

INTRODUCTION

1.1 Natural Language Processing

Natural Language Processing (NLP) refers to the use and ability of systems to process sentences in a natural language such as English, Telugu rather than in a specialized artificial computer language such as C, C++, java etc; The development of NLP applications is challenging because computers traditionally require humans to speak to them in computer programming language that is precise, unambiguous and highly structured commands. Human language, however, is not always precise. It is often ambiguous and the linguistic structure can depend on many complex variables, including slang, regional dialects and social context. The ultimate goal of NLP is to do away with computer programming languages altogether. Instead of specialized languages such as Java or Ruby or C, there would only be "human".

1.2 History

The first use of computers to manipulate natural languages was in the 1950s with attempts to automate translation between Russian and English [Locke & Booth]. These systems were spectacularly unsuccessful requiring human Russian-English translators to pre-edit the Russian and post-edit the English. Based on World War II code breaking techniques, they took individual words in isolation and checked their definition in a dictionary. They were of little practical use. Popular tales about these systems cite many mis-translations including the phrase "hydraulic ram" translated as "water goat".

In the 1960s NLP systems started to examine sentence structure but often in an ad hoc manner. These systems were based on pattern matching and few derived representations of meaning. The most well-known of these is Eliza [Weisenbaum] though this system was not the most impressive in terms of its ability to extract meaning from language.

Serious developments in NLP took place in the early & mid 1970s as systems started to use more general approaches and attempt to formally describe the rules of the language they worked with. LUNAR [Woods 1973] provided an English interface to a database holding details of moon rock samples. SHRDLU [Winograd] interfaced with a virtual

robot in a world of blocks, accepting English commands to move the blocks around and answer questions about the state of the world. Since that time there has been parallel development of ideas and technologies that provide the basis for modern natural language processing systems. Research in computer linguistics has provided greater knowledge of grammar construction [Gazdar] and Artificial Intelligence researchers have produced more effective mechanisms for parsing natural languages and for representing meanings [Allen]. NLP systems now build on a solid base of linguistic study and use highly developed semantic representations.

Up to the 1980s, most NLP systems were based on complex sets of handwritten rules. Starting in the late 1980s, however, there was a revolution in NLP with the introduction of machine learning algorithms for language processing. This was due both to the steady increase in computational power resulting from Moore's Law and the gradual lessening of the dominance of Chomsky theories of linguistics (e.g. transformational grammar), whose theoretical underpinnings discouraged the sort of corpus linguistics that underlies the machine-learning approach to language processing. Some of the earliest-used machine learning algorithms, such as decision trees, produced systems of hard if-then rules similar to existing hand-written rules. Increasingly, however, research has focused on statistical models, which make soft, probabilistic decisions based on attaching real-valued weights to the features making up the input data. Such models are generally more robust when given unfamiliar input, especially input that contains errors (as is very common for real-world data), and produce more reliable results when integrated into a larger system comprising multiple subtasks.

Many of the notable early successes occurred in the field of machine translation, due especially to work at IBM Research, where successively more complicated statistical models were developed. These systems were able to take advantage of existing multilingual textual corpora that had been produced by the Parliament of Canada and the European Union as a result of laws calling for the translation of all governmental proceedings into all official languages of the corresponding systems of government. However, most other systems depended on corpora specifically developed for the tasks (and often continues to be) a major limitation in the success of these systems. As a result, a great deal of research has gone into methods of more effectively learning from limited amounts of data.

Recent research has increasingly focused on unsupervised and semi supervised learning algorithms. Such algorithms are able to learn from data that has not been hand-annotated with the desired answers, or using a combination of annotated and non-annotated data. Generally, this task is much more difficult than supervised learning, and typically produces less accurate results for a given amount of input data. However, there is an enormous amount of non-annotated data available (including, among other things, the entire content of the World Wide Web), which can often make up for the inferior results.

1.3 Major tasks in NLP

The following is a list of some of the most commonly researched tasks in NLP. Some of these tasks have directly real-world applications, while others more commonly serve as subtasks that are used to aid in solving large tasks.

- **Anaphora (Linguistics):** It is the use of an expression the interpretation of which depends upon another expression in context (it's antecedent or postcedent).
- **Automatic summarization:** Produce a readable summary of a chunk of text. Often used to provide summaries of text of a known type, such as articles in the financial section of a newspaper.
- **Collocation Extraction:** It is the task of extracting collocations automatically from a corpus using a computer. Collocation is defined as a sequence of words or terms which co-occur more often than would be expected by chance. 'crystal clear', 'middle management', 'nuclear family', 'riding boots', 'cosmetic surgery', 'motor cyclist' are some of the examples of collocated pair of words.
- **Coreference resolution:** Given a sentence or larger chunk of text, determine which words ("mentions") refer to the same objects ("entities").
- **Discourse analysis:** This rubric includes a number of related tasks. One task is identifying the discourse structure of connected text, i.e., the nature of the discourse relationships between sentences (Ex: elaboration, explanation, contrast). Another possible task is recognizing and classifying the speech acts in a chunk of text (Ex: yes-no question, content question, statement, assertion, etc.,).

- **Machine Translation:** Automatically translate text from one human language to another. This is one of the most difficult problems, and is a member of a class of problems colloquially termed "AI-complete", i.e., requiring all of the different types of knowledge that humans possess (grammar, semantics, facts about the real world, etc.) in order to solve properly.
- **Morphological segmentation:** Separate words into individual morphemes and identify the class of the morphemes. The difficulty of this task depends greatly on the complexity of the morphology (i.e., the structure of words) of the language being considered. English has fairly simple morphology, especially inflectional morphology, and thus it is often possible to ignore this task entirely and simply model all possible forms of a word (Ex: "open, opens, opened, opening") as separate words. In languages such as Turkish, however, such an approach is not possible, as each dictionary entry has thousands of possible word forms.
- **Named Entity Recognition:** It is a task to discover the Named Entities in a document and then categorize these NES into diverse Named Entity classes such as Name of Person, Location, River, Organization etc.,
- **Natural Language Generation:** Convert information from computer databases into readable human language.
- **Natural Language Understanding:** It is a subtopic of NLP in artificial intelligence that deals with machine reading comprehension. It converts chunks of text into more formal representations such as first-order logic structures that are easier for computer programs to manipulate. Natural Language Understanding involves the identification of the intended semantic from the multiple possible semantics which can be derived from a natural language expression which usually takes the form of organized notations of natural language concepts. Introduction and creation of language metamodel and ontology are efficient however empirical solutions. An explicit formalization of natural languages semantics without confusions with implicit assumptions such as closed world assumption vs. open world assumption or subjective yes/no vs. objective

true/false is expected for the construction of a basis of semantics formalization.

- **Optical Character Recognition (OCR):** Given an image representing printed text, determine the corresponding text.
- **Part-of-speech (POS) tagging:** In corpus linguistics, POS tagging also called as grammatical tagging or word-category disambiguation, is the process of marking up a word in a text (corpus) as corresponding to a particular part of speech, based on both its definition and its context i.e., its relationship with adjacent and related words in a phrase, sentence or a paragraph.
- **Parsing:** It is also called as syntactic analysis, is the process of analyzing a string of symbols, either in natural language or in computer languages, conforming to the rules of a formal grammar and generate a parse tree for a given sentence.
- **Question Answering:** Given a human-language question, determine its answer. Typical questions have a specific right answer (such as "What is the capital of Canada?"), but sometimes open-ended questions are also considered (such as "What is the meaning of life?").
- **Relationship Extraction:** Given a chunk of text, identify the relationships among NES (Ex: who is the wife of whom?).
- **Sentence Breaking:** It is also known as "Sentence boundary disambiguation". On giving a chunk of text, find the sentence boundaries. Sentence boundaries are often marked by periods or other punctuation marks.
- **Sentiment Analysis:** Extract subjective information usually from a set of documents, often using online reviews to determine "polarity" about specific objects. It is especially useful for identifying trends of public opinion in the social media, for the purpose of marketing.
- **Speech Recognition:** When a sound clip of a person or people speaking is given as input, then Speech Recognizer analyze the input and need to give corresponding text as output.

