrated the feasibility of our framework by implementing it on a social networking site.

## References

[1] L. Barbosa and J. Freire. Combining classifiers to identify online databases. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 431–440, New York, NY, USA, 2007. ACM.

[2] S. P. Borgatti and M. G. Everett. A graph-theoretic perspective on centrality. *Social Networks*, 28(4):466–484, October 2006.

[3] C. F. Enterprise, A. Jsang, and S. Pope. Auscert conference 2005. In *in Asia Pacific Information Technology Security Conference, AusCERT2005, Austrailia*, pages 77–89, 2005.

[4] Facebook. Facebook statistics. *http://www.facebook.com/press/info.php*, April 2009.

[5] A. Jsang, M. A. Zomai, and S. Suriadi. Usability and privacy in identity management architectures. In *ACSW '07: Proceedings of the fifth Australasian symposium on ACSW frontiers*, pages 143–152, Darlinghurst, Australia, Australia, 2007. Australian Computer Society, Inc.

[6] J. Kittler, M. Hatef, R. P. Duin, and J. Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–239, 1998.

[7] Last.fm. Last fm platform. *http://www.last.fm*, 2009.

[8] B. Liu. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data (Data-Centric Systems and Applications)*. Springer, first edition, 2007.

[9] T. E. Maliki and J.-M. Seigneur. A survey of user-centric identity management technologies. In *SECUREWARE '07: Proceedings of the The International Conference on Emerging Security Information, Systems, and Technologies*, pages 12–17, Washington, DC, USA, 2007. IEEE Computer Society.

[10] Neil Swidey. Friends in a facebook world. *Globe Newspaper Company*, Nov. 2008.

[11] M. E. J. Newman. Scientific collaboration networks. II. Shortest paths, weighted networks, and centrality. *Physical Review E*, 64(1):016132+, June 2001.

[12] J. Perols, K. Chari, and M. Agrawal. Information market-based decision fusion. *Manage. Sci.*, 55(5):827–842, 2009.

[13] The JUNG Framework Development Team. Java Universal Network/Graph Framework. *http://jung.sourceforge.net/*, 2009.

[14] The University of Waikato. WEKA Machine Learning Project. *http://www.cs.waikato.ac.nz/ ml/index.html*, 2009.

[15] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, second edition, 2005.