The Development of the Back-End of Learn2Mine, a Platform for Teaching Data Science

An essay submitted in partial fulfillment of

the requirements for graduation from the

Honors College at the College of Charleston

with a Bachelor of Science in

Data Science (Molecular Biology Cognate) and Computer Science

Clayton Turner

May 2014

Advisor: Dr. Paul Anderson

Abstract

Data science is a new field in which students and scientists alike are training in order to harness the power founded in the specialization. Data science is the culmination of three disciplines: computer science, mathematics, and a domain. Data scientists, typically, come from a variety of fields because of its interdisciplinary nature; data scientists can come from business, biology, economics, mathematics, and many other fields. The problem that is arising from these data scientists coming from a multitude of different fields is that no data scientists have the same training. There may be data scientists who understand the algorithms which can be utilized on big data, while other data scientists may be better geared to interpret domain-specific results from the output of a classifier. The solution to this problem is to standardize the teaching for data scientists. This standardization has finally been created: Learn2Mine.

Learn2Mine is a mature project that has been under development for multiple years. Learn2Mine is a system where users can navigate to the site and learn basic and advanced programming skills and learn basic and complex data science techniques all while using domain-specific datasets so users are able to learn about all the different facets of data science. This allows the proper training of data scientists by exposing them to every skillset a data scientist can possess.

Over the years, Learn2Mine has undergone many iterations of its system and has become a system which students and scientists alike have been able utilize in order to learn data science. The back-end of Learn2Mine has reflected this change through every iteration. Initially, the back-end was open to the users of Learn2Mine and it became apparent that the system had to evolve. This evolution resulted in the masking of the back-end and the usage of RESTful web calls in order to create a more efficient, easier-to-use system. The back-end implementation of Learn2Mine has been evaluated by users of the system through the recording of their results and modifying the system accordingly.

I. Introduction

Throughout the ever-present technological revolution currently in effect, the development of cross-disciplinary fields, such as data science, has completely metamorphosed the nature of academia. Developments have been made in order to train people, regardless of age, to take advantage of and acquire the knowledge attainable through the meshing of academic fields. When considering data science, it is, unfortunately, still often the case that collaborations between computer scientists and domain experts, those whose primary discipline is grounded in science or business, are limited by the stark differences in the fundamental nature of their skill sets. Attempting to minimize and eliminate this gap, computer scientists have developed domain-specific software and algorithms, but the use of these algorithms typically requires significant computing expertise. Bridging this gap has been made easier, yet not trivial, through the usage and implementation of modern programming languages, such as python or java. Languages like these have allowed developers to minimize platform dependencies, and attempts have been made to connect the domain expert with informatics systems (e.g., Galaxy (Blankenberg et al., 2001; Giardine et al., 2005; Goecks et al., 2010), Weka (Hall et al., 2009), RapidMiner (Mierswa et al., 2006), and Taverna (Wolstencroft et al.)); however, these applications often require additional dependencies for specific domains, lack an intuitive mechanism for feedback and training, and requires both extensive computer science expertise and dedicated computing resources for large datasets.

The amount of data present in today's society has become enormous – so much so that there is an increased demand for people whom possess skills necessary to analyze these datasets that are consistently growing at an increased rate (i.e., big data). While undergraduate and graduate programs have begun to emerge in the field of data science (Anderson et al., 2014; Argamon; Cheshire; Dubois), the number of open positions for computer scientists and applied mathematicians with the training and curiosity to make discoveries in the world of big data has exponentially increased over the last few years. At present, the community lacks engaging data science and analytics software aimed at teaching aspiring students to explore and find patterns in large datasets.

Learn2Mine is a learning environment which seeks to introduce students to data science – computer science, statistics, and domain expertise. This is done through succeeding where other environments have failed - by reducing the need for local computing resources, removing programming as a prerequisite, and engaging the students via gameful experiences – increasing students' accessibility to proper training in data science and analytics. Learn2Mine is a cloud-based application intended for use by academics, industry professionals, and students interested in the field of data science and its related sub-disciplines (data mining, statistics, etc.). To further engage students, gameful experiences are built into the Learn2Mine environment; lessons with rewards, in the form of badges and a progressively-built skill tree that includes many common algorithms and key programming paradigms. Algorithms introduced include k-Nearest Neighbors, Market-Basket Analysis, Neural Networks, and Naïve Bayes, among others, while programming paradigms include basics such as File I/O and defining of Functions, and even more advanced techniques such as the implementing of classification prediction through the use of Partial Least-Squares Regression. Learn2Mine provides easy access to tools for these techniques that can be used for user-driven projects or research.

Learn2Mine is a project that has been under development for 3 years and, thusly, has induced the need for multiple developers to work on the project. My work has been focused upon the implementation of the back-end of the system, while the front-end of Learn2Mine was created by Jacob Dierksheide, a fellow undergraduate student at the College of Charleston.

II. Motivation and Related Work

A. Rise of Data Science

Data science, a conglomeration of computer science, statistics, applied mathematics, and a domain expertise, aims to glean new knowledge from existing information. Data science works best whenever applied to a very specific domain. When applied to biology, for example, the specialization is often referred to as bioinformatics. Bioinformaticians would work with datasets that only make sense in the field of biology – so they could be working with data from gene sequencing. A task common to bioinformaticians with genetic data is the alignment of genes from two different species in order to examine the relative relatedness between the species, observe/analyze the effects of convergent and divergent evolution, and any other task they so desired. The algorithms used by these bioinformaticians are specialized versions of algorithms that can be used in other domains. For example, the Smith-Waterman and Needleman-Wunsch algorithms commonly utilized by bioinformaticians are merely variations on a dynamic programming algorithm which is typically used for extracting information from graphs/networks and common string algorithms, like Levenshtein distance. Biological expertise is required to interpret and modify the algorithms that are initially taught and utilized in computer science, so knowledge from both bases is vital. An easier to comprehend version of what it means to be a data scientist can be sign in figure 1.

