Online Courses Recommendation based on LDA

Rel Guzman Apaza, Elizabeth Vera Cervantes, Laura Cruz Quispe, José Ochoa Luna
National University of St. Agustin
Arequipa - Perú
{r.guzmanap,elizavvc,lvcruzq,eduardo.ol}@gmail.com

Abstract

In this paper we propose a course recommendation system based on historical grades of students in college. Our model will be able to recommend available courses in sites such as: Coursera, Udacity, Edx, etc. To do so, probabilistic topic models are used as follows. On one hand, Latent Dirichlet Allocation (LDA) topic model infers topics from content given in a college course syllabus. On the other hand, topics are also extracted from a massive online open course (MOOC) syllabus. These two sets of topics and grading information are matched using a content based recommendation system so as to recommend relevant online courses to students. Preliminary results show suitability of our approach.

1 Introduction

Nowadays, the amount of educational resources spread at Internet is huge and diverse (Martin, 2012). Massive Online Open Courses (MOOCs) such us Coursera, Udacity, EdX, to name a few, are gaining momentum (Fischer, 2014). It is possible to find courses from almost every knowledge domain. This vast offer overwhelm any user willing to find courses according his/her background. This task can be tedious because it involves access to each platform, search available courses, select some courses, read carefully each course syllabus, and choose appropriate content. This process can be unmanageable if we extend our search beyond online courses to educational content.

In this work we propose a system for online courses recommendation, although MOOCs courses are primarily focused. To do so, we rely on Topic Models (Blei, 2012), an unsupervised probabilistic generative model, which given a set of documents and a number of topics as input, automatically returns a relevant set of words probabilistically associated for each topic. Why this scheme is valuable?, consider for instance a huge number of digitalized books of a public library, this algorithm can automatically discover main topic words and therefore allows one to gain insights about content in books.

Currently educational systems and data mining is an emerging research area (Romero and Ventura, 2010), these systems use different recommendation techniques in order to suggest online learning activities, based on preferences, knowledge and data from other students with similar interests (Romero et al., 2007). In (Kuang et al., 2011) the author provides resource recommendation for users in the e-learning system based on contents and user log activities. There was proposed a method for resource recommendation based on topic modeling in an e-learning system, that system used Latent Dirichlet Allocation (LDA) to get a low dimension vector, and to do inference it used Gibbs sampling, then in resource recommendation it applied cosine similarity in document topic distribution to find neighbor resources. The authors from (Haruechaiyasak and Damrongrat, 2008) also recommended documents, in this case it recommended articles from wikipedia by calculating the similarity measures among topic distributions of the articles. The model proposed in (Sadikov and Bratko, 2011) is an hybrid recommendation system where the core of the system is a linear regression model, based on stochastic gradient descent. For predicting the rank of a lecture, they used and compared the predictions made by content-based and collaborative-based methods. In this paper they established manually the attributes that represent each video-lecture, unlike our paper, where the attributes for the courses are defined by the LDA algorithm. In (Sadikov and Bratko, 2011), to find a rank they measured the correlation between an old lecture (a lecture the visitor has already seen), and the new lectures (lectures that visitor has not seen yet), and then they ordered theses measures in a list, where the lowest comes first, theses computations were used in the linear regression model. Also they said that there was not to much difference between using content-based or collaborative-based methods, but they said that their system could have been improved if they used textual attributes, which is our case.

In our proposal, Latent Dirichlet Allocation (LDA) (Blei et al., 2003) topic model is mainly used as feature descriptor of courses. Thus, we assume that each course has a set of inherent topics and therefore relevant words that summarize them. In our content-based recommendation setting those are input features that describe courses. We are concerned in discovering the parameter vector of users, i.e., weights over topic words that denote user preferences on courses. In order to infer this user vector, we rely on supervised machine learning algorithms thus, we assume grading obtained in college courses as ratings, learn user weights and ratings are predicted for unseen MOOCs courses. Preliminary results show suitability of this approach.

The paper is organized as follows. In Section 2 background is given. In Section 3, our proposal is presented. Section 4 shows experimental results. Finally, Section 5 concludes the paper.