- **Speech Synthesis:** When a text is given as input to speech Synthesizer, it need to analytic the and give corresponding speech as output.
- **Speech Segmentation:** Given a sound clip of a text people separate it into A of Speech Recognition and typically grouped With it,
- **Topic Segmentation and Recognition:** Given chunk of tent, seperate it into segments of which is denoted to a topic. and identify the topic Of the segment.
- **Word Segmentation:** Separate a chunk of continuous text into scratatc words. For a language like English, this is fairly trival, since words are usually separated by spaces. However, some written languages like Chinese. Japanese and Thai do not mark word boundaries in such a fashion, and In those languages text segmentation is a significant task requiring knowledge of the vocabulary and morphology of words in the language.
- **Word Sense Disambiguation (WSD):**Many words have more than one meaning; we have to select the meaning which makes the most sense in context. For this problem, we are typically given a list of words and associated word senses, Ex: from a dictionary or from an online resource such as WordNet.

In some cases, sets of related tasks are grouped into subfields of NLP that are often considered separately from NLP as a whole. Examples include:

- **Information Retrieval (IR):** This is concerned with storing, searching and retrieving information. It is a separate field within computer science (closer to databases), but IR relies on some NLP methods (for example, stemming). Some current research and applications seek to bridge the gap between IR and NLP.
- **Information Extraction (IE):** This is concerned in general with the extraction of semantic information from text. This covers tasks such as NER, Conference resolution, Relationship Extraction, etc.,

1.4 Problems

Recommender systems help users by predicting interesting products and services in situations where the number and complexity of offers fastens the user's capability to survey them and make a decision. Such systems are capacitated to predict suggestions that a user would give it on an item or any social entity. As stated in, recommender systems help in providing users with recommendations about items that people with similar views and preferences have liked in the past. There are two ways in which recommender systems can produce recommendations collaborative filtering and content-based filtering. Both of these approaches are often combined and are termed as Hybrid Recommender Systems. Collaborative filtering is one of the major approaches used for recommender systems that depend on the opinions and views expressed by other users. For example, Foursquare recommends places based on the previous check-ins of the user. Content-based filtering analyzes a series of different characteristics of an item in order to recommend other items with similar properties. For example, Flipkart.com suggests items based on the number of people purchasing an item and the feedback on the item.

Based on the above concepts, we have proposed a system that is quite useful in case when a user goes to a new place and needs some good recommendations for different items. Based on the previous feedbacks on the places related to the queried item nearby the user, the proposed system builds score analysis with the help of sentiment analysis. Sentiment analysis is an approach to find out users' opinions. It comes as a part of natural language processing and text analytics to determine and extract useful information from the given information. Sentiment analysis helps in identifying positive and negative responses, emotions and views. We use a hybrid recommender system approach for performing sentiment analysis. The proposed system has been integrated with a cloud platform.

In everyday life, people make decisions based on the past experiences and histories. Likewise, our proposed system takes into account the past visited places of the user and the feedback on the same item. Our system uses LA to perform the score analysis based on the users past responses on a particular item or place. The proposed LA approach is useful in estimating the number of positive or negative responses of a place or item by an individual user. The use of LA approach for recommender systems is demonstrated in recent literature. Forsati et al. describes an approach for effective page recommendation using the concepts of distributed learning automata and association rule mining. Talabeigi

et al. illustrates a method for web recommendation system that uses hybrid approach and concepts of cellular automata.

1.5 SENTIMENT ANALYSIS

Sentiment analysis (sometimes known as opinion mining or emotion AI) refers to the use of natural language processing, text analysis, computational linguistics, and biometrics to systematically identify, extract, quantify, and study affective states and subjective information. Sentiment analysis is widely applied to voice of the customer materials such as reviews and survey responses, online and social media, and healthcare materials for applications that range from marketing to customer service to clinical medicine.

Generally speaking, sentiment analysis aims to determine the attitude of a speaker, writer, or other subject with respect to some topic or the overall contextual polarity or emotional reaction to a document, interaction, or event. The attitude may be a judgment or evaluation, affective state (that is to say, the emotional state of the author or speaker), or the intended emotional communication.

1.6 Motivation

In this work, we use LA concepts along with sentiment analysis for recommendation on cloud platforms and social networks such as Foursquare and Facebook. In the earlier days of Internet, finding places was very difficult. But currently, various social networking sites such as Facebook and Foursquare possess a lot of social network information, which allows using their data sets to analyze and integrate our recommender system with them and, thus, helping finding places nearby user's location.

This dissertation work is organized as follows. Literature survey for Learning automata based sentiment analysis for recommender system is presented in chapter 2. A detailed description Learning automata based sentiment analysis for recommender system is provided in chapter 3. Problem statement and proposed work is presented in chapter 4. Algorithm and System design is presented in chapter 5. Experimental results and performance metrics are presented in chapter 6.

LITERATURE SURVEY

Literature survey is the most important step in software development process. Before developing the tool it is necessary to determine the time factor, economy n company strength. Once these things are satisfied, ten next step is to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need lot of external support. This support can be obtained from senior programmers, from book or from websites. Before building the system the above consideration are taken into account for developing the proposed system.

2.1History of Learning Automata based Sentiment Analysis for Recommender System on cloud

The accuracy of a sentiment analysis system is, in principle, how well it agrees with human judgments. This is usually measured by variant measures based on precision and recall over the two target categories of negative and positive texts. However, according to research human raters typically only agree about 80% of the time (see Inter-rater reliability). Thus, a program which achieves 70% accuracy in classifying sentiment is doing nearly as well as humans, even though such accuracy may not sound impressive. If a program were "right" 100% of the time, humans would still disagree with it about 20% of the time, since they disagree that much about any answer. On the other hand, computer systems will make very different errors than human assessors, and thus the figures are not entirely comparable. For instance, a computer system will have trouble with negations, exaggerations, jokes, or sarcasm, which typically are easy to handle for a human reader: some errors a computer system makes will seem overly naive to a human. In general, the utility for practical commercial tasks of sentiment analysis as it is defined in academic research has been called into question, mostly since the simple one-dimensional model of sentiment from negative to positive yields rather little actionable information for a client worrying about the effect of public discourse on e.g. brand or corporate reputation.

In recent years, to better fit market needs, evaluation of sentiment analysis has moved to more task-based measures, formulated together with representatives from PR agencies and market research professionals. The focus in e.g. the RepLab evaluation data set is less

on the content of the text under consideration and more on the effect of the text in question on brand reputation.

Web 2.0

The rise of social media such as blogs and social networks has fueled interest in sentiment analysis. With the proliferation of reviews, ratings, recommendations and other forms of online expression, online opinion has turned into a kind of virtual currency for businesses looking to market their products, identify new opportunities and manage their reputations. As businesses look to automate the process of filtering out the noise, understanding the conversations, identifying the relevant content and actioning it appropriately, many are now looking to the field of sentiment analysis. Further complicating the matter, is the rise of anonymous social media platforms such as 4chan and Reddit. If web 2.0 was all about democratizing publishing, then the next stage of the web may well be based on democratizing data mining of all the content that is getting published.