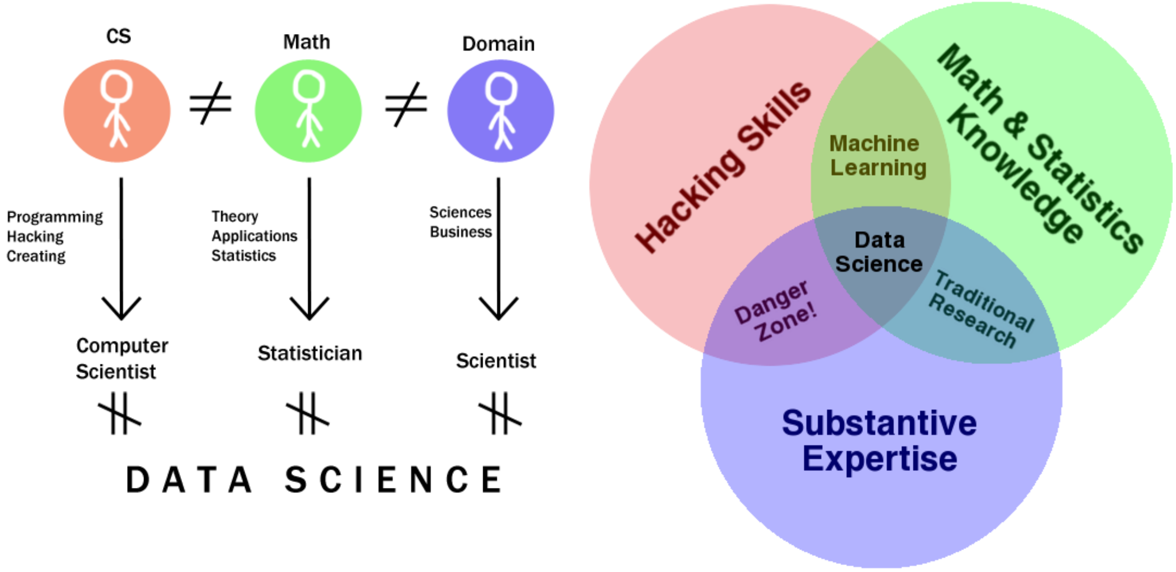


Figure 1: Data scientists do not come from a sole field of study – data science comes about through the culmination of three areas: computer science, mathematics, and a domain expertise.

Ultimately, data science aims to extract meaningful information from large and complex datasets, commonly referred to as big data, which, as aforementioned, is usually bounded by domain knowledge. This domain expertise is a stringent requirement: no matter what part of data science with which is being worked- database management, classification techniques, natural language processing, computer vision, or any other facet – this domain knowledge prerequisite is a vital, inescapable component. So what better way to make a software system than to embrace this idea? Learn2Mine tackles this in its design – especially in the back-end implementation.

B. Gamification

Gamification is a concept that is being embraced more and more from a pedagogical viewpoint. Put simply, gamification is the application of game mechanics in non-game settings (Deterding et al., 2011; Gerber, 2012). So pedagogical software that makes use of gamification is not to be considered a game, in any regard, but rather software that takes techniques proven to be useful in having people replay and enjoy video games and apply them in a pedagogical context. Additionally, gamification is often mixed up with "edutainment" – games which have a bonus educational goal. Learn2Mine fits this concept as it is primarily a learning tool with game elements sprinkled in rather than a game with educational elements sprinkled in. So the goal of gamification can be described as the invocation of joy and engagement which mirrors that of an actual game while retaining the ideas and thought-provoking nature of lessons and concepts that would be received in a classroom. Concrete concepts commonly implemented in applications utilizing gamification techniques include:

- Reward Schedules (Direkova, 2012; Glover, 2003; Muntean, 2011; Nah et al., 2013)
- Constant Feedback (Direkova, 2012; Muntean, 2011)
- Reputation (Direkova, 2012)
- Advanced User Paths (Direkova, 2012)
- Content Unlocks (Direkova, 2012; Muntean, 2011)
- Collaboration (Direkova, 2012; Muntean, 2011; Nah et al., 2013)

Gamification allows users to, in addition to the techniques above, fail without any repercussions, similar to a "Game Over" in a video game with a proceeding message asking the user to try again. This allows users to proceed in a trial-and-error style of problem solving which causes users to rely more on their own ability to solve problems, thusly creating an experience which is more engaging, from the user's perspective (Nah et al., 2013; Kapp, 2012). Rosalind, a platform for teaching students how to program common bioinformatics algorithms in the Python language, is a strong example of a system which implements these forms of gamification (Vyahhi et al.). Learn2Mine's gamification utilizes these gamification concepts:

- Immediate feedback and results upon lesson submission
- Users unlock various badge types for individual lessons
- Users can choose their "class" (programming language)
- Users can compete on leaderboards

III. Methods

A. Overall Learn2Mine Design

In order to capitalize on these gamification techniques, Learn2Mine has become a multi-faceted system. On the frontend, Learn2Mine has Google App Engine and RStudio as the primary interfaces for users, while, on the back-end, Learn2Mine has Galaxy. Bridges have been programmatically forged between these three systems in order to create what is wholly known as Learn2Mine and, additionally, this has bridged the gap between domain experts and algorithm developers. The communication between these environments is crucial in order for Learn2Mine to seamlessly implement these gamification techniques.

