Feature Factory: A Crowd Sourced Approach to Variable Discovery From Linked Data

Kiarash Adl

Advisor: Kalyan Veeramachaneni,
Any Scale Learning for All
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology

Contents

List of Figures

1 Introduction

Big data analysis is one of the most pertinent scientific endeavors. In addition to storage, retrieval, analytics and visualizations, predictive modeling is a primary focus to derive value from data. There have been many machine learning techniques developed for different problems. For example, predictive modeling has been used extensively in analytical customer relationship management when its important to make a model to predict why consumers make certain actions. Another example which is more in focus in this paper is why online students make certain decisions while taking classes on Massive Open Online Course (MOOC) platforms.

One of the key requirements of building a predictive model is to define variables from which these models would be built. For example when seeking why student stopout from a MOOC platform, one could suggest to look at the number of hours each student spends on the platform or another one could be the number of homework problems each student got right. As we can see, defining variables usually requires human intuition and ingenuity.

For some types of data, for example images and signals, machine learning systems have tried to eliminate the need for human involvement, variable definition or data representation. For instance, as part of the Google brain project, Google used a deep learning algorithm and fed it with many images to identify common objects such as cat images. In this case they didnt need to define what a cat is to the algorithm but the deep learning method takes care of it.

However, for some data sources, such as the ones that come from monitoring human activity or behavior, one can not use these algorithms and there is still a need for human intuition to define variables from this type of data for building predictive models. For example predicting churn from online store-online activity data for the customers is recorded and variables need to be formed to predict churn. Another example would be predicting if a transaction is fraud or not based on previously stored details about millions of transactions.

One of the most challenging decisions to make when doing machine learning research on human activities data is what variables to choose. There are already many powerful machine learning tools but representation of data as features has to be done case by case, often solely based on the researchers experience. However, one researchers opinion might not cover all the possibilities of features to analyze. In real applications we often see that the differentiator that leads to the success of a platform is the features that they chose. This suggests that we need to find a better solution to feature discovery for such problems.

Solution to this problem is crowdsourcing the feature discovery. One of the newly known platforms for this problem is CrowdAnalytix. CrowdAnalytix breaks up projects in two phases: feature engineering and modeling. During the feature engineering phase, data scientists are presented with a problem (independent variable(s)) and are asked to propose features (predictors) and brief explanations for why they might prove useful. A panel of judges evaluate features based on the accompanying evidence and explanations. The modeling phase is a traditional machine-learning competition (entries compete on standard quantitative metrics), using data sets that incorporate features culled from the earlier phase.

Existence of CrowdAnalytix is another evidence that there is a need in this area. However, one central system cant solve the problem and there needs to be a version of such platform where the data cant be transferred for security reasons. In our case, we wont be able to give database access to a third-party and thus a native platform could work much better. We have mock data on our online platform where users can submit and test their scripts but the real database is securely inaccessible by public.

With a goal to develop such general platform we first focused on one problem. As mentioned earlier of the cases that we need human intuition for feature definition is when we are working with student activity data. Massive Open Online Courses (MOOCs) are widely influencing the education system and unlike

the traditional classes, they offer the opportunity to collect wide range of data about them. This data includes students interaction with the course such as their observations, submissions (i.e. assessments) and collaborations. Machine learning has been one of the areas of research that has potential to allow researchers benefit from this data to improve educational and learning outcomes. Researchers are applying different machine learning techniques on the students interaction data to understand different aspects of the online courses to improve learning outcomes.

For example, one of the problems that we are working on is to predict student stop out on MOOCs. There could be many reasons that lead to students stopping to be active on an online course and there are many possible features to extract to be able to predict stopout. One might suggest that the total number of weekly visits to the website for each student is a good indicator. Another suggestion might be to extract total time spent on the site by each student. The vast range of possibilities makes this crowd-sourcing platform a significant help to the researchers to get the right features.