One step towards this aim is accomplished in research. Several research teams in universities around the world currently focus on understanding the dynamics of sentiment in e-communities through sentiment analysis. The CyberEmotions project, for instance, recently identified the role of negative emotions in driving social networks discussions.

The problem is that most sentiment analysis algorithms use simple terms to express sentiment about a product or service. However, cultural factors, linguistic nuances and differing contexts make it extremely difficult to turn a string of written text into a simple pro or con sentiment. The fact that humans often disagree on the sentiment of text illustrates how big a task it is for computers to get this right. The shorter the string of text, the harder it becomes.

Even though short text strings might be a problem, sentiment analysis within micro blogging has shown that Twitter can be seen as a valid online indicator of political sentiment. Tweets' political sentiment demonstrates close correspondence to parties' and politicians' political positions, indicating that the content of Twitter messages plausibly reflects the offline political landscape.

2.2 Approaches for Learning Automata based Sentiment Analysis for Recommender System on cloud

A basic task in sentiment analysis is classifying the polarity of a given text at the document, sentence, or feature/aspect level—whether the expressed opinion in a document, a sentence or an entity feature/aspect is positive, negative, or neutral. Advanced, "beyond polarity" sentiment classification looks, for instance, at emotional states such as "angry", "sad", and "happy".

Precursors to sentimental analysis include the General Inquirer, which provided hints toward quantifying patterns in text and, separately, psychological research that examined a person's psychological state based on analysis of their verbal behavior.

Subsequently, the method described in a patent by Volcani and Fogel, looked specifically at sentiment and identified individual words and phrases in text with respect to different emotional scales. A current system based on their work, called EffectCheck, presents synonyms that can be used to increase or decrease the level of evoked emotion in each scale.

Many other subsequent efforts were less sophisticated, using a mere polar view of sentiment, from positive to negative, such as work by Turney, and Pang who applied different methods for detecting the polarity of product reviews and movie reviews respectively. This work is at the document level. One can also classify a document's polarity on a multi-way scale, which was attempted by Pang and Snyder among others: Pang and Lee expanded the basic task of classifying a movie review as either positive or negative to predict star ratings on either a 3- or a 4-star scale, while Snyder performed an in-depth analysis of restaurant reviews, predicting ratings for various aspects of the given restaurant, such as the food and atmosphere (on a five-star scale).

First steps to bringing together various approaches—learning, lexical, knowledge-based, etc.—were taken in the 2004 AAAI Spring Symposium where linguists, computer scientists, and other interested researchers first aligned interests and proposed shared tasks and benchmark data sets for the systematic computational research on affect, appeal, subjectivity, and sentiment in text.

Even though in most statistical classification methods, the neutral class is ignored under the assumption that neutral texts lie near the boundary of the binary classifier, several researchers suggest that, as in every polarity problem, three categories must be identified. Moreover, it can be proven that specific classifiers such as the Max Entropy and the SVMs can benefit from the introduction of a neutral class and improve the overall accuracy of the classification. There are in principle two ways for operating with a neutral class. Either, the algorithm proceeds by first identifying the neutral language, filtering it out and then assessing the rest in terms of positive and negative sentiments, or it builds a three-way classification in one step. This second approach often involves estimating a probability distribution over all categories (e.g. naive Bayes classifiers as implemented by the NLTK). Whether and how to use a neutral class depends on the nature of the data: if the data is clearly clustered into neutral, negative and positive language, it makes sense to filter the neutral language out and focus on the polarity between positive and negative sentiments. If, in contrast, the data are mostly neutral with small deviations towards positive and negative affect, this strategy would make it harder to clearly distinguish between the two poles.

A different method for determining sentiment is the use of a scaling system whereby words commonly associated with having a negative, neutral, or positive sentiment with them are given an associated number on a -10 to $+10$ scale (most negative up to most positive) or simply from 0 to a positive upper limit such as $+4$. This makes it possible to adjust the sentiment of a given term relative to its environment (usually on the level of the sentence). When a piece of unstructured text is analyzed using natural language processing, each concept in the specified environment is given a score based on the way sentiment words relate to the concept and its associated score. This allows movement to a more sophisticated understanding of sentiment, because it is now possible to adjust the sentiment value of a concept relative to modifications that may surround it. Words, for example, that intensify, relax or negate the sentiment expressed by the concept can affect its score. Alternatively, texts can be given a positive and negative sentiment strength score if the goal is to determine the sentiment in a text rather than the overall polarity and strength of the text.

2.2.1 Subjectivity/objectivity identification

This task is commonly defined as classifying a given text (usually a sentence) into one of two classes: objective or subjective. This problem can sometimes be more difficult than polarity classification. The subjectivity of words and phrases may depend on their context and an objective document may contain subjective sentences (e.g., a news article quoting people's opinions). Moreover, as mentioned by Su, results are largely dependent on the definition of subjectivity used when annotating texts. However, Pang showed that removing objective sentences from a document before classifying its polarity helped improve performance.

2.2.2 Feature/aspect-based

It refers to determining the opinions or sentiments expressed on different features or aspects of entities, e.g., of a cell phone, a digital camera, or a bank. A feature or aspect is an attribute or component of an entity, e.g., the screen of a cell phone, the service for a restaurant, or the picture quality of a camera. The advantage of feature-based sentiment analysis is the possibility to capture nuances about objects of interest. Different features can generate different sentiment responses, for example a hotel can have a convenient location, but mediocre food. This problem involves several sub-problems, e.g., identifying relevant entities, extracting their features/aspects, and determining whether an opinion expressed on each feature/aspect is positive, negative or neutral. The automatic identification of features can be performed with syntactic methods, with topic modeling, or with deep learning. More detailed discussions about this level of sentiment analysis can be found in Liu's work

2.3 Prior work on Learning Automata based Sentiment Analysis for Recommender System on cloud

A Scalable, Accurate Hybrid Recommender System

Recommender systems apply machine learning techniques for filtering unseen information and can predict whether a user would like a given resource. There are three main types of recommender systems: collaborative filtering, content-based filtering, and demographic recommender systems. Collaborative filtering recommender systems

recommend items by taking into account the taste (in terms of preferences of items) of users, under the assumption that users will be interested in items that users similar to them have rated highly. Content-based filtering recommender systems recommend items based on the textual information of an item, under the assumption that users will like similar items to the ones they liked before. Demographic recommender systems categorize users or items based on their personal attribute and make recommendation based on demographic categorizations. These systems suffer from scalability, data sparsity, and cold-start problems resulting in poor quality recommendations and reduced coverage. In this paper, we propose a unique cascading hybrid recommendation approach by combining the rating, feature, and demographic information about items. We empirically show that our approach outperforms the state of the art recommender system algorithms, and eliminates recorded problems with recommender systems.

Customizing knowledge-based recommender system by tracking analysis of user behavior

In this work, we reviewed the major problems in the existing recommender systems and presented a tracking recommender approach based on user's behavior information and two-level property of items. Our proposed approach defined user profile model, knowledge resources model and constructed Formal Concept Analysis (FCA) mapping to guide a personalized recommendation for user. We simulated a prototype recommender system that can make the quality recommendation by tracking user's behavior. The experimental result showed our strategy was more robust against the drawbacks and preponderant than conventional recommender systems.

Cybernetics and Learning Automata

Stochastic learning automata are probabilistic finite state machines which have been used to model how biological systems can learn. The structure of such a machine can be fixed or can be changing with time. A learning automaton can also be implemented using action (choosing) probability updating rules which may or may not depend on estimates from the environment being investigated. This chapter presents an overview of the field of learning automata, perceived as a completely new paradigm for learning, and explains how it is related to the area of cybernetics.