Google App Engine is a Platform as a Service (PaaS) which allows developers to build applications to run on Google's cyber-infrastructure. These applications are, naturally, cloud-based and, because of the cloud-nature, Learn2Mine is able to operate as a cloud application. This allows any user, regardless of operating system or browser, to use Learn2Mine. Selecting Google App Engine for the interface was crucial in that it is cheap (free with limited resources), allows for easy scalability, and supports many common languages (Python, Java, PHP, and Go). The Google App Engine frontend also allows for storage in a datastore – this datastore is a NoSQL database which can be programmatically queried with GQL queries.

Also on the frontend, RStudio is a browser-based IDE for creating, modifying, and executing R code. Learn2Mine has its own RStudio server for which users can request a username and password. Alternatively, users can deploy their own instance of RStudio. In this case, Learn2Mine has a set of packages that users are recommended to install in order to allow for more ease-of-use. However, users are not limited to the specifications that Learn2Mine outlines. Users can install any third-party package they desire as well as customize the layout of RStudio itself.

Google App Engine and RStudio would not amount to much if it were not for the back-end management provided by Learn2Mine's extended version of Galaxy. Galaxy is an open source project which was pioneered as a biomedical workflow analysis tool. The workflow system within Galaxy allows programs, which can be called from a terminal (or command line), to be interacted with at a scripting level. These programs, typically atomically broken down into a specific algorithm, are often referred to as tools. These tools can be written in any scripting language (e.g. Python, Perl, etc.) and are then masked by a user interface written in JavaScript and organized using an XML wrapper, which must be created along with the tool's script file (Blankenberg et al., 2001; Giardine et al., 2005; Goecks et al., 2010). Utilizing these concepts, Learn2Mine has repurposed and extended Galaxy for usage as a back-end system for grading and producing viable feedback. Learn2Mine initially created individual tools for running data science algorithms and for grading the results that students received. In its most novel implementation, Learn2Mine is now using Galaxy's workflow management system to create new, gradable lessons on the fly.

B. Lesson Creation and Performance

1. Alpha Deployment

The alpha deployment of the back-end of Learn2Mine utilized a very different schematic than the current, working version of the system (Turner et al., 2014; Turner et al., 2012). In this version of the deployment users were directly interacting with the back-end, there existed security issues, and the creation of new lessons required extensive knowledge of the Galaxy framework.

When used in its alpha stage, Learn2Mine required users to visit two (three if they utilized the RStudio server) sites when submitting lessons for grading. First, users would venture to the Learn2Mine virtual portfolio frontend in order to read about an algorithm or technique and to read the lesson which they are trying to perform. When a user believes they have a working answer, they proceeded to Galaxy. There is a lot to the Galaxy interface that may perplex a user, initially.

A common use case for this version of Learn2Mine would proceed as follows:

1) The user conducts all frontend needs and has an answer they would like to submit to Learn2Mine

2) The user proceeds to Galaxy
3) The user selects a grading tool and submits their solution
4) The user receives feedback about their solution with visualization
5) The user either refines their solution or proceeds to a new lesson

Having users submitting their answers in this fashion resulted in a very confounded interface in which users saw no gained benefit, but, programmatically, there existed a huge gain in potential.

Some lessons would require users to upload datasets to Galaxy so their algorithm can interact with those datasets. Every time a user uploaded anything to Galaxy, a file was created in the working directory of Galaxy, a folder which could become astronomically large very easily. This would not result in a slowdown when it comes to the performance of Galaxy, but could potentially become a problem if a user were to try and upload very large files for analysis. Learn2Mine wants to allow large files to be used, though, because analysis of big data is extremely important as that is, largely, what data science aims to tackle. So this presented a problem which would need to be addressed in the next iteration of Learn2Mine, as users did not enjoy this interface and there was a potential memory problem.

Additionally, a lot of the algorithms built into Galaxy resulted in the creation of even more data than that which was uploaded by the user. For example, the neural network tool that was included in the alpha version of Learn2Mine would run up to 10 neural network classifiers on a dataset. This would create 10 separate images in Galaxy's working directory folder as well as creating a comma-separated-values (CSV) file which contained the dataset and labels for each row of the test dataset. This creates a possible exponentiation of large datasets which can clog up memory on the back-end of Learn2Mine. As long as a user did not clear their history, then these images and data would never be purged.

The database managing the Galaxy back-end is initially packaged as a SQLite database, but, in order to allow for concurrency, Galaxy's universe_wsgi.ini file was altered to have a PostgreSQL database connection. This is done by altering the database_connection variable in the initialization file and, effectively, pointing to a database which was created for Learn2Mine. This database, however, ran into issues due to the aforementioned exponentiation of large datasets. Every user would have their own set of records in the database as Galaxy had to remember which user had access to what images, algorithm outputs, and datasets.

Efficiency was not the only problem with Learn2Mine's extension of Galaxy, however. Security was a problem with which could not be easily handled. Initially, users would navigate to Galaxy and could start running algorithms right away. Since Galaxy is inherently separate from the other branches of Learn2Mine, there had to be a way to tie a user's Galaxy session to their virtual portfolio on the frontend. The initial methodology was to create a key in the datastore of the virtual portfolio and have a user paste this key into a Galaxy tool. So this is a security hole because a user could impersonate another user by simply obtaining another user's key. Learn2Mine had built-in mechanisms for users to change their key if they felt that their key had been compromised, but that is not secure enough.

