OBILLSK: Using Predictive Analytics to Anticipate Interlibrary Loans

Ryan Litsey, Scott Luker, and Weston Mauldin
Texas Tech University, USA

Abstract

The methods of assessment used by academic libraries have evolved to include a variety of methodologies and outcomes. Until now assessment has been under the purview of research methods and is usually survey based. Developments in technology and computing power have led to new types of computational power that can harness cutting edge algorithms to open new avenues of research. These avenues are predictive analytics and machine learning. Harnessing computer algorithms can lead to an assessment that not only occurs in near real time, but also creates a system that can respond to patron preferences at a moment's notice. Our paper seeks to describe how these types of technology can be designed using everyday library questions. We also demonstrate the potential power these types of assessment can have. It is the hope of the authors to begin a discussion of a new type of assessment, one that does not rely on static data, but rather modern computer power to provide assessment as patrons interact with the library, creating a data biosphere instead of statics-assessed data.

Assessment continues to play a large role in the academic library. Libraries use assessment for a variety of reasons—everything from budget justification, collection decisions, instructional ideas and several others. What is important in assessment above all else is data collection. Data collection can take on a variety of formats. Data could be gathered statically by data gathering software like circulation systems. Data can be input into an aggregator like Springshare or other metric collection systems. All of these systems work well when trying to analyze static data. What we mean by static data is data that must first be stored, then accessed in a specific way and analyzed. The downside to static data, aside from its temporal nature, is that there is little you can do in the way of data-driven decision making at the point of input. A library cannot collect real-time data and then deploy it to solve useful problems immediately. The literature on the importance of library assessment is extensive. Much of it is geared toward demonstrating the value of the academic library. As Megan Oakleaf argues in *Value of Academic Libraries: A Comprehensive Review and Report*,[1] libraries should create assessment management systems. These systems seek to demonstrate the importance of the academic library to the larger university. While this is a noble cause and one that is indeed relevant, there needs to be a next step. That step is applying the data collected to make accurate, adaptable, and quick real time decisions, at the point of data generation. In order to take assessment systems to the next level, we have to examine cutting edge computer development. While an assessment management system is well and good for demonstrating library value, it is a post analysis. Even if the data was gathered yesterday, the decision is already made. If the data shows that the value of the library is drifting from the needs of the user, the damage is already done by the time it can be corrected by the next assessment. What we need is real time analysis completed by a thinking/learning machine. It is only then we can harness the data and quickly deploy the results to aid library staff in adapting seamlessly to the needs of the user. No longer is it justifying needs. The library becomes a reflection of the user by adapting and learning in real time. This is the next evolution in assessment.

In order to accomplish such a monumental challenge, we started small with a library unit that already had experience with data collection. Interlibrary loan requests are tracked comprehensively through a variety of systems. To that end we developed a system called the Online Based Inter-Library Loan Statistical Kit (OBILLSK). This system is a user-activated data harvesting system. It gathers data from a user's ILLiad SQL database, sends that information to a webserver, and presents visualized ILL data for an entire consortium for analysis. We programmed the system with a variety of tools. Visual Studio software was used to write C# source code in the .NET framework for the client software and web application. The website is supported by a Microsoft SQL Server database. The front-end framework utilizes a variety of tools focused on data visualization concepts, which include Bootstrap, jQuery, ShieldUI, and jVectorMap. The interactive map is populated by JSON-formatted text files, which are periodically generated by a Python script. The Python script was developed using Aptana Studio. We chose these tools because OBILLSK was designed to extract, analyze, and display data from multiple ILLiad databases. We decided the best approach would be to mirror the system requirements of ILLiad provided in Atlas Systems documentation. In order to maximize programming time, we used several third-party JavaScript and CSS libraries rather than designing the web interface from scratch. These front-end frameworks are HTML5 compliant, incorporate native responsive design, and use AJAX for efficient communication with server-side scripts. The reason we set up the system this way was largely based on the amount of data we were seeking to analyze.

The amount of data required to provide meaningful statistics for multiple institutions was substantial. As of this writing, the database table used to hold the ILL records contains approximately 7.5 million records. The first challenge was to build an efficient and secure solution for acquiring the ILL data from member institutions. The second challenge was to calculate and display the statistical analysis on the website barring excessive load times. We developed desktop software for users to download and execute on local workstations to acquire the data. The software prompts users to enter connection credentials for their ILL database. A .csv file is generated and saved on the user's workstation. This process allows the user to view the data prior to sending to OBILLSK. Please note that no patron data is queried by the software or included in the file. The file is then uploaded to the OBILLSK website. The entire process of generating and uploading the data takes one to ten minutes depending on the amount of transactional data included in the file. The development team was provided with a series of metrics used to calculate various turnaround times. This process was automated using a series of SQL stored procedures allowing for the calculations to be performed at any desired frequency. The basic idea was to store the results of the calculations in an ancillary database table and reference the web application instead of performing the calculations on every page load. One of the most significant lessons learned with regards to system efficiency was database field indexing. Once we indexed key fields, such as transaction number and ILL number, the stored procedures and page load times significantly increased. With the ability to analyze the ILL data from up to 35 different institutions we turned our technological development questions inward and began to ask ourselves what we could do with this data aside from justify the importance of consortial ILL. This question led to the development of a learning machine, using predictive analytics and K-means clustering that we have developed to not only predict ILL requests, but also mathematically model the libraries' entire collection in real time. Using Google's Tensorflow open source machine learning algorithms, we were then able to teach a computer to analyze and make decisions based on this behavior. The system we designed we have taken to calling the Automated Library Information Exchange Network, or ALIEN.