In this work we developed a platform to enable crowd-source feature discovery and piloting the frame work for a MOOC data problem. One researchers opinion on what to form as variable could be great but the instructors or the students working on an online course might have opinions on what features are relevant to an event as well. In addition, other experienced researchers who cannot access to the data (i.e. privacy issues) might be able to help with the feature discovery. For these reasons, a platform that allows people collaborate on the feature discovery can provide improvements to the final result.

2 ALFA and Previous Attempts to Crowd Sourcing Feature Discovery

ALFA group at MIT started the preparation for crowdsourcing feature discovery for MOOC data in fall, 2013. One of the first crucial steps was to create a standardized linked data model to hold the MOOC data. Such shared data model would allow researchers to collaborate and investigate on a dataset only based on the data schema and without a need to have access to the original data. This data schema called MOOCdb also allows researchers to be able to share feature extraction scripts independent from the data. For example, researchers at Stanford with access to Coursera data can use the same scripts as of MIT researchers using MITx MOOC.

In the first attempt to crowd source feature discovery, ALFA group conducted a survey from students and professors participating in one of the courses about online learning platform, 6.MITx. They made a presentation to the class about the data, data model and what they are seeking from the data. They asked to fill out a google form with their ideas of possible features to extract that would predict stopout. Many students and professors participated in the survey and many features were suggested. Over time, researchers in the ALFA group extracted some of those features and experimented with them. However, the need to repeat the experiment with a bigger crowd became more obvious.

Another step was to make a mock database and make it available online. The original data cant be accessed by the public for privacy reasons. However, even in the existence of the MOOCdb schema, there needed to be a database with data to test scripts. ALFA created two databases of small and large size, very similar to the original dataset, but with artificial data for testing purposes. The schema is exactly the same as the standardized MOOCdb schema which makes it possible to test scripts easier without having access to any real data.

All these previous efforts made it possible to realize this web platform. With this platform any researcher or instructor can contribute in the feature discovery and extraction.

3 Feature Factory

In this work we developed a platform for crowdsourcing feature discovery which is called Feature Factory. Feature factory allows participants to get involved in the process of feature discovery with three types of contributions.

1. They could propose feature ideas
2. They could comment or up-vote on an existing idea to help refine it
3. Finally, they could write SQL code to extract features for one of their own ideas or for an already existing idea.

As illustrated in figure 1, the Feature Factory website has 4 sections. Section 1 describes the general nature of the project and explains the methodology. In section 2, we describe the current prediction problem that we are seeking ideas for variables. As mentioned earlier, this tool can be used to solve problems wherever there human activities data but at each time we solve one specific problem. Section 3, is a button that will allow the user to add a feature idea to our problem. Finally, all the ideas and scripts submitted to the platform will be listed on the website as shown in section 4.

3.1 Enter an idea

The most important part of feature discovery process is coming up with new ideas. Multiple volunteers from different backgrounds can be helpful in producing the right feature ideas. Volunteers might have teaching experience such as instructors of a MOOC platform, or they could be students in the system. They could also be researchers with experience in feature extraction and machine learning. In feature factory platform all these volunteers are allowed to submit their ideas and publish it online.The idea submission is made very easy as shown in figure 2. Furthermore, these volunteers can help with filtering the more relevant ideas out of many submissions. Feature factory website has employed a commenting and voting mechanism which allows volunteers to help refine the best of ideas which would be available for extraction. Figure 3 shows the comment box for an idea.

3.2 Enter a code and debug

One of the main goals of feature factory was to allow crowdsourcing of feature discovery process as much as possible. As a result the platform allows participants to write and debug SQL code to extract features. Figure 4 shows the code submission form for an idea. For a participant, they have the chance to review all the existing ideas in the platform and they can browse any one with their extraction code. Section [1] of the figure 4 shows the existing code for an idea. They could also write code and submit for evaluation shown in part [2] of figure 4.

The code evaluation happens online within seconds of submission. The submitted code will be run on two mock datasets to test syntax as well as running time. The user will then receive the status of the submission via an email. Figures 5 and 6 show two type of responses that a volunteer might receive.