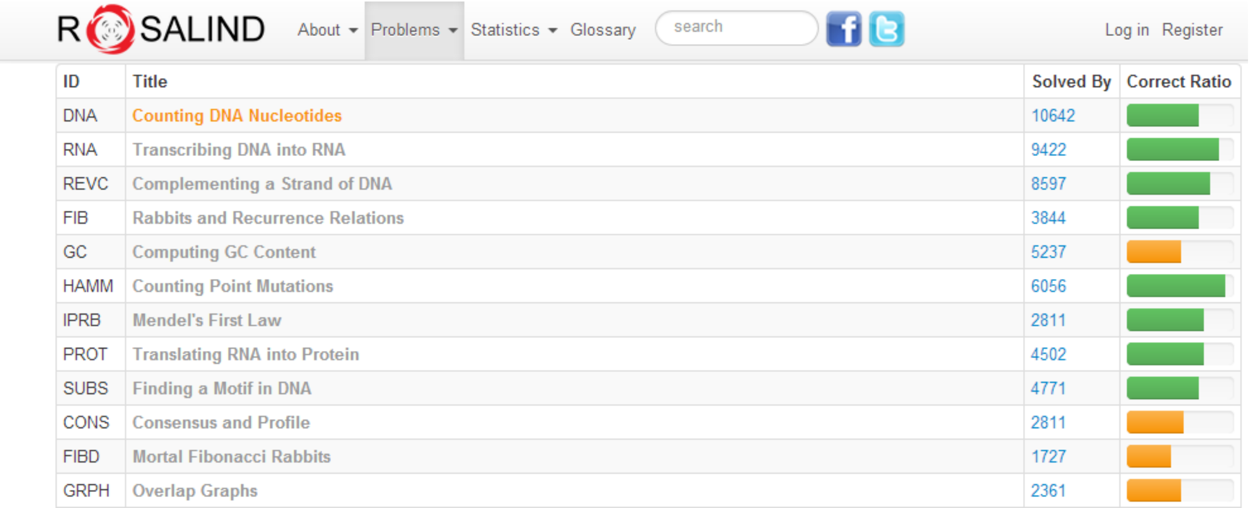
The iteration of Learn2Mine proceeding this utilized Galaxy's built-in mechanisms to allow OpenID to authenticate logins. Using OpenID was crucial for a few reasons. Users authenticate into the virtual portfolio using their Google account – Google accounts can be used for OpenID. So now users were being forced to create a login for Galaxy through OpenID. In order to receive credit for any lessons submitted, though, users would have to use the same email for the virtual portfolio and Galaxy. If the correct email address was incorrectly used on either end of Learn2Mine, then there would be datastore access faults, never seen by the user as this is all conducted in a post request, and correct submissions would not receive credit, but users would still be able to receive feedback for correct and incorrect answers since that is not reliant on a handshake between the virtual portfolio and Galaxy. So there is still an issue since users are being held responsible for a part of the process – ideally, users would never have to even create their own account on Galaxy, an issue tackled in the beta deployment of Learn2Mine.

In conjunction with the security issues, Learn2Mine had to build in a way for users to create accounts on RStudio. Again, this is a 3rd party system integrated into Learn2Mine and users need to be able to gain access onto RStudio so that they can code solutions to problems in R. Initially, users had RStudio accounts created whenever they ran the key tool in Galaxy. The way that this worked is that a user’s email would be their account name and their key number (or session ID) would be their password. So Galaxy and RStudio had the same credentials in the initial implementation of Learn2Mine and the RStudio password was able to be changed behind the scenes. This was forced to metamorphose whenever the Galaxy system was upgraded to the OpenID version of logging in. Whenever users associated their OpenID/Google account with Galaxy in this iteration of Learn2Mine, their encrypted account credentials were passed to a BASH script which would conduct a command line call that creates the RStudio account for that user. This version of the system did not allow for users to change their password on their own – whatever their Google account password was at the time of Galaxy account creation became their permanent RStudio password. The only way the password could be changed was for a user to email the core Learn2Mine team and then the RStudio account creation could be re-ran with a user’s new credentials, another security issue which was addressed in the beta version of Learn2Mine.

2. *Beta Deployment*

The current implementation of Learn2Mine is starkly different than its alpha predecessor. Galaxy is now hidden behind a representational state transfer interface (RESTful interface), users can submit lessons in multiple languages, and the security problem where users can impersonate other users is no longer in existence.