The idea for ALIEN came after development for OBILLSK was well underway. As we mentioned in the OBILLSK section of the paper, we were already efficiently capturing ILL request data to analyze the turnaround times for ILL transactions between various universities. We wanted to know if we could use the same data from OBILLSK to predict how many times a university would request a book in future semesters. Though both OBILLSK and ALIEN begin with data from ILLiad databases, the two programs use the data for their own unique purposes. The next section describes how ALIEN uses the ILLiad data to make predictions about how libraries make ILL requests.

ALIEN used the exact same .csv file that is used in OBILLSK. The OBILLSK .csv file was used by a Python script to generate a new .csv file that broke down the number of requests for a book by year, semester, and week. For example, a single row from this new file of book counts contains the book's OCLC number, the calendar year the requests were made in, the total number of requests for that book in the spring, summer, and fall semesters, and the total number of requests for that book on a per week basis. Requests are broken into either a completed request or a cancelled request based on the status changes of the finished request transaction. With OBILLSK, we were only concerned with recent data, but for ALIEN, we needed as much data as possible. Processing almost ten years of OBILLSK data was taking a few hours, so we decided to make the data more efficient. Since ALIEN does not use all of the data that OBILLSK does, we were able to make an ALIEN version of the OBILLSK client that extracted a much smaller subset of data from the ILLiad databases. By slimming down the OBILLSK

.csv file, we reduced the processing time from a few hours to less than 10 minutes without losing any important data.

After the book counts file is created, a second Python script makes the predictions for the next year's requests. The book counts file is grouped by OCLC number and year, so ALIEN creates predictions one book at a time. The structure of a book prediction contains the OCLC number of the book, the year being predicted, the predicted range of requests for that book in the spring, summer, and fall semesters, and how confident the ALIEN system is in its predictions. The data of the first year the book is requested is used to make a conservative base line prediction. Book requests from subsequent years are used to shape the prediction to be more accurate. As ALIEN becomes more confident in its predictions, the predicted range of requests will begin to tighten. Though this technique led to accurate predictions in some cases, there were enough problems in other situations that made us reevaluate how ALIEN looked at the data we gave it. The next section will give some details about what problems we encountered and what changes are being made to overcome these problems.

As mentioned in the previous section, we encountered a few problems that made us look at predicting ILL requests in a different light. This section will explain the main problem that came up and how ALIEN is being adapted to offer more useful predictions.

While there were many smaller problems, most of them fell under the larger problem of lack of information. Information is the most important resource in machine learning and predictive analytics, but oftentimes there are gaps in the data that must be worked around. For a typical machine learning system, it can take dozens or even thousands of generations of data before the system can learn to be truly accurate. With the initial design of ALIEN, a single generation of data for a given book was one year of ILLiad data. Since our ILLiad data only goes back to 2006, a book could have at most 10 generations of data for ALIEN to learn from. For many of our most popular requests, the books would have data for only two to four generations. The system can begin to make predictions off of fewer generations, but having more data creates a more robust system. Additional generations could be added to some books by accounting for different book editions, but we are still limited to 10 generations because of the amount of ILLiad data. Other factors to consider are new professors favoring different books for similar classes, new classes being added, old classes being removed, classes changing between spring, summer, and fall semesters, and classes changing from being offered year round to being offered a single semester. By reviewing the generational limit and other data limiting factors, we decided that focusing on singular books may not be the best approach for ALIEN. This decision led us to data clustering. Data clustering is grouping large amounts of data into a much smaller number of clusters in order to give clearer high level analysis.

Rather than basing predictions off of individual books, we turned to basing predictions off of the requests as a whole. Individual books did not give us as many data generations as we would like, so instead we have begun looking at a book's genre and subject. ALIEN extracts a list of OCLC numbers from its previously made book counts file and queries WorldCat to fill in the genre and subject for each book Texas Tech has requested through ILL. We are currently working to compile a list of OCLC numbers from our circulation and collection data. Once we have data from these three sources (requests, circulation, and collection), we will use data clustering to highlight what genres and subjects are important to our library. After discovering the most important genres and subjects, the library can make more informed decisions about what kinds of books should be added to their collection or continue to be requested through ILL.

References

1. Association of College and Research Libraries, *Value of Academic Libraries: A Comprehensive Research Review and Report*, researched by Megan Oakleaf (Chicago: Association of College and Research Libraries, 2010), http://www.ala.org/acrl/sites/ala.org.acrl/files/content/issues/value/val_report.pdf.