A multilayer ontology-based hybrid recommendation model

We propose a novel hybrid recommendation model in which user preferences and item features are described in terms of semantic concepts defined in domain ontologies. The concept, item and user spaces are clustered in a coordinated way, and the resulting clusters are used to find similarities among individuals at multiple semantic layers. Such layers correspond to implicit Communities of Interest and enable enhanced recommendations.

FORK: A novel two-pronged strategy for an agent-based intrusion detection scheme in ad-hoc networks

In this work, we introduce FORK, a novel two-pronged strategy to an agent-based intrusion detection system for ad-hoc networks. We follow two different but complementary approaches for intrusion detection in our proposed scheme. We perform intrusion detection for power-aware ad-hoc networks. We introduce a novel power and reputation-based auctioning scheme for distributing agent-tasks in the network. Nodes compete for, and win auctions for performing the tasks based on a competitive power-efficient mechanism that permits collaboration between nodes. The chosen nodes perform the intrusion detection using our proposed anomaly detection algorithm that is modeled on popular evolutionary algorithms techniques. We evaluate our system both in terms of the task allocation algorithm as well as results of actual intrusion detection performed in some session log files. The outcome is promising and offers scope for some interesting additional research. In the next chapter we are going to discuss the System Analysis.

LEARNING AUTOMATA BASED SENTIMENT ANALYSIS FOR RECOMMENDER SYSTEM ON CLOUD

3.1 LEARNING AUTOMATA

Learning automata is one type of Machine Learning algorithm studied since 1970s. Compared to other learning scheme, a branch of the theory of adaptive control is devoted to **Learning Automata** surveyed by Narendra and Thathachar (1974) which were originally described explicitly as finite state automata. Learning automata select their current action based on past experiences from the environment. It will fall into the range of reinforcement learning if the environment is stochastic and Markov Decision Process (MDP) is used.

Research in learning automata can be traced back to the work of Tsetlin in the early 1960s in the Soviet Union. Together with some colleagues, he published a collection of papers on how to use matrices to describe automata functions. Additionally, Tsetlin worked on reasonable and collective automata behaviour, and on automata games. Learning automata were also investigated by researches in the United States in the 1960s. However, the term learning automaton was not used until Narendra and Thathachar introduced it in a survey paper in 1974.

3.1.1 Definition

A learning automaton is an adaptive decision-making unit situated in a random environment that learns the optimal action through repeated interactions with its environment. The actions are chosen according to a specific probability distribution which is updated based on the environment response the automaton obtains by performing a particular action.

With respect to the field of reinforcement learning, learning automata are characterized as policy iterators. In contrast to other reinforcement learners, policy iterators directly manipulate the policy π . Another example for policy iterators are evolutionary algorithms.

Formally, Narendra and Thathachar define a **stochastic automaton** to consist of:

- a set x of possible inputs,
- a set $\Phi = \{ \Phi_1, \dots, \Phi_s \}$ of possible internal states,
- a set $\alpha = \{ \alpha_1, \dots, \alpha_r \}$ of possible outputs, or actions, with $r \leq s$,
- an initial state probability vector $p(0) = \langle p_1(0), \dots, p_s(0) \rangle$,
- a computable function A which after each time step t generates $p(t+1)$ from $p(t)$, the current input, and the current state, and
- a function $G: \Phi \rightarrow \alpha$ which generates the output at each time step.

In this work, they investigate only stochastic automata with $r=s$ and G being bijective, allowing them to confuse actions and states. The states of such an automaton correspond to the states of a "discrete-state discrete-parameter Markov process".^[1] At each time step $t=0,1,2,3,\dots$, the automaton reads an input from its environment, updates $p(t)$ to $p(t+1)$ by A , randomly chooses a successor state according to the probabilities $p(t+1)$ and outputs the corresponding action. The automaton's environment, in turn, reads the action and sends the next input to the automaton. Frequently, the input set $x = \{ 0,1 \}$ is used, with 0 and 1 corresponding to a nonpenalty and a penalty response of the environment, respectively; in this case, the automaton should learn to minimize the number of penalty responses, and the feedback loop of automaton and environment is called a "P-model". More generally, a "Q-model" allows an arbitrary finite input set x , and an "S-model" uses the interval $[0,1]$ of real numbers as x .

3.2 Problem in Learning Automata based Sentiment Analysis

The initial phase of the system, works on identifying the geographic location of the connected nodes, which can include computer terminals or the mobile phones. A request is to be sent to the device to allow identifying its location. On acceptance by the user, the call back response is collected on the server, which includes the latitude and the longitude representing a real-world geographic location. According to geo-location detected, various items (places) are searched in the nearby radius based on the user's query. These results are indexed on the server and are used in the next phase.

This phase includes processing of the data indexed by the server on the user's query request. The comments and the user's feedback are fetched and processed for sentiment analysis for getting a positive or negative classification on it. This is also combined with a confidence level indicated by probability factor between 0 and 1. The score is calculated by turning these feedback responses into numeric values by converting positive to +1 and negative to -1 and multiplying by the calculated probability.

The learning phase bases on the user's previous history, including the feedback and check-ins at various places and comments on it, and, thus, gives a suitable personalized recommender system. This information is collected from various data sets and is processed. Thus, the system also learns by itself and improves the efficiency to provide better recommendations.

Naive Bayes classifiers belong to a family of simple probabilistic classifiers based on applying Bayes theorem with durable (naive) independent presumptions between the features (options). Naive Bayes strategies are a group of supervised learning algorithms consider applying Bayes' theorem (hypothesis) with the "naive" presumption of independence between every pair of choices.

A recursive neural network (RNN) is a sort of deep neural network created by applying the consistent set of weights recursively over a structure, to produce a structured forecast over variable-length input, or a scalar prediction on that, by traversing a given structure in topological arrangement. RNNs are productive in learning sequence and tree structures in natural language processing (NLP), mainly phrase and sentence continuous representations based on word embedding. The organization of the network relies on the structure of the parsed tree for the sentence.

3.3 Design challenges in LA based Sentiment Analysis

Sentiment Analysis is the task of differentiating positive and negative opinions (views), emotions, and interpretations". Sentiment Analysis has many names. It is sometimes observed as subjectiveness analysis, Opinion mining, and appraisal extraction, with some connections to affective computing (computer recognition and expression of emotion).

As a significant part of user interface (UI), sentiment analysis engines are employed across a range of social and review aggregation websites. However, the domain of the applications for Sentiment Analysis reaches a little far from that. It contributes intuition for businesses, giving them immediate remarks on product, and calibrating the impact of their social advertising and marketing approaches. In the same mode, it is extremely applicable in political campaigns, or the alternative platform that concerns public ideas. It even has applications to stock markets and recursive trade dealing engines. It must be stated that adequate sentiment analysis does not just about understand the overall sentiment of a document or a single (one) paragraph. As an instance, in product reviews regularly the author doesn't restrict his view to a solo aspect of the product. The foremost informational and valuable reviews (evaluations) are those that discuss totally different features, and provide an extensive list of benefits and drawbacks. Therefore, it is necessary to be able to extract sentiments on a very granular level, and relate every sentiment to the facet it corresponds to.

Learning Automata (LA) is a self-operating learning model, where “learning” refers to the processing of gaining knowledge during the execution of a simple machine/code (automaton), and to decide on actions to be taken in the future by using the gained knowledge. This model has three main components – the Automaton, the Environment and the Reward/Penalty structure.