The frontend virtual portfolio now contains an Asynchronous JavaScript and XML (AJAX) call which sends a message to Galaxy. Galaxy needs to know specific information, which is passed through this call. The Galaxy RESTful API can be called through the passing of a unique API key, unique in that it refers to a specific Galaxy instance. This is a security safeguard so that outside users, users that know the technical details of Galaxy, cannot make calls to Learn2Mine’s instance of Galaxy. The next piece of information needed is a link to the workflows directory of the Galaxy instance referred to by the API key. This is done by simply appending a “/api/workflows/” to the Galaxy link. So, in Learn2Mine’s case: http://www.portal.cs.cofc.edu/learn2mine/api/workflows/ would be the link that Learn2Mine calls in this RESTful call. The next few parameters in the RESTful call refer to specific lessons in Learn2Mine. These arguments are workflow_id and history_id. In this iteration of Learn2Mine, each lesson corresponds to one specific workflow within Galaxy. In Galaxy, a workflow is a pipeline of tools running in succession with each tool in the workflow would produce its own history output. The reason for this is that a tool in Galaxy can be built in such a way that the input for one tool could be the output of another tool; so a pipeline makes the most sense as a structure for implementing a grading system. So a workflow is created for each lesson as each lesson has its own information, including amount of subproblems, the correct solution, and specific feedback. The back-end is able to communicate to the frontend through use of this workflow_id because a specific POST request can be created to communicate with the frontend and issue feedback to the user according to the degree of correctness of their solution. Each lesson, additionally, is associated with its own history. The history is referred to through the usage of the history_id variable. For the users, it does not matter what history that their work is filtered into, but it makes sense from a programmatic standpoint. This is a way to sort information about lessons and, eventually, new information will be available to users about lessons. Since each lesson has its own history, users will eventually be able to view the difficulty of a problem by seeing how often users got the problem incorrect and what percentage of users have completed a problem. This is a concept that is utilized very well in Rosalind and their bioinformatics implementation of such information can be seen in figure 2 (Vyahhi et al.). There is one last, significant parameter that is passed to Galaxy: the student's code. Galaxy accepts a user's code and decodes it. This decoding has to occur since the code is sent in a POST request and url POSTs have to be encoded upon sending so special characters and symbols can be sent across web pages. Formatting is able to be preserved in code in this same way – which is important for languages like Python where spacing is a vital element for code execution. Preservations of symbols is crucial because languages like R require blocks of code to be segmented off by curly braces, but formatting is not too important.



ID	Title	Solved By	Correct Ratio
DNA	Counting DNA Nucleotides	10642	🟩
RNA	Transcribing DNA into RNA	9422	🟩
REVC	Complementing a Strand of DNA	8597	🟩
FIB	Rabbits and Recurrence Relations	3844	🟩
GC	Computing GC Content	5237	🟧
HAMM	Counting Point Mutations	6056	🟩
IPRB	Mendel's First Law	2811	🟩
PROT	Translating RNA into Protein	4502	🟩
SUBS	Finding a Motif in DNA	4771	🟩
CONS	Consensus and Profile	2811	🟧
FIBD	Mortal Fibonacci Rabbits	1727	🟧
GRPH	Overlap Graphs	2361	🟧

Figure 2: Table view of Rosalind's lessons. The table shows statistical information about the difficulty of lessons by showing the amount of users that have solved the problem, while providing a visual progress bar showing the correctness ratio.

This iteration of Learn2Mine is the first to introduce the freedom for users to pick what language in which they desire to code. In the past, only R was supported as it is a very useful language for conducting statistical tasks and allows for ease when integrating packages pertinent to data science with graphical libraries (e.g. the ggplot2 library for visualization and the e1071 package for creating and using support vector machines). A lot of users do not have a lot of experience with R, so a need for supporting other languages arose. The only other language supported in this iteration of Learn2Mine is Python 2.7. Not using R makes the problems objectively more difficult because Python does not have a lot of the built-in statistical and data manipulation tools that R possesses. Because of this, Learn2Mine automatically incorporates 3rd party packages, including NumPy, SciPy, and pandas. NumPy is required because a lot of the return values for the lessons require a specific form of a vector and NumPy makes vector creation relatively easy. SciPy is included because it contains ways to manipulate data that are not included in the base installation of Python. The pandas library is implemented because it gives Python a lot of the advantages of R, such as the usage of dataframes and series, while still retaining the simplistic and easy-to-follow syntax of Python. Also, the pandas library contains a lot of tools which were created for use in machine learning algorithms.

Security is a much less significant problem within Learn2Mine. In previous iterations of Learn2Mine users had to handle authentication at some step in the process and, in order to be fully secure, users should not have to even know about cross-site authentication. Since users no longer have to navigate to Galaxy, there no longer exists a need to create an account on Galaxy. Instead, users are having their email, their username for all intents and purposes, sent as part of a POST request to Galaxy. When results are able to be gathered, Galaxy is able to send a POST request back to the virtual portfolio with that same email, eliminating any need for users to log in to Galaxy. In previous iterations of Learn2Mine, users could receive access to RStudio by creating a Galaxy account. Since users do not have direct access to Galaxy anymore, a new mechanism needs to be put in place to create RStudio accounts. Currently, users can request an account by contacting the core Learn2Mine team with a request, much like password changes were handled. Since RStudio is an IDE used only for the R language, it is useless to users who would rather code in Python, so there is no need to make an account for every user. To elaborate, the pandas library gives users a simplistic way to implement map-reduce algorithms, binning algorithms, reading in data with missing fields, etc.

C. Back-end of Gamification

Learn2Mine is a highly specialized version of a data science education platform. In addition to teaching students data science, Learn2Mine seeks to further motivate students to learn more by implementing gameful experiences through gamification. Specifically, the back-end of Learn2Mine aids in the following gameful experiences: awarding of badges and feedback mechanisms.

1. Badges

Badges are modified versions of achievements. Typically, in a video game you may receive an achievement for taking on and successfully completing a daunting task. In Learn2Mine you can earn these achievements by completing the final, challenging problem of a lesson. Even though there exist final, challenging problems for users to complete, there are still varying degrees of problems and difficulties. For example, there is a Basic R lesson which goes through basic techniques in the R programming language. After earning the first badge for showing proficiency in R, users can proceed to a harder version of the Basic R lesson, where all the problems, including the sub-problems, have increased in difficulty. So a user can have multiple badges for one skill with one requiring much more advanced skills to acquire.