2 Background

2.1 Probabilistic Topic Modeling

Topic models are probabilistic models that have been mainly used to discover topics in a big collection of text documents. They are non supervised learning (Duda et al., 2012) techniques that do not require any prior annotations or labeling of the documents: the topics emerge from the analysis of the original texts (Blei, 2012). To do so, they assume each document is a combination of topics and each topic is a probability distribution over words (Blei et al., 2003). Topic models are a type of graphical model based on Bayesian networks.

The generative process described by a topic model does not make any assumptions about the order of words as they appear in documents. The only information relevant to the model is the number of times words are produced, this is known as the “bag-of-words” assumption (Steyvers and Griffiths, 2007).

There are two main topic models: LDA (Blei et al., 2003) and Probabilistic Latent Semantic Analysis (pLSA) (Hofmann, 1999). In this work we use LDA due to its general model. It is also worth noting that LDA has been previously used in recommendation systems (Romero and Ventura, 2010; Romero et al., 2007; Kuang et al., 2011).

2.2 Topics Modeling using Latent Dirichlet Allocation

Latent Dirichlet allocation (LDA) (Blei et al., 2003) is widely used for identifying topics in a set of documents, building on previous work by Hofmann (Hofmann, 1999). The corresponding graphical model representation is depicted in Figure 1, where each document is represented as a mixture of a fixed number of topics, with topic z receiving weight $\theta_z^{(d)}$ in document d, and each topic is a probability distribution over a finite vocabulary of words, with word i having probability $\phi_i^{(z)}$ in topic z.

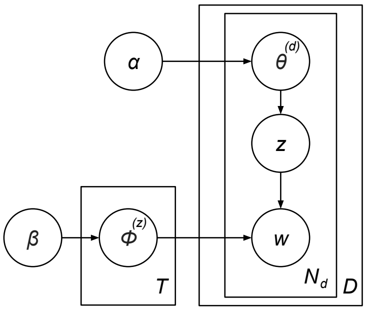

Figure 1: Graphical model for the topic modeling using plate notation

Symmetric Dirichlet priors are placed on $\theta^{(d)}$ and $\phi^{(}j)$, with $\theta^{(d)} \sim \text{Dirichlet}(\alpha)$ and $\phi^{(}j) \sim \text{Dirichlet}(\beta)$, where α and β are hyper-parameters that affect the sparsity of these distributions. The hyper-parameter α can be interpreted as a prior observation count for the number of times a topic is sampled in a document, and β as the prior observation count on the number of times words are sampled from a topic before any word from the corpus is observed. This smooths the word distribution in every topic, with the amount of smoothing determined by β. The goal of inference in this model is to identify the values of ϕ and θ, given a corpus of D documents represented by a vocabulary of W words.

In our proposal, each course is a document d that has its related sequence of N_d word tokens, N words in the overall corpus.

2.3 Gibbs Sampling Algorithm

There are many algorithms proposed to obtain the main variables of interest θ and ϕ in the literature, (Hofmann, 1999) used the expectation-maximization (EM) algorithm, this approach suffers from problems involving local maxima of the likelihood function, which has motivated a search for better estimation algorithms like the ones proposed in (Blei et al., 2003; Buntine, 2002; Minka and Lafferty, 2002).

Instead of directly estimating the variables for each document, another approach is the algorithm called "Gibbs sampling" (Griffiths and Steyvers, 2004), which provides a relatively efficient method of extracting a set of topics from a large corpus. Gibbs sampling considers each word token in the text collection in turn, and estimates the probability of assigning the current word token to each topic, conditioned on the topic assignments to all other word tokens. From this conditional distribution, given a document, a topic is sampled and stored as the new topic assignment for this word token. We write this conditional distribution as:

$$P(z_j|z_{N\backslash j}, w_N) = \frac{n^{(w_j)}_{z_j,N\backslash j} + \beta}{n^{(\cdot)}_{z_j,N\backslash j} + W\beta} \cdot \frac{n^{(d_j)}_{z_j,N\backslash j} + \alpha}{n^{(d_j)}_{\cdot,N\backslash j} + T\alpha}$$

where:

$w_N = (w_1, \ldots, w_N)$ are the words in the entire corpus

$z_N = (z_1, \ldots, z_N)$ are the topic assignments of the words

$z_{N\backslash j}$ indicates $(z_1, \ldots, z_{j-1}, z_{j+1}, \ldots, z_N)$

W is the size of the vocabulary

$n^{(w_j)}_{z_j,N\backslash j}$ is the number of times a word wj is assigned to topic z_j

$n^{(\cdot)}_{z_j,N\backslash j}$ is the total number of words assigned to topic z_j

$n^{(d_j)}_{z_j,N\backslash j}$ is the number of times a word in document d_j is assigned to topic z_j

$n^{(d_j)}_{\cdot,N\backslash j}$ is the total number of words in document d_j

From this probability distribution it is possible to make inference, in order to compute conditional probability of topic structure given the observed document. The probability distribution of topics in a document represents a feature vector for that document.

2.4 Recommender Systems

According to (Ricci et al., 2011), recommender systems are software tools and techniques providing items suggestions for a given user. Suggestions provided are aimed at supporting their users in various decision-making processes, such as what items to buy, what music to listen, or what news to read.

As a rule, in a recommendation-system application there are two classes of entities, which we shall refer to as users and items. Users have preferences for certain items and these preferences must be teased out of the data (Rajaraman and Ullman, 2012). The data itself is represented as a utility matrix, giving for each user-item pair, a value that represents what is known about the degree of preference of that user for that item. Values come from an ordered set, e.g., integer $1-5$ representing the number of stars that the users gave as a rating for that item. We assume that the matrix is sparse, meaning that most entries are unknown. An unknown rating implies that we have no explicit information about the user's preference for the item. The goal of a recommendation system is to predict the blanks in the utility matrix.

There are two basic architectures for a recommendation system (Rajaraman and Ullman, 2012):

- Content-based systems focus on properties of items. Similarity of items is determined by measuring the similarity in their properties

- Collaborative-Filtering system focus on the relationship between users and items. Similarity of items is determined by the similarity of the ratings of those items by the users who have rated both items.

In a content-based system, we must construct a profile for each item, which is a record of collections of records representing important characteristics of that item. In simple cases, the profile consist of some characteristics of the item that are easily discovered. For example, in a movie there are the set of actors, the director, the genre of general type of movie. In documents it is not immediately apparent what the values of features should be. There are many kinds of documents for which a recommendation system can be useful. For example, there are many news articles published each day, and we cannot read all of them. A recommendation system can suggest articles on topics a user is interested in. Unfortunately, documents do not tend to have available information giving features. A substitute that has been useful in practice is the identification of words that characterize the topic of a document. An approach is to compute the TF(Term frequency) - IDF(Inverse document frequency) score for words in the document. The ones with the highest scores are the words that characterize the document. In this sense, documents are represented by sets of words. In this paper we have used a different approach which relies on finding document topic information by using topic modeling algorithms such as LDA.

3 Proposal

In order to recommend online courses, each course is considered a document which has a given content. To characterize each course, LDA is used to uncover the semantic structure hidden in the document. Since LDA allow us to get a topic distribution for each course, this output is used as a feature vector for courses (items according to a content-based recommendation setting). A recommendation system is built using item profiles and utility matrices and we treat the problem as one of machine learning. Regard the given data as a training set, and for each user, build a classifier that predicts the rating of all items. The rest of this section describes our main design choices.

courses	*c. features*	*profile user(1)—rating ...*
Calculus	$x_1, \dots x_n$	$\theta_1^{(1)}, \dots, \theta_n^{(1)}$ — **12**
...	...	...
ML(Mooc)	$x_1^{'}, \dots x_n^{'}$	$\theta_1^{(1)}, \dots, \theta_n^{(1)}$—**?**