3.4 Outline of the Project

We used Automata- Based Sentiment Analysis which recommends the locations local the present location of the users by taking and analyzing the opinions that are taken from the other users who already visited that areas and calculated the ranking. By using LASA we got the reviews about the movies and restaurants. We also got the location based reviews. The reviews we got by using this strategy is either positive or negative. To improve the performance we have used Dual Sentiment Analysis which can further provide positive, negative and neutral comments. Here we have also used deep learning which uses Naïve bayes and Recursive Neural Network classifiers for sentiment classification. We also got the accuracy between the two classifiers. We also got the nearest neighbour count and the retrieval time to get all the reviews by entering our location in terms of of latitude and longitude. We can also assign rating to the movies and restaurants. We can also give feedback (comments) to the movies and restaurants.

SYSTEM ANALYSIS

4.1 Problem Statement

Sentiment Analysis is the task of differentiating positive and negative opinions (views), emotions, and interpretations”. Sentiment Analysis has many names. It is sometimes observed as subjectiveness analysis, Opinion mining, and appraisal extraction, with some connections to affective computing (computer recognition and expression of emotion).

4.2 Proposed Solution

We propose a recommendation system using Learning automata (LA) and sentiment analysis. LA is used to optimize the recommendation score produced by the proposed system using sentiment analysis. The proposed Learning Automata-Based Sentiment Analysis

System (LASA) recommends the places nearby the current location of the users by analyzing the feedback from the places and thus calculating the score based on it.

Advantages

- Recommends the places nearby the current location of the users by analyzing the feedback from the places.
- Easy to identify best places of a particular location.

4.3 Existing System

In the earlier days of Internet, finding places was very difficult. But currently, Recommender systems help in providing users with recommendations about items that people with similar views and preferences have liked in the past. There are two ways in which recommender systems can produce recommendations – collaborative filtering and content-based filtering. Both of these approaches are often combined and are termed as Hybrid Recommender Systems. Collaborative filtering is one of the major approaches used for recommender systems that depend on the opinions and views expressed by other users. For example, Foursquare recommends places based on the previous check-ins of the user. Content-based filtering analyzes a series of different characteristics of an item in order to recommend other items with similar properties.

4.3.1 Disadvantages of Existing system

- Recommendations about items that people with similar views and preferences have liked in the past.
- Recommends places based on the previous check-ins of the user.
- Difficult to identify best places of a particular location.

4.4 System requirement specifications

4.4.1 Hardware specifications

➤ Hard Disk	:	150 GB
➤ Processor	:	3.0 GHz
➤ Ram	:	4 GB

4.4.2 Software specifications

➤ Operating system	:	Windows 7 / Android 4.4
➤ Coding Language	:	JAVA and Azure
➤ Database	:	MYSQL
➤ Server	:	Apache
➤ Cloud	:	Microsoft Azure

4.5 Features of selected software

Cloud computing is the delivery of computing services servers, storage, databases, networking, software, analytics and more over the Internet (“the cloud”).

4.5.1 Azure

Azure is a comprehensive set of cloud services that developers and IT professionals use to build, deploy and manage applications through our global network of datacenters. Integrated tools, DevOps and a marketplace support you in efficiently building anything from simple mobile apps to internet-scale solutions.

Azure is the only consistent hybrid cloud

Build and deploy wherever you want with Azure, the only consistent hybrid cloud on the market. Connect data and apps in the cloud and on-premises for maximum portability and value from your existing investments. Azure offers hybrid consistency in application development, management and security, identity management and across the data platform.

Extend Azure on-premises and build innovative, hybrid apps with Azure Stack. Connect on-premises data and apps to overcome complexity and optimise your existing assets. Distribute and analyse data seamlessly across cloud and on-premises.

Azure is the cloud for building intelligent apps

Use Azure to create data-driven, intelligent apps. From image recognition to bot services, take advantage of Azure data services and artificial intelligence to create new experiences that scale and support deep learning, HPC simulations and real-time analytics on any shape and size of data.

- Develop breakthrough apps with built-in AI.
- Build and deploy custom AI models at scale, on any data.
- Combine the best of Microsoft and open source data and AI innovations.

4.5.2 Java Technology

Java technology is both a programming language and a platform. The Java Programming Language The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

- ♣ Simple
- ♣ Architecture neutral
- ♣ Object oriented
- ♣ Portable
- ♣ Distributed
- ♣ High performance
- ♣ Interpreted

- ♣ Multithreaded
- ♣ Robust
- ♣ Dynamic
- ♣ Secure

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called Java byte codes —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.

You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it's a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make “write once, run anywhere” possible. You can compile your program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation, or on an iMac.

The Java Platform

A platform is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components: • The Java Virtual Machine (Java VM) • The Java Application Programming Interface (Java API) You've already been introduced to the Java VM. It's the base for the Java platform and is ported onto various hardware-based platforms. The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The

Java API is grouped into libraries of related classes and interfaces; these libraries are known as packages. The next section, What Can Java Technology Do? Highlights what functionality some of the packages in the Java API provide. The following figure depicts a program that's running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.

How does the API support all these kinds of programs? It does so with packages of software components that provides a wide range of functionality. Every full implementation of the Java platform gives you the following features:

- The essentials: Objects, strings, threads, numbers, input and output, data structures, system properties, date and time, and so on.
- Applets: The set of conventions used by applets.
- Networking: URLs, TCP (Transmission Control Protocol), UDP (User Datagram Protocol) sockets, and IP (Internet Protocol) addresses.
- Internationalization: Help for writing programs that can be localized for users worldwide. Programs can automatically adapt to specific locales and be displayed in the appropriate language.
- Security: Both low level and high level, including electronic signatures, public and private key management, access control, and certificates.
- Software components: Known as JavaBeans™, can plug into existing component architectures.
- Object serialization: Allows lightweight persistence and communication via Remote Method Invocation (RMI).
- Java Database Connectivity (JDBC™): Provides uniform access to a wide range of relational databases. The Java platform also has APIs for 2D and 3D graphics, accessibility, servers, collaboration, telephony, speech, animation, and more. The following figure depicts what is included in the Java 2 SDK.

ODBC

Microsoft Open Database Connectivity (ODBC) is a standard programming interface for application developers and database systems providers. Before ODBC became a de facto

standard for Windows programs to interface with database systems, programmers had to use proprietary languages for each database they wanted to connect to. Now, ODBC has made the choice of the database system almost irrelevant from a coding perspective, which is as it should be. Application developers have much more important things to worry about than the syntax that is needed to port their program from one database to another when business needs suddenly change. Through the ODBC Administrator in Control Panel, you can specify the particular database that is associated with a data source that an ODBC application program is written to use. Think of an ODBC data source as a door with a name on it. Each door will lead you to a particular database. For example, the data source named Sales Figures might be a SQL Server database, whereas the Accounts Payable data source could refer to an Access database. The physical database referred to by a data source can reside anywhere on the LAN. The ODBC system files are not installed on your system by Windows 95. Rather, they are installed when you setup a separate database application, such as SQL Server Client or Visual Basic 4.0. When the ODBC icon is installed in Control Panel, it uses a file called ODBCINST.DLL. It is also possible to administer your ODBC data sources through a stand-alone program called ODBCADM.EXE. There is a 16-bit and a 32-bit version of this program and each maintains a separate list of ODBC data sources. From a programming perspective, the beauty of ODBC is that the application can be written to use the same set of function calls to interface with any data source, regardless of the database vendor. The source code of the application doesn't change whether it talks to Oracle or SQL Server. We only mention these two as an example. There are ODBC

JDBC

In an effort to set an independent database standard API for Java; Sun Microsystems developed Java Database Connectivity, or JDBC. JDBC offers a generic SQL database access mechanism that provides a consistent interface to a variety of RDBMSs. This consistent interface is achieved through the use of “plug-in” database connectivity modules, or drivers. If a database vendor wishes to have JDBC support, he or she must provide the driver for each platform that the database and Java run on. To gain a wider acceptance of JDBC, Sun based JDBC's framework on ODBC. As you discovered earlier in this chapter, ODBC has widespread support on a variety of platforms. Basing JDBC on ODBC will allow vendors to bring JDBC drivers to market much faster than developing a

completely new connectivity solution. JDBC was announced in March of 1996. It was released for a 90 day public review that ended June 8, 1996. Because of user input, the final JDBC v1.0 specification was released soon after.