These badges are not novel to Learn2Mine. While it is true that Learn2Mine comes up with its own badge images, its own lessons for the badges, and its own mechanisms for earning the badges, Learn2Mine's badge issuing schematic is an implementation of Mozilla's Open Badges standard (Knight et al., 2014). Mozilla Open Badges is meant to act as a virtual résumé. Typically, a person will claim to have a skill without a project or any kind of proof for having said skill. With Mozilla Open Badges one can show a badge to say that they have that requisite skill. This is because Mozilla Open Badges rely on having institutions creating them.

Badges work through the incorporation and communication between several JavaScript Object Notation (JSON) files. The storage of most of these files resides in Galaxy's static directory – a directory which hosts files that can be read by any facet of Learn2Mine. First, there must exist an “organization.json” file for Mozilla Open Badges to start allowing the issuing of badges. This JSON file contains a single JSON object containing the name of the issuer (Learn2Mine), an optional image (a logo), a url for the site that is issuing the requests, and an associated email address. This, effectively, is setting up administration and credibility for the badge issuer. Second, for each lesson that exists within Learn2Mine, a JSON file containing one JSON object must be created. This object contains the name of the badge (typically the lesson name for Learn2Mine), a brief description for the badge, an image for the badge for when it is showcased, tags used when searching badges, and an issuer. The issuer section refers back to the “organization.json” file. Next, when a user successfully completes a lesson and earns a badge, then a new JSON file is created. This JSON file, again, contains one JSON object containing a unique badge id, a recipient (Learn2Mine uses email as the identity here), an image (the same image that the lesson badge), a link to the badge’s JSON file, the issue time for the badge, and a verification by giving a link to where this individual badge will be located (Galaxy’s static directory).

2. *Feedback*

In video games, a motivational technique used is for users to reach a game over point. Typically, when a game over happens, the user has learned from mistakes and will not make the same mistakes, effectively giving the user permission to fail and retry until they have mastered the level. Learn2Mine implements this by dynamically issuing feedback to users whenever they submit an answer to a problem, or level. When a student submits a coding solution, their code is ran on Galaxy. Depending on the problem, some functions may be restricted and result in an automatic failure – for example, in the Conditionals R lesson, users are asked to find the minimum of a vector, but students will be marked incorrect if they try utilizing the built in minimum function. Otherwise, though, their code is ran to completion. In parallel, the solution code is also ran. The outputs of the two are compared – these outputs are usually variables, often taking the form of vectors and matrices, and those are compared. Any difference in the two variables results in the problem being incorrect, but, through the usage of regular expressions, extra spacing and problems will not cause a problem to be incorrect. There is an issue with overhead of adding new lessons because of this, however. Since Learn2Mine is expanding and attempting to give users freedom to use a programming language of their choice, a different solution has to be created for each individual language. So, as of right now, there are, at most, two solutions for most of the lessons currently in Learn2Mine.

IV. Results

A. *Alpha Deployment Results*

The alpha version of the system was piloted in the Introduction to Data Science (DATA 101) class at the College of Charleston in Fall 2013. The DATA 101 class covers an array of topics in data science without going into the depth seen in the data mining, statistical learning, and artificial intelligence classes at the College of Charleston, but, rather, the students survey different parts of the field.

In the Fall 2013 section of DATA 101 students were tasked, primarily, with learning the R programming language in parallel to the survey of data science techniques and algorithms. This allowed Learn2Mine to be a powerful tool for students to use. Students were slowly introduced to Learn2Mine as this was the first time Learn2Mine had been used outside of development.

Lesson	Major Category	Secondary Category	Submission Style	Avg. Number of Submissions Per Student	Number of Times Submitted
Introduction to R	Programming	Basic R	Results	17	1358
File IO	Programming	Intermediate R	Results	13	1005
Unknown Values	Data Science	Processing	Results	10	798
Prediction	Data Science	Prediction	Results	9	708
Prediction 2	Data Science	Prediction	Results	8	622
Introduction to R (Level 2)	Programming	Basic R	Results	8	614
File IO (Level 2)	Programming	Intermediate R	results	8	606
Prediction (Level 2)	Data Science	Prediction	Results	6	508
Functions	Programming	Intermediate R	Code	5	382
Loops with Functions	Programming	Intermediate R	Code	3	224
Conditionals with Functions	Programming	Intermediate R	Code	3	220
Predicting Stock Market	Data Science	Classification	Results	6	460
Evaluation Criteria	Data Science	Classification	Code	4	294
Simulated Trading	Data Science	Classification	Results	17	1365

Table 1: Results of Data Science 101 students submissions into Learn2Mine during the Fall 2013 semester at the College of Charleston. The last entry in the table was part of the students' final exam. Entries in the table are in order by the date assignments were issued.

B. Beta Deployment Results

The beta version of the system was piloted in the Data Mining (CSCI 334) class at the College of Charleston during the Spring 2014 semester. Heading into the beta, two hypotheses were formulated at the beginning of the semester:

Hypothesis 1: Students will prefer to complete lessons that are automatically graded with badges over traditional grading.

Hypothesis 2: The number of incorrect submissions will decrease if the students are given the badge/skill only if they successfully complete the lesson within the first 3 submissions.

V. Discussion

A. Analysis of Pilots

The alpha pilot has run to its completion, finishing in December 2013, while the beta pilot is still underway. The results from the DATA 101, alpha pilot influenced the creation of the hypotheses for the beta pilot.