Table 1: Utility matrix for courses

Consider the utility matrix in Table 1 used to represent a content-based recommendation system. First column contains courses names (college and MOOC's courses). Second column contains feature descriptors for courses. Each row denotes a different course, therefore each course has a different feature vector. Third column shows the user vector profile $\Theta^{(1)}$ for user 1. This vector could comprise user 1 preferences about art, math, biology and social sciences in general. In this same column is also showed user 1 ratings for each course (they are in fact grades obtained in college for user 1, see for instance rating 12 for calculus). Further columns for user 2, user 3 and so on should be added accordingly. Our goal is to predict missing ratings for MOOC's courses (? symbol in last row) for user 1 (user 2, 3, etc.). In order to do so, we should perform the following steps:

- Extract item vectors for courses: item vectors are defined by courses content, i.e., text that describes courses, such as "about the course" information. In order to construct item vectors (features from documents), we rely on Latent Dirichlet Allocation algorithm which extracts topic information from text as probability distribution of words. Since we use a machine learning setting, item vectors are features of a regression/classification problem, which we denote $X = \{X_1, X_2, \dots, X_n\}$.

- Learn user's vector: interests about topic courses can be modeled by user's vector which should be learned for each user. To do so, we use a machine learning approach, all available ratings (grading information in college) are used to train a multilinear regression model (Bishop and others, 2006). The user's vector is therefore the resulting set of parameters (or weights), $\Theta^{(1)} = \{\theta_1^{(1)}, \dots, \theta_n^{(1)}\}$

learned from training data (for instance, all courses and gradings of user 1). There are m (number of users) set of parameters. In a multi-linear regression algorithm we want to find the values for Θ, that minimize the cost function: $J(\Theta_0, \Theta_1, \ldots, \Theta_n) = \frac{1}{2m}\sum_{i=1}^{m}(h_\Theta(x^{(i)}) - y^i)^2$

We define an hypothesis:
$h_\Theta(x) = \Theta^T x = \Theta_0 x_0 + \Theta_1 x_1 + \Theta_2 x_2 + \ldots + \Theta_n x_n$
Where $\Theta_0, \Theta_1, \ldots, \Theta_n$ are the parameters we want to predict minimizing the cost function. One way to minimize the cost function is by using gradient descent method, where each iteration of gradient descent makes the parameters θ_j come closer to the optimal values that will minimize the cost function $J(\theta)$.

For $n > 1$
Repeat {
$\Theta_j := \Theta_j - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\Theta(x^{(i)}) - y^i){x_j}^{(i)}$
(simultaneously update Θ_j for j = 0, ...n) }

- Given item and user vectors the goal is to predict a rating R_C for a MOOC course C with feature vector X_C for user U, i.e., user vector profile $\Theta^{(U)}$, the resulting predicted rating is given by:

$$R_C = X_C^T \Theta^{(U)}$$

An overview of the recommendation system is depicted in Figure 2 where we estimate the ratings for a student and to recommend a course we consider a "top-10 best recommendations" approach thus, each student get always 10 recommended courses. Those are the most related MOOCs to courses in which a student get the 10 lowest grades.

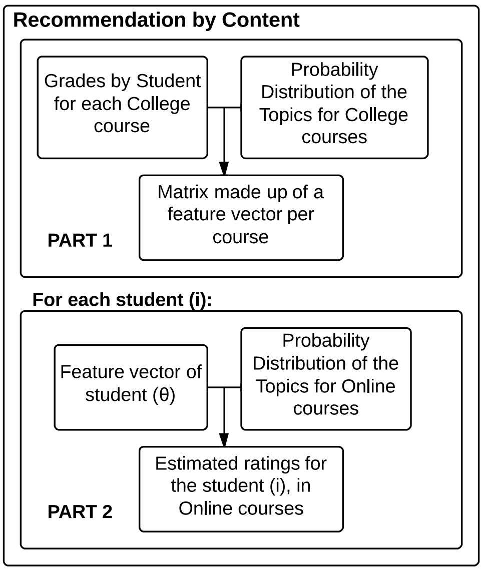
Recommendation by Content
Grades by Student for each College course
Probability Distribution of the Topics for College courses
Matrix made up of a feature vector per course
PART 1
For each student (i):
Feature vector of student (θ)
Probability Distribution of the Topics for Online courses
Estimated ratings for the student (i), in Online courses
PART 2