4.5.3 MYSQL

MySQL is the most popular Open Source Relational SQL Database Management System. MySQL is one of the best RDBMS being used for developing various web-based software applications. MySQL is developed, marketed and supported by MySQL AB, which is a Swedish company. This tutorial will give you a quick start to MySQL and make you comfortable with MySQL programming.

A database is a separate application that stores a collection of data. Each database has one or more distinct APIs for creating, accessing, managing, searching and replicating the data it holds.

Other kinds of data stores can also be used, such as files on the file system or large hash tables in memory but data fetching and writing would not be so fast and easy with those type of systems.

Nowadays, we use relational database management systems (RDBMS) to store and manage huge volume of data. This is called relational database because all the data is stored into different tables and relations are established using primary keys or other keys known as **Foreign Keys**.

A **Relational DataBase Management System (RDBMS)** is a software that

- Enables you to implement a database with tables, columns and indexes.
- Guarantees the Referential Integrity between rows of various tables.
- Updates the indexes automatically.
- Interprets an SQL query and combines information from various tables.

4.6 Feasibility Study

Feasibility study is conducted once the problem is clearly understood. The feasibility study which is a high level capsule version of the entire system analysis and design process. The objective is to determine whether the proposed system is feasible or not and it helps us to the minimum expense of how to solve the problem and to determine,

if the Problem is worth solving. The following are the three important tests that have been carried out for feasibility study.

4.6.1 Technical feasibility

In the technical feasibility study, one has to test whether the proposed system can be developed using existing technology or not. It is planned to implement the proposed system in PHP and Cloud Environment. The project entitled is technically feasible because of the following reasons.

- All necessary technology exists to develop the system.
- The existing system is so flexible that it can be developed further.

4.6.2 Economic feasibility

As a part of this, the costs and benefits associated with the proposed systems are to be compared. The project is economically feasible only if tangible and intangible benefits outweigh the cost. We can say the proposed system is feasible based on the following grounds.

But in order to business and research oriented we deployed in Microsoft Azure Virtual Machines its cost is moderated and high elasticity in nature depends on users load cloud mainly reduces operational and maintenance cost.

4.6.3 Operational feasibility

The project is operationally feasible because there is sufficient support from the project management and the users of the proposed system. Proposed system definitely does not harm and will not produce the bad results and no problem will arise after implementation of the system.

User-friendly

Customer will use the forms for their various transactions i.e. for adding new routes, viewing the routes details. Also the Customer wants the reports to view the various transactions based on the constraints. These forms and reports are generated as user-friendly to the Client.

Reliability

The package will pick-up current transactions on line. Regarding the old transactions, User will enter them in to the system.

Security

The web server and database server should be protected from hacking, virus etc.

Availability

This software will be available always.

In next chapter we are going to discuss system design and implementation.

SYSTEM DESIGN AND IMPLEMENTATION

5.1 System Design

5.1.1 Block Diagram

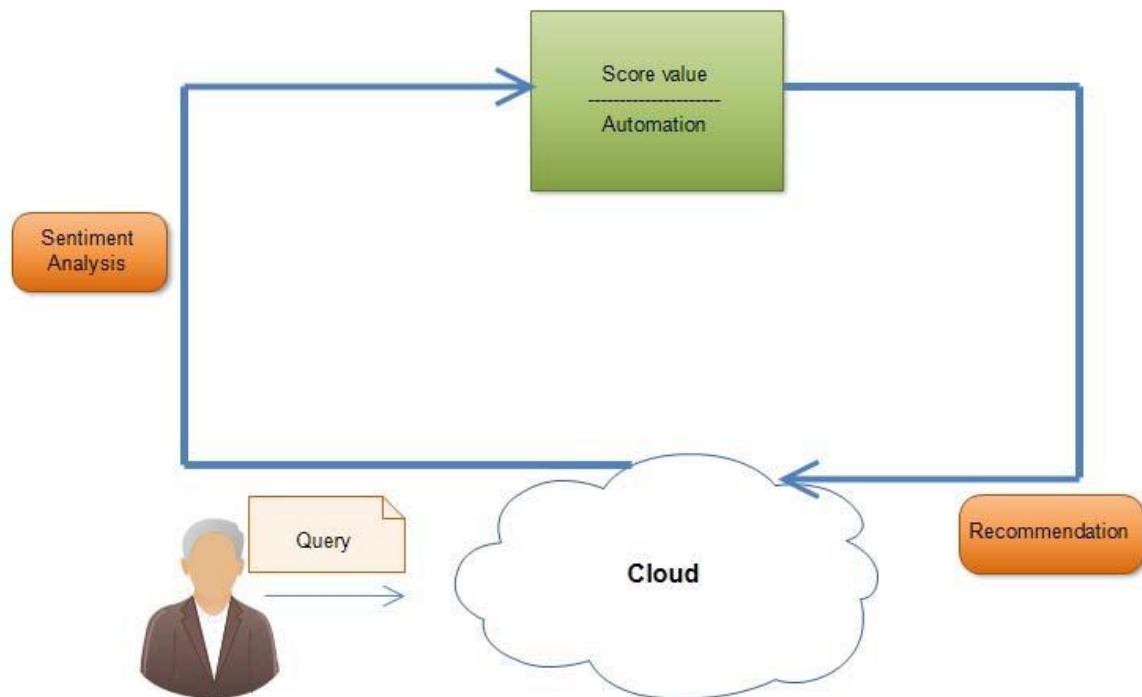


Figure 5.1.1 Interactions between the cloud and the learning system

5.1.2 Input and Output design

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?

- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

OBJECTIVES

The system model for the recommendation system includes the Cloud Architecture with all the nodes connected to it. We use cloud platform provided by phpfogapp, which is a platform-as-a-service hosted on Amazon Web Services. The end-users of the system are referred to as different nodes around the system. The users (nodes) request the system to suggest the best place in reference of a particular item and this information is passed on to the central system which is held responsible for performing all operations and responding back to the user. The fundamental idea behind the proposed system is to collect information provided by the nodes and then applying sentiment analysis on the information, we extract the sentiment of the comment, either a positive or a negative response. Then, based on the previous experiences of the user and LA, we build score analysis and predict the information in reference to the type of the query asked by the user. Initially, the user is prompted to give input of the type of query and detect the geo-location of the node (user) with user permission and formulate into latitude and longitude information.

The central system is then queried with the item name and the extracted geo-location parameters. If the response is null, then it means that there is no place related to the item nearby the present location and the user is asked to repeat the process with a new query.

If the response is not null, the system finds out all places in a finite radius of geo-location from the resulted data-sets and stores them. Then, iterating over all the information that is being stored, the comments and the specific tags are fetched from it. Sentiment analysis is performed on the comments and the extracted tags and the response is recorded either as positive or negative. If the response is positive, then score is incremented by a unit factor, otherwise, if the response is negative, the score is decremented by the same factor.