Since the Learn2Mine system was not in its mature state of production at the time, it was obvious that students would have issues with the system. Usage of the system had an inherent learning curve which was grounded in the learning and utilization of the three systems belonging to Learn2Mine. One major part of this was the students' direct interaction with Galaxy. This learning curve coincides, additionally, with the programming skills that students were learning. In table 1, students' results can be viewed as they progressed in the class. Students using Learn2Mine became better programmers with the R language, as evidenced by the average number of submissions per student drastically decreasing toward the middle of the semester and continuing on toward the end of the semester. Looking at the first few lessons which had a very large number of submissions per student, it can be inferred that students submitted more often because they did not completely understand how Galaxy was able to grade their work. Students may have submitted the wrong information (e.g. submitting results of an algorithm rather than the algorithm itself) or mistakenly created duplicate submissions.

The alpha pilot had lessons being added approximately a week before the lessons were assigned to students. This was to promote adaptation to the students learning ability. If the DATA 101 students needed to focus on a specific technique, in the R language, for a little bit longer, then a quick switch was able to be made because of the nature and speed of lesson creation – it is vital to have this flexibility on an educational platform. For example, assignments 6, 7, and 8 (seen as records 6, 7, and 8 in table 1) were all second iterations of lessons that students had already seen. The reason for this is to refine the skills that the students possessed. These second iteration lessons allowed students to unlock more advanced versions of badges they had already obtained, reflecting their increased skill.

The second iteration lessons reflected students' progression and techniques they have obtained throughout their class and usage of Learn2Mine. This is evidenced through the fact that students were able to complete the harder version of the lesson in, on average, less attempts (see table 1), ultimately showing a progression of skill.

B. Future Work

1. Open Source

Learn2Mine was initially proposed as a learning environment that would be open source (Turner et al., 2012). This has not faded from the vision of Learn2Mine as open source is still a goal being worked towards. Learn2Mine's open source trajectory will follow the recipe given by Sourceforge, but will use its own version of each of the described facets (Sourceforge, 2014):

- Integrated Issue Tracking
  Learn2Mine will be using Trello to track issues in the system, this includes feature requests, bugs, and possible ideas. Trello allows developers to communicate with the other core developers, open source contributors, and end users of a single system. Developers can create cards which signify a single issue, as predefined. Trello also allows the sorting of issues by priority, difficulty, etc.
- Threaded Discussion Forums
  Discussion forums are the least mature part of Learn2Mine's aim to become open source as this issue has not been implemented at all. Ultimately, a discussion forum will be able to integrate with the code repository. The discussion forum can be viewed as an alternative to issue tracking, but also supports the ability for users to ask questions for clarity or submit information which could be a bug, or, perhaps, just a misuse of an already implemented feature.
- Code Repository

Currently, the code repository is up-to-date and lies within a private Bitbucket repository. Eventually, this is going to be migrated to Github for more exposure so developers that want to be involved can more easily get involved with the project.

- Documentation

  Documentation is always in a state of needing to be updated within Learn2Mine as new iterations of the Learn2Mine system requires completely new documentation. The current documentation for Learn2Mine depicts how developers can add frontend information for new lessons to Learn2Mine. This includes the creation of badges and setting up the communication between the virtual portfolio and Galaxy. The documentation currently does not include a tutorial to teach users how to add a workflow to Galaxy, effectively fully adding a lesson to Learn2Mine.

2. Future Lessons

The current implementation of lessons in Learn2Mine is going to completely evolve with the next iteration of the system. Currently, there is a Python file which generates all the text for lessons; this includes set up code, reminders, and the problem description. This Python file finds the correct lesson by coupling the usage of conditionals with templates providing by the webapp2 framework provided by Google App Engine. This is all on the frontend and, realistically, this is something that could be more cleanly implemented on the back-end of the system.

Learn2Mine is currently being set up to move all the information about lessons to the datastore, located on the back-end of the virtual portfolio. This will allow lessons to be added simply by making a simple GQL insert query into the datastore, something can be done without having to update the Learn2Mine system. This increases the system's availability, the percentage of time in which a system remains functional and operational, as Learn2Mine does not have to be taken down and pushed back up to the web with updated code.

3. Expansion to Other Universities

Learn2Mine seeks to teach data science and not just to College of Charleston students. As aforementioned, data science is growing across the nation and world and so should the technologies teaching such a discipline. As there currently does not exist a program like Codecademy or Rosalind for strict data science, it is hoped that Learn2Mine can become a widespread tool for teaching data science. The first step for becoming widespread is to have other universities adopt this pedagogical tool.

Adoption can happen in two different ways: universities can either contribute to the current implementation of Learn2Mine by adding lessons or universities could extend Learn2Mine as an open source project and make their own version. While it is hoped that all universities creating lessons would add their contributions to the main Learn2Mine system, it is important to consider that universities may want to make their lessons private or, perhaps, even completely alter the system itself within their own installation.

4. Conclusion

Learn2Mine is a pedagogical tool utilized in order to teach users data science. Learn2Mine is constantly ramping up the amount of lessons it contains in order to encompass all vital aspects of data science: computer science, statistics, and domain expertise. By being able to teach people all three facets of data science, data scientists will start to become a standard in industry; every company and institution can benefit from data scientists, but the proper teaching mechanisms have never been present, until now.

VI. References

Anderson, P., Bowring, J., McCauley, R., Pothering, G., & Starr, C. (2014). An Undergraduate Degree in Data Science: Curriculum and a Decade of Implementation Experience. Proceedings of the 45th Technical Symposium on Computer Science Education.