Figure 2: Block diagram for the recommendation system

4 Experimental Results

This section shows preliminary experimental results conducted on real world data sets. Courses and users grading information where extracted from a Peruvian university. Some MOOC's courses were extracted from Coursera, the following categories were considered: "business and management", "computer science - artificial intelligence", "computer science - software engineering", "computer science - systems and security", "computer science - theory", "mathematics", "statistics and data analysis". The most significant information from each course is given by "Introduction", "About the Course" and "FAQ" sections.

All extracted information has been preprocessed according to the following process: remove non ASCII characters, strip HTML tags, remove special strings, remove multiple spaces and blank lines.

After that we built a corpus further used by the LDA algorithm. The number of Coursera courses considered was 69, while the number of college courses was 43, which gives rises to 112 courses. The topic modeling algorithm used the gibbs sampling inference procedure and according to (Blei, 2012) we set parameters $\alpha = 50/T$, $\beta = 0.01$. The number of iterations was chosen to be large enough to guarantee convergence, $N = 200$.

To measure performance, accuracy was considered by counting the number of correct matches between college courses and Coursera courses. Figure 3 illustrates the impact of the number of topics T in the topic model. A higher accuracy is achieved when we use a higher number of topics, then we set the number of topics $T =$ number of Coursera courses because of the precision.

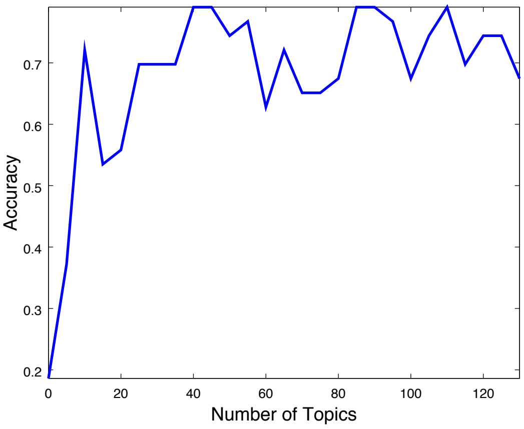

Figure 3: Accuracy of the recommendation system according to the number of topics, a better precision and also efficiency is obtained when the number of topics is equal to the number of coursera courses, T =69

The goal of our proposal is to recommend courses for students who have received low grades in college therefore, we are using grades as ratings. To keep a recommendation system setting, we have decided to invert grading information thus, 20 grade turns out 0 rating and viceversa (this step might not be necessary in other recommendation systems). Mean normalization is also used to get a more reliable recommendation for students with few grades available, for instance, first year students.

For testing, we define a variable "top-N" which denotes the number of courses to recommend. For instance, for student "a" we recommend the "top-N" courses from Coursera where he/she has gotten the greatest ratings. In Figure 4, the x-axis denotes several values for "top-N", and the y-axis denotes accuracy obtained. An cccuracy over 0.6 is achieved for "top-N" greater than or equal to 10.

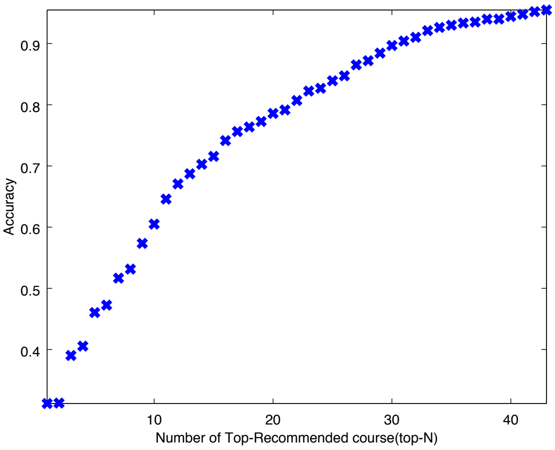

Figure 4: Recommendation system accuracy according to the number of recommended courses

In Figure 5, a comparison between ratings of "coursera courses" and "college courses" for one student is showed. We intend to show proximity of predicted data (ratings on "coursera courses") and provided data (ratings on college courses).