5.1.3 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

Use case diagram:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

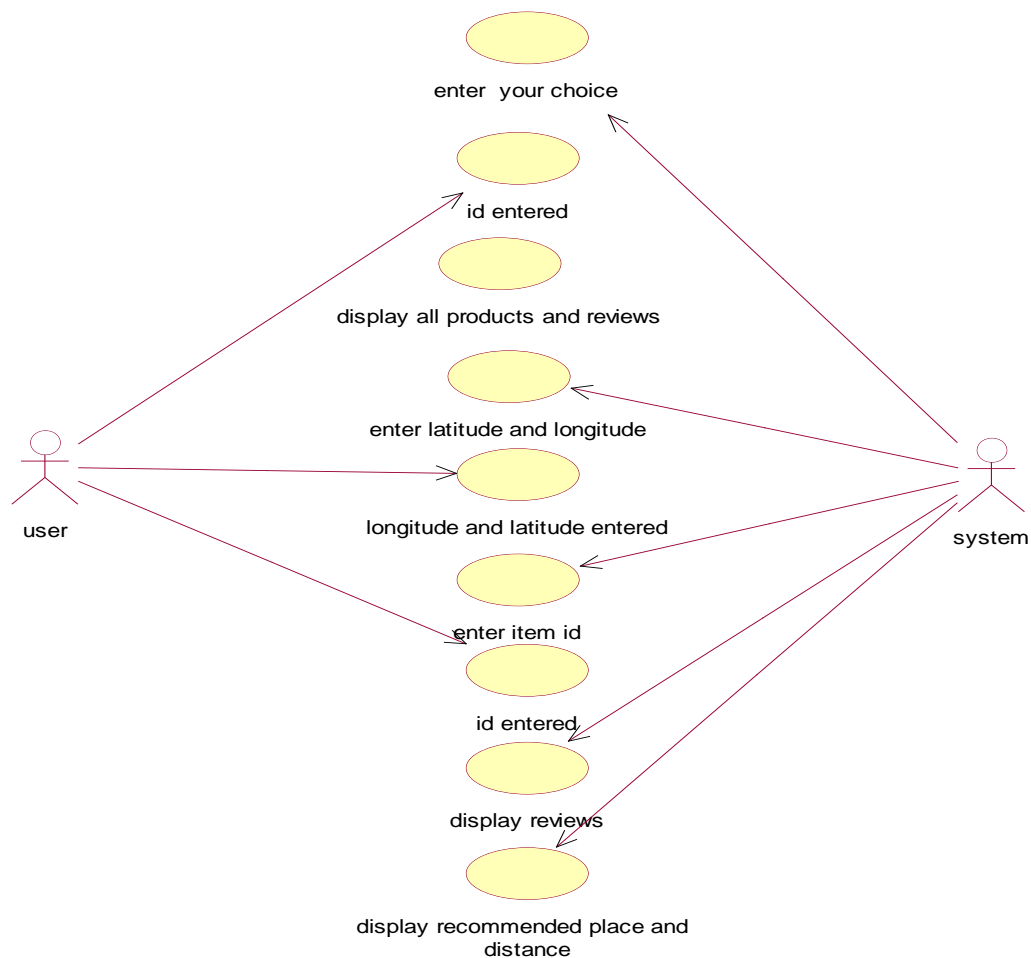


Figure 5.1.3.1: Use Case

Class diagram:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

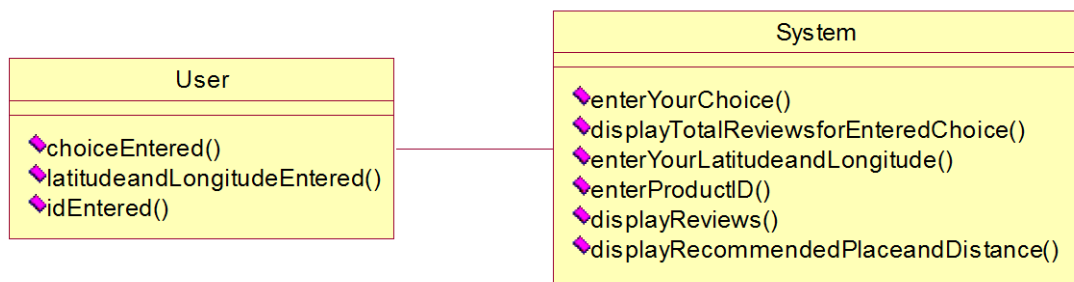


Figure 5.1.3.2: Class Diagram

Sequence diagram:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



Figure 5.1.3.3: Sequence diagram

Collaboration diagram:

A collaboration diagram, also called a communication diagram or interaction diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML). The concept is more than a decade old although it has been refined as modeling paradigms have evolved.

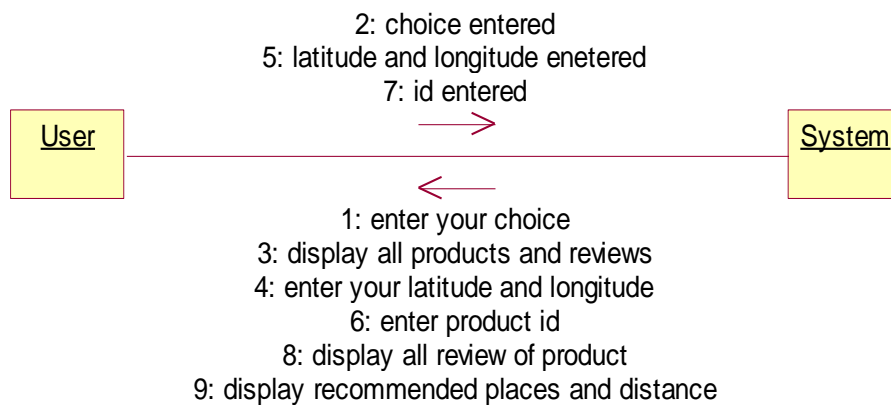


Figure 5.1.3.4: Collaboration diagram

Activity diagram:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

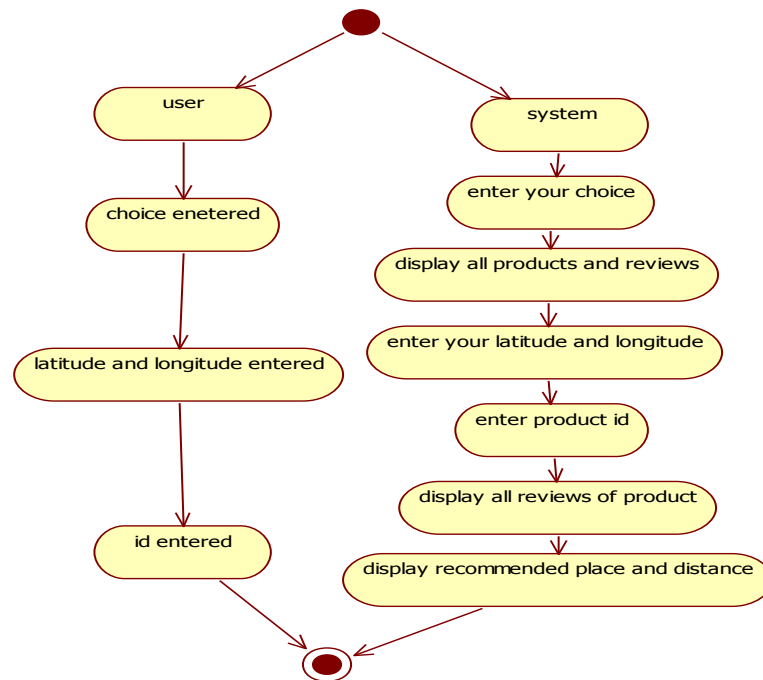


Figure 5.1.3.5: Activity Diagram

Component diagram:

A component diagram describes the organization and wiring of the physical components in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required functions is covered by planned development.



