Course Recommendation System

Picture 2

Under the Guidance of

Ms. Charu Gupta

Assistant Professor, CSE Dept

Submitted By: Guide Signature

Achintya Sarkar , 00720802715, C.S.E A

Himanshu Ratnani , 03220802715 , C.S.E A

Abhishek Jain, 00620802715 , C.S.E A

Aayush Tiwari, 00420802715 , C.S.E

**Introduction**

Nowadays, the amount of educational resources spread at Internet is huge and diverse . Massive Online Open Courses (MOOCs) such us Coursera, Udacity, EdX, to name a few, are gaining momentum[3] . It is possible to find courses from almost every knowledge domain[1]. 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 course reviews , NLP [4] and we use a set of tools for using sentiments and classify courses based on them . Accordingly each course is assigned to a set of users who get a particular scores based on correlation[2] .

Each document is a combination of topics and each topic is a probability distribution over words . Topic models are a type of graphical model based on Bayesian networks or SVM Classifiers [1].

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 [1].

**Problem Statement**

**Need For the project**

Nowadays, the amount of educational resources spread at Internet is huge and diverse . Massive Online Open Courses (MOOCs) such us Coursera , Udacity, EdX, to name a few, are gaining momentum. It is easy for a sophomore or even some senior year students to get lost in choosing the right course . The recommendation system we propose is trying to solve the issue where the course is recommended based on users general rating score based on their answers and academic record.

**Survey**

**Work Done**

The current progress has been the data collection and labelling of reviews from websites . The algorithms used to make a classifier for the reviews and get its sentiment has been able to differentiate and give scores to reviews with a classifier.score of 70% but it is a work in progress for multi class classification for differential scoring of reviews . The training of the classifier was done for only english reviews and nltk library was used to get the features from the classifier.

**Future Work**

Our Next step would be to implement a recommendation algorithm for the labeled courses and also to optimise the obtained features such that only the necessary information is extracted from each review and the training takes lesser time to implement . But before training appropriate training data (User , Item matrix etc. ) has to be extracted. Once goes as planned the recommendation system might be able to work in real word cases .

**Design Phase**

**Review Classification**

1. **Loading the data (Data Cleaning )**

The training data included lots of reviews in language other than english . This might have become a problem so we used enchant library [5] for language detection and to get clean data . To remove stop words we used nltk.stopwords[6] from nltk library .

**2. Word features**

To get the most common and used words so as to make them as the features for training in the algorithm (Naive Bayes Binary classifier [7]). For this we used nltk library and found the top 3000 most common words and set them as feature.

**3. Training & Testing the algorithm**

To train the algorithm (Naive Bayes [7]) we used the nltk provided classifier .train function which took approx 5 min on training 1500 reviews

and gave a classifier.score of approx 70%.

The desired accuracy might be possible if we adjust the feature words so as and also implement a multi class classifier .

**References**

[1] GitXiv General Adaptive Research Models with Code From: [http://www.gitxiv.com](http://www.gitxiv.com/)

[2]. Arxiv Cornell University database of Archived Research : [https://arxiv.org](https://arxiv.org/)

[3]. Github Public Database For Open Projects : <https://github.com/achin1tya/NLP>

[4]. Sentiment Analysis Tools Research Based Open Articles Platform :

<https://medium.com/@datamonsters/sentiment-analysis-tools-overview-part-1-positive-and-negative-words-databases-ae35431a470c>

[5] enchant : <https://pypi.org/project/pyenchant/>

[6] nltk stopwords :<https://pythonprogramming.net/wordnet-nltk-tutorial/>

[7] Naive bayes classifier :<https://pythonprogramming.net/naive-bayes-classifier-nltk-tutorial/>