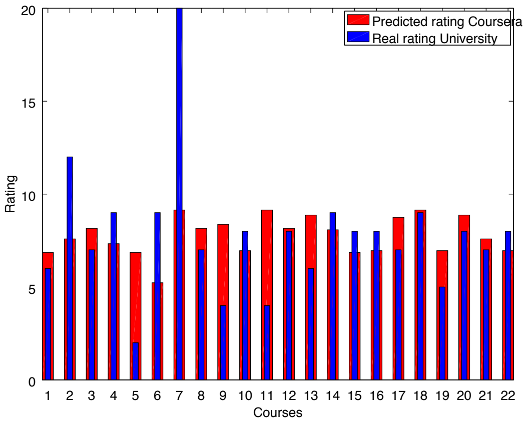

Figure 5: Comparison chart of ratings for one student

5 Conclusion

We have introduced a novel approach for recommending online courses that combines the probabilistic topic model LDA and content-based recommendation systems. In short, we use a machine learning approach where LDA allow us to extract feature descriptors from courses, rating prediction in this setting is performed by inferring user profile parameters using multilinear regression. Preliminary experimental results show that our algorithm performs well when compared to a similar approach based on cosine similarity with LDA.

Although we have focused on MOOCs as source of recommendation content, nothing prevent us from using this approach beyond such domain. In fact, further domains can be included by performing feature topic extraction. Future work will be addressed to investigate scalability issues. In this sense, topic models such as LDA, have scalable versions available. For instance, a MapReduce implementation is given in the Apache Mahout library[1]. There are also scalable versions for multilinear regression.

References

Christopher M Bishop et al. 2006. *Pattern recognition and machine learning*, volume 1. springer New York.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March.

David M Blei. 2012. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84.

Wray Buntine. 2002. Variational extensions to em and multinomial pca. In *Machine Learning: ECML 2002*, pages 23–34. Springer.

Richard O Duda, Peter E Hart, and David G Stork. 2012. *Pattern classification*. John Wiley & Sons.

Gerhard Fischer. 2014. Beyond hype and underestimation: identifying research challenges for the future of moocs. *Distance Education*, 35(2):149–158.

Thomas L Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National academy of Sciences of the United States of America*, 101(Suppl 1):5228–5235.

Choochart Haruechaiyasak and Chaianun Damrongrat. 2008. Article recommendation based on a topic model for wikipedia selection for schools. 5362:339–342.

Thomas Hofmann. 1999. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM.

Wei Kuang, Nianlong Luo, and Zilei Sun. 2011. Resource recommendation based on topic model for educational system. 2:370–374, Aug.

Fred G. Martin. 2012. Will massive open online courses change how we teach? *Commun. ACM*, 55(8):26–28, August.

Thomas Minka and John Lafferty. 2002. Expectation-propagation for the generative aspect model. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, pages 352–359. Morgan Kaufmann Publishers Inc.

Anand Rajaraman and Jeffrey David Ullman. 2012. *Mining of massive datasets*. Cambridge University Press, Cambridge.

Francesco Ricci, Lior Rokach, and Bracha Shapira. 2011. *Introduction to recommender systems handbook*. Springer.

C. Romero and S. Ventura. 2010. Educational data mining: A review of the state of the art. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 40(6):601–618, Nov.

Cristbal Romero, Sebastin Ventura, JoseAntonio Delgado, and Paul De Bra. 2007. Personalized links recommendation based on data mining in adaptive educational hypermedia systems. 4753:292–306.

Er Sadikov and Ivan Bratko. 2011. Recommending videolectures with linear regression.

Mark Steyvers and Tom Griffiths. 2007. Probabilistic topic models. *Handbook of latent semantic analysis*, 427(7):424–440.

[1] https://mahout.apache.org/