All the validated code and ideas are compiled and displayed on the website. This would allow the researchers to have access to the all the validated feature code which they could easily run on other MOOCdb databases. This also allows participants to learn from the existing code on the system and also to make sure overlapping work is minimized. The ultimate goal of the platform is to produce a rich database of ideas and their extraction code which can help many researchers working on similar problems.

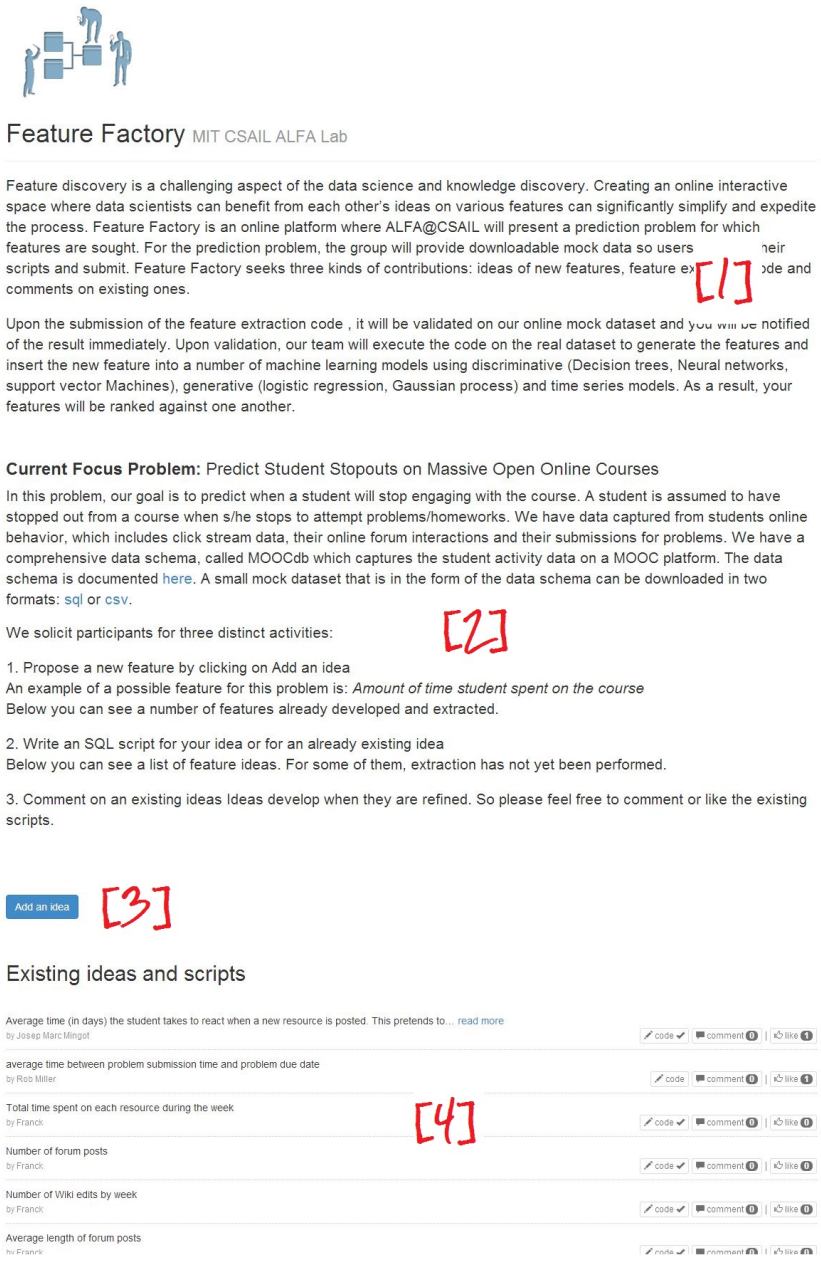

Figure 1: A complete view of the website

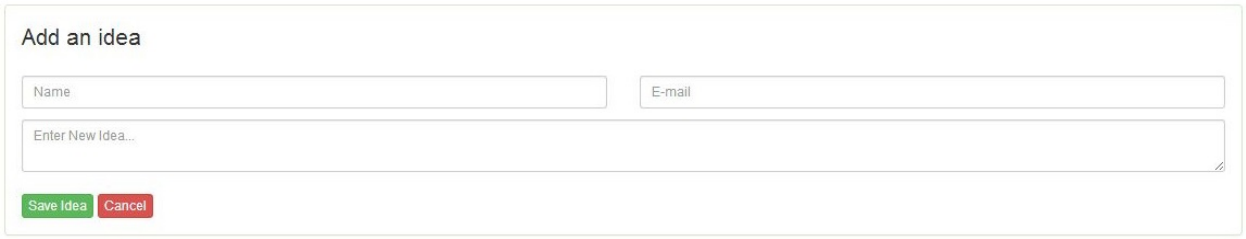

Figure 2: Form to submit new ideas

Figure 3: Volunteers are allowed to comment on all published ideas.

4 Feature Factory Technical Underpinnings

The platform is build using Django framework with MySQL databases to host the mock MOOCdb as well as the for internal server needs. There are four modules that will be described in here. There is the core of server that keeps track of submissions including code or ideas. There is also a real-time task queue which allows the submitted code to run on the mock data in the background and with some order to avoid overloading the server. Finally, there is the voting system.