Argamon, S. (n.d.). Data Science @ IIT. Retrieved from http://www.iit.edu/csl/cs/programs/undergrad/data_science_undergrad.shtml

Asaj, N., Bastian, K., Poguntke, M., Schaub, F., Weber, M., Ritter, D., … Groh, F. (2012). Gamification: State of the Art Definition and Utilization. In Research Trends in Media Informatics (pp. 39–45). Ulm, Germany. Retrieved from http://vts.uni-ulm.de/docs/2012/7866/vts_7866_11380.pdf

Blankenberg, D., Kuster, G. Von, Coraor, N., Ananda, G., Lazarus, R., Mangan, M., … Taylor, J. (2001). Galaxy: A web-based genome analysis tool for experimentalists. Current Protocols in Molecular Biology Edited by Frederick M Ausubel et Al, Chapter 19.

Cheshire, C. (n.d.). UC Berkeley School of Information: Data Science. Retrieved from http://www.ischool.berkeley.edu/research/datascience

Deterding, S., Sicart, M., Nacke, L., O'Hara, K., & Dixon, D. (2011). Gamification. using game-design elements in non-gaming contexts. In Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems - CHI EA '11 (pp. 2425–2428). New York, New York, USA: ACM Press. doi:10.1145/1979742.1979575

Direkova, N. (2012). Game On: 16 Game Mechanics for User Engagement. Slideshare. Retrieved August 02, 2013, from http://www.slideshare.net/gzicherm/nadya-direkova-game-on-16-design-patterns-for-user-engagement

Domínguez, A., Saenz-de-Navarrete, J., De-Marcos, L., Fernández-Sanz, L., Pagés, C., & Martínez-Herráiz, J. J. (2012). Gamifying learning experiences : Practical implications and outcomes. Computers & Education, (63), 380–392. Retrieved from http://www.sciencedirect.com/science/article/pii/S0360131513000031

Dubois, P. L. (n.d.). UNC Charlotte: Data Science and Business Analytics Initiative. Retrieved from https://dsba.uncc.edu/academic-programs

Gerber, H. (2012). Can Education be Gamified?: Examining Gamification, Education, and the Future. Huntsville, Texas. Retrieved from http://shsu.academia.edu/HannahGerber/White-Papers

Giardine, B., Riemer, C., Hardison, R. C., Burhans, R., Elnitski, L., Shah, P., … Nekrutenko, A. (2005). Galaxy: A platform for interactive large-scale genome analysis. Genome Research, 15, 1451–1455. doi:10.1101/gr.4086505

Glover, I. (2003). Play As You Learn : Gamification as a Technique for Motivating Learners. Sheffield, UK. Retrieved from http://www.editlib.org/p/112246?nl

Goecks, J., Nekrutenko, A., & Taylor, J. (2010). Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. Genome Biology, 11, R86. doi:10.1186/gb-2010-11-8-r86

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. (2009). The WEKA Data Mining Software. SIGKDD Explorations, 11(1), 10–18.

Kapp, K. (2012). The Gamification of Learning and Instruction. (M. Davis, M. Zelenko, K. D. Davies, D. Kilgore, R. Taff, & B. Morgan, Eds.) (p. 48). San Francisco, CA: Pfeiffer.

Knight, E., Casilli, C., Lee, S., Goligoski, E., McAvoy, C., Brennan, B., Larsson, M., Klein, J., Varelidi, C., Varma, A., Cole, M., Forester, J. (2014). Mozilla Open Badges. Retrieved from http://www.openbadges.org/

Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., & Euler, T. (2006). YALE: Rapid Prototyping for Complex Data Mining Tasks. In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining (Vol. 2006, pp. 935–940). doi:10.1145/1150402.1150531

Muntean, C. I. (2011). Raising engagement in e-learning through gamification. In The 6th International Conference on Virtual Learning ICVL 2011 (pp. 323–329). Retrieved from http://icvl.eu/2011/disc/icvl/documente/pdf/met/ICVL_ModelsAndMethodologies_paper42.pdf

Nah, F. F., Telaprolu, V. R., Rallapalli, S., & Venkata, P. (2013). Gamification of Education Using Computer Games Background: Gamification and Its Application to Education. Rolla, Missouri, USA. Retrieved from http://link.springer.com/chapter/10.1007/978-3-642-39226-9_12#page-1

Sourceforge (2014). Create a Project. Slashdot Media. San Francisco, CA. Retrieved from http://sourceforge.net/create/

Thom, J., Millen, D. R., Dimicco, J., & Street, R. (2012). Removing Gamification from an Enterprise SNS. In Incentives (pp. 1067–1070). Seattle, WA. Retrieved from http://dl.acm.org/citation.cfm?id=2145362

Turner, C. A., & Anderson, P. E. (2012). Learn2Mine: An Open-Source Cloud-Based Informatics Platform For Integrated Teaching and Data Exploration. *IEEE International Conference on eScience, 8.* Retrieved from: http://www.ci.uchicago.edu/escience2012/pdf/escience2012_submission_194.pdf

Turner, C. A., Dierksheide, J. L., & Anderson, P. E. (2014). Learn2Mine: Data Science Practice and Education through Gameful Experiences. *International Journal of e-Education, e-Business, e-Management and e-Learning, 4,* 243-248. Retrieved from: http://www.ijeeee.org/Papers/338-C00021.pdf

Vyahhi, N., Compeau, P., Balandin, A., Kladov, A., Sosa, E., Dvorkin, M., … Rayko, M. (n.d.). Rosalind. Retrieved January 08, 2013, from http://rosalind.info/

Wolstencroft, K., Haines, R., Fellows, D., Williams, A., Withers, D., Owne, S., … Goble, C. (n.d.). The Taverna workflow suite: designing and executing workflows of Web Services on the desktop, web or in the cloud. Nucleic Acids Research. Retrieved from http://nar.oxfordjournals.org/content/41/W1/W557