4.1 Database to organize and store ideas and comments

The core object in the django model is an Idea object. Volunteers can submit ideas and after that, code or comments can be submitted for each idea. In the database schema, we have an idea object which contains name and email of the person submitting the idea as well as the idea description. The volunteer will submit these three values through a form on the website. Code and comment are the two other objects relating to an idea in our django database schema. When submitting a comment, the user only needs to choose an idea and write the text in a form box. However, we also record their ip address and submission time automatically to potentially stop abuse of the system.

Each submitted code will also form a row in the django internal database which related the code to an idea. Further we require the volunteers to submit their name and email address for us to be able to send them the result of the submission. They also need to choose whether this code is their final code or if they

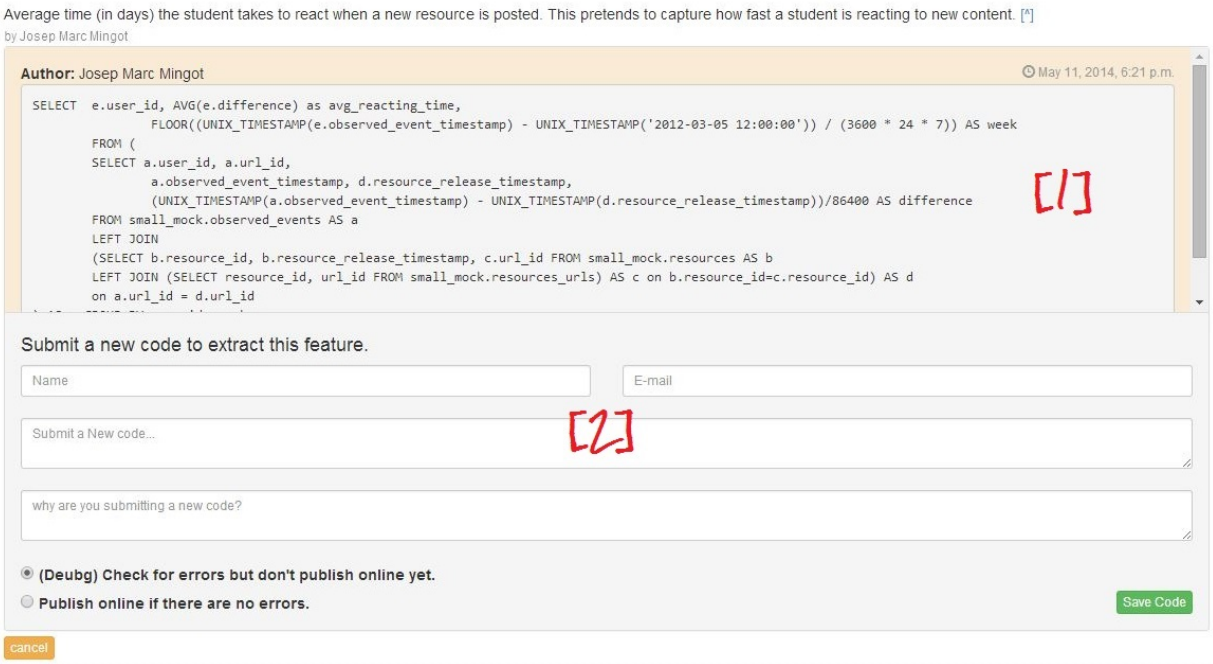

Figure 4: One can submit code for an idea or review the published ones.

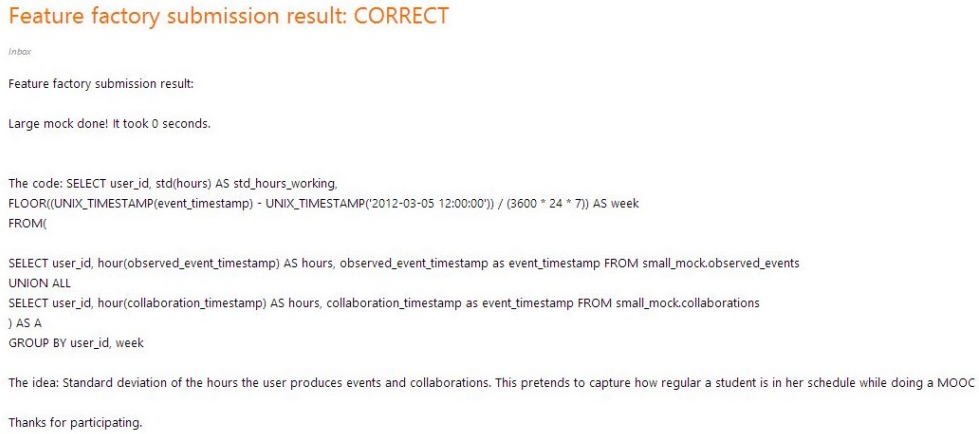

Figure 5: Code evaluation. Correct codes will receive the running timing.

are simply using our platform to debug their code. In the latter case, we wont publish the code online even if it shows no syntax errors. There are multiple private fields in the code object which gets filled when we run

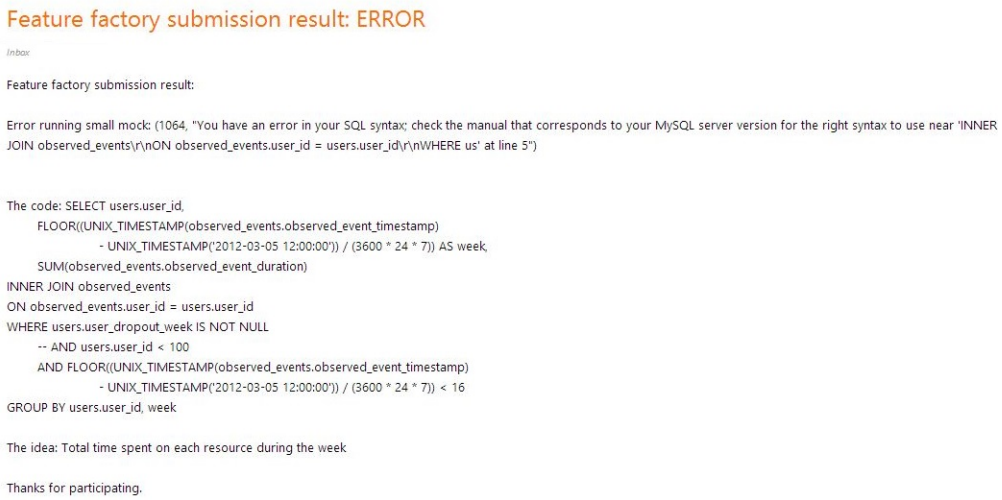

Figure 6: Code evaluation. Invalid codes will receive the error message.

the code in the background by the automatic distributed task queue. For example, we record how long the code took to run or whether it worked without errors. Full description of these three objects are illustrated in figure 7.

4.2 An online code evaluation system

When a code is submitted, we would like to run it on the mock dataset as soon as possible and inform the volunteer of the result. However, this needs to be done in the background because it could take several minutes for the code to run. It is also possible that several people submit code at the same time and we would need to run them in an order. To solve these problems, we decided to use Celery which is a well-known production level distributed task queue. Celery uses Django database in the backend to keep track of tasks and it can have multiple worker threads in the background running the submissions in a distributed fashion. At the end of each process, it will email the users with the result and it will update our internal database. In addition, in case of any failure in running the code, the server will stay online but there will be delay in processing the submissions. Overall, Celery will allow feature factory platform to run code in the background seamlessly.

4.3 Management of voting

A voting system is obtained to avoid abusing the vote mechanism. For each idea, we keep track of the number of up-votes given by viewers and we sort the ideas based on the number of votes. Without user logins it is impossible to be certain that a user does not vote more than once, but there are several methods to limit abuses. We decided to use an existing open source code to extend our voting mechanism to make it secure. We use Django-secretballot developed by Sunlight Labs. Secretballot allows to solve the voting problem by keeping track of a token generated from each request. For our case, to limit the votes, we set

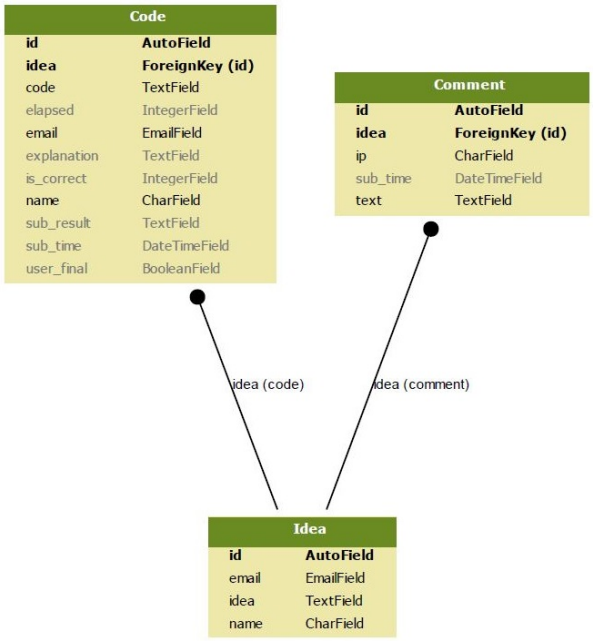

Figure 7: Feature Factory internal data schema.

the token to the users IP address. This way a user accessing the website from a particular IP address can only vote once on each idea.

4.4 Server hosting

Currently, we have decided to host the server on our internal Openstack framework while hosting all our databases on separate central CSAIL MySql server. An Apache server is utilized to host the Django platform on an Openstack machine. Furthermore, Celery is installed on the machine to process the tasks in a distributed fashion in the background. This design seems to be stable and capable to rebooting without harm if needed. The database is regularly backed up by the CSAIL MySql system and as a result rebooting the server will be almost with no harm. In addition, Celery is capable of restarting the tasks if interrupted.

5 Existing Submissions On Feature Factory

In this section we summarize some of the submissions currently on the platform. Many of these submissions werent originally submitted through feature factory but through surveys. But we decided to initialize the platform with this data. At this point, because of very recent launch of the platform, we only have few original submissions but we are looking forward to see the result after advertising the platform widely.

The problem on the platform at this time is to predict when a student will stop engaging with the course. The platform allows people to experiment with two mock datasets of MOOCdb format and some of the

submission are as below:

one of the submission is Average time (in days) the student takes to react when a new resource is posted. This pretends to capture how fast a student is reacting to new content. by Josep Marc Mingot. He also submitted a valid code for this feature and as an example of such submissions, the code is shown in figure 8.

```
SELECT e.user_id,
       AVG(e.difference) AS avg_reacting_time,
       FLOOR((UNIX_TIMESTAMP(e.observed_event_timestamp) - UNIX_TIMESTAMP('2012-
03-05 12:00:00')) / (3600 * 24 * 7)) AS week
FROM
  ( SELECT a.user_id,
           a.url_id,
           a.observed_event_timestamp,
           d.resource_release_timestamp,
           (UNIX_TIMESTAMP(a.observed_event_timestamp) -
UNIX_TIMESTAMP(d.resource_release_timestamp))/86400 AS difference
   FROM small_mock.observed_events AS a
   LEFT JOIN
     (SELECT b.resource_id,
             b.resource_release_timestamp,
             c.url_id
      FROM small_mock.resources AS b
      LEFT JOIN
        (SELECT resource_id,
                url_id
         FROM small_mock.resources_urls) AS c ON b.resource_id=c.resource_id) AS d
ON a.url_id = d.url_id) AS e
GROUP BY user_id,
         week;
```

Figure 8: Sample feature extraction script on Feature Factory. Average time (in days) the student takes to react when a new resource is posted.

Professor Rob miller has some feature suggestions on the platform. average time between problem submission time and problem due date and fraction of observed resource time spent on each day of the week (7 variables for Mon-Sun that add up to 1.0). Just for previous week, and averaged over all weeks so far are two of them.

Some ideas are explained in more detail while some of them are brief. For example an idea submitted by Chris Terman is Total number of weekly interactions for each student = # of observations + # of submission + # of collaborations Feature is if current week total number of interactions is significantly less (2 sigma) than averages of previous weeks. Theory: if students fall behind for some reason, they are likely to give up. and another idea by Franck is Duration of longest observed event.

6 Discussion and Conclusion

There are many requirements for feature factory to become a successful system of feature discovery. This includes a powerful publicity to get volunteers get involved with the platform. But there are also ways to improve the platform to make it easier or more fun for the volunteers which makes the overall performance

better. Some of these improvements are already done but some of them are to be done as extensions to the existing platform.

As mentioned before, idea submission is an important part of feature factory. Producing high quality and filtered ideas is the goal of the system. At this point, we have been able to provide the support for submission, comments and votes in our platform. We have also entered a set of ideas acquired in the past as a starting point of the platform and we have had few volunteers enter new ideas. From this point on, we need to advertise the platform to help getting more volunteers propose ideas or to just comment or vote on them to help filter the best of them. We are planning on using MIT news as our next venue to advertise the platform. It would be helpful to continue doing presentations at the end of MOOC courses offered at MIT to introduce the idea to the students as well as instructors of the course.

The second part of the platform that needs help of publicity is the code extraction. Unlike the idea submission which was possible to participate with minimum knowledge of coding, here we need people with programming skills. However, the number of people needed for this part is less because one person can code many ideas in rather short of amount of time. For this part we would need to motivate the programmers with prizes or karma. This could also become a way for beginners to advance their programming skills.

An important way to motivate programmers as well as idea producers to participate more actively in the platform is to show them the result of their work. People would like to know how their participation influenced the research but they also wouldnt like to wait for months to see the result. Therefore it is motivating if we could produce some exciting results based on submissions. The existing naive solution to this has been that we validate the code and publish them online if they are good. This way participants will see a preliminary action taken place on their submission. However, a more complete solution is needed in the future.

One of the suggestions for future improvement is to bring a real dataset into the platform and complete the loop. For security purposes at this point, we are not hosting any real datasets on the internet. The submitted code are tested on mock dataset and the scripts are available to researchers to try on real data. It is possible in the future to host the real data online while keeping it safe. This brings up the opportunity to run the submitted code on the real data and extract real features. Its further possible to compare the features to each other and rank them. We could be able to rank their uniqueness or we could also run machine learning algorithms on the features and showcase the result. Any of such actions would motivate the participants to be more active on this competitive environment.

Finally, it is helpful to have a clear guidance for participants. Currently, we are providing participants a clear problem. We do also provide complete information on the the database schema and we provide a small mock database to download. We are planning on improving this aspect of the work even further. A video tutorial as well as a step by step guidance on how to work on the system can help the volunteers to easily adapt this crowd sourcing approach.