Figure 5.1.3.6: Component Diagram

Deployment Diagram:

A deployment diagram in the Unified Modeling Language models the physical deployment of artifacts on nodes. To describe a web site, for example, a deployment diagram would show what hardware components ("nodes") exist (e.g., a web server, an application server, and a database server), what software components ("artifacts") run on

each node (e.g., web application, database), and how the different pieces are connected (e.g. JDBC, REST, RMI).

The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have subnodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.

There are two types of Nodes:

1. Device Node
2. Execution Environment Node

Device nodes are physical computing resources with processing memory and services to execute software, such as typical computers or mobile phones. An execution environment node (EEN) is a software computing resource that runs within an outer node and which itself provides a service to host and execute other executable software elements.

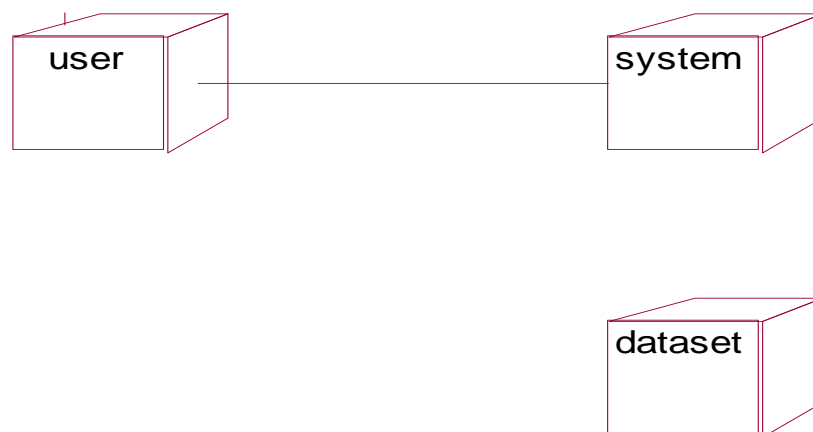


Figure 5.1.3.7: Deployment Diagram

5.2 Implementation

MODULES

- Naive Bayes methods
- Recursive neural network

5.2.1 MODULES DESCRIPTION:

5.2.1.1 Naive Bayes methods

Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of independence between every pair of features. Given a class variable y and a dependent feature vector x_1 through x_n , Bayes' theorem states the following relationship:

$$P(y | x_1, \dots, x_n) = \frac{P(y)P(x_1, \dots, x_n | y)}{P(x_1, \dots, x_n)}$$

Using the naive independence assumption that

$$P(x_i | y, x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n) = P(x_i | y),$$

for all i , this relationship is simplified to

$$P(y | x_1, \dots, x_n) = \frac{P(y) \prod_{i=1}^n P(x_i | y)}{P(x_1, \dots, x_n)}$$

Since $P(x_1, \dots, x_n)$ is constant given the input, we can use the following classification rule:

$$\begin{aligned} P(y | x_1, \dots, x_n) &\propto P(y) \prod_{i=1}^n P(x_i | y) \\ &\Downarrow \\ \hat{y} &= \arg \max_y P(y) \prod_{i=1}^n P(x_i | y), \end{aligned}$$

and we can use Maximum A Posteriori (MAP) estimation to estimate $P(y)$ and $P(x_i | y)$; the former is then the relative frequency of class y in the training set.

The different naive Bayes classifiers differ mainly by the assumptions they make regarding the distribution of $P(x_i | y)$.

In spite of their apparently over-simplified assumptions, naive Bayes classifiers have worked quite well in many real-world situations, famously document classification and spam filtering. They require a small amount of training data to estimate the necessary parameters. (For theoretical reasons why naive Bayes works well, and on which types of data it does, see the references below.)

Naive Bayes learners and classifiers can be extremely fast compared to more sophisticated methods. The decoupling of the class conditional feature distributions means that each distribution can be independently estimated as a one dimensional distribution. This in turn helps to alleviate problems stemming from the curse of dimensionality.

On the flip side, although naive Bayes is known as a decent classifier, it is known to be a bad estimator, so the probability outputs from `predict_proba` are not to be taken too seriously.

Naïve Bayes classification of positive and negative:

```
public void learn(K category, Collection<T> features) {
    this.learn(new Classification<T, K>(features, category));
}

*/

public void learn(Classification<T, K> classification) {

    for (T feature : classification.getFeatureset())
        this.incrementFeature(feature, classification.getCategory());
    this.incrementCategory(classification.getCategory());
    this.memoryQueue.offer(classification);
    if (this.memoryQueue.size() > this.memoryCapacity) {
        Classification<T, K> toForget = this.memoryQueue.remove();
        for (T feature : toForget.getFeatureset())
            this.decrementFeature(feature, toForget.getCategory());
        this.decrementCategory(toForget.getCategory());
    }
}
```

5.2.1.2 Recursive Neural Network

A recursive neural network (RNN) is a kind of deep neural network created by applying the same set of weights recursively over a structure, to produce a structured prediction over variable-length input, or a scalar prediction on it, by traversing a given structure in topological order. RNNs have been successful in learning sequence and tree structures in natural language processing, mainly phrase and sentence continuous representations based on word embedding.

Recursive Neural Network:

```
public static void init(){
    if(pipeline == null){
        Properties props = new Properties();
        props.setProperty("annotators", "tokenize, ssplit, pos, lemma, ner, parse,
dcoref, sentiment");
        pipeline = new StanfordCoreNLP(props);
    }
}

public static int findSentiment(String tweet) {
    int mainSentiment = 0;
    if(tweet != null && tweet.length() > 0){
        int longest = 0;
        Annotation annotation = pipeline.process(tweet);
        for(CoreMap sentence : annotation.get(CoreAnnotations.SentencesAnnotation.class)){
            Tree tree = sentence.get(SentimentCoreAnnotations.SentimentAnnotatedTree.class);
            int sentiment = RNNCoreAnnotations.getPredictedClass(tree);
            String partText = sentence.toString();
            if(partText.length() > longest){
                mainSentiment = sentiment;
                longest = partText.length();
            }
        }
        return mainSentiment;
    }
}
```

In spite of their apparently over-simplified assumptions, naive Bayes' classifiers have worked quite well in many real-world situations, famously document classification and spam filtering. They require a small amount of training data to estimate the necessary parameters.

Here the code for implementation of user interaction.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<meta name="keywords" content="free design template, CSS template, HTML website"
/>

<meta name="description" content="Free Design Template, Free CSS Website, XHTML
CSS layout" />

<link href="templatemo_style.css" rel="stylesheet" type="text/css" />

</head>

<body>

<!-- Download Free CSS Templates from www.templatemo.com -->

<div id="templatemo_header_panel">

    <div id="templatemo_header_section">

        <div id="tagline"></div>

    </div>

</div> <!-- end of haeder -->
```



```

<div id="templatemo_menu_panel">

    <div id="templatemo_menu_section">

        <ul>

            <li><a href="index.jsp">Home</a></li>

                <li><a href="Login.jsp">Existing&nbsp;User</a></li>

                <li><a href="Register.jsp" class="last">New&nbsp;User</a></li>

        </ul>

    </div>

</div> <!-- end of menu -->

<br/>

<table align="center" width="80%">

    <tr>

        <td>

            <p align="justify"><font size="3" color="black" style="font-family: Comic Sans MS">

                <center>    Learning Automata Based Sentiment Analysis for

Recommender System on Cloud</center>

            </p>

        </td>

    </tr>

</table>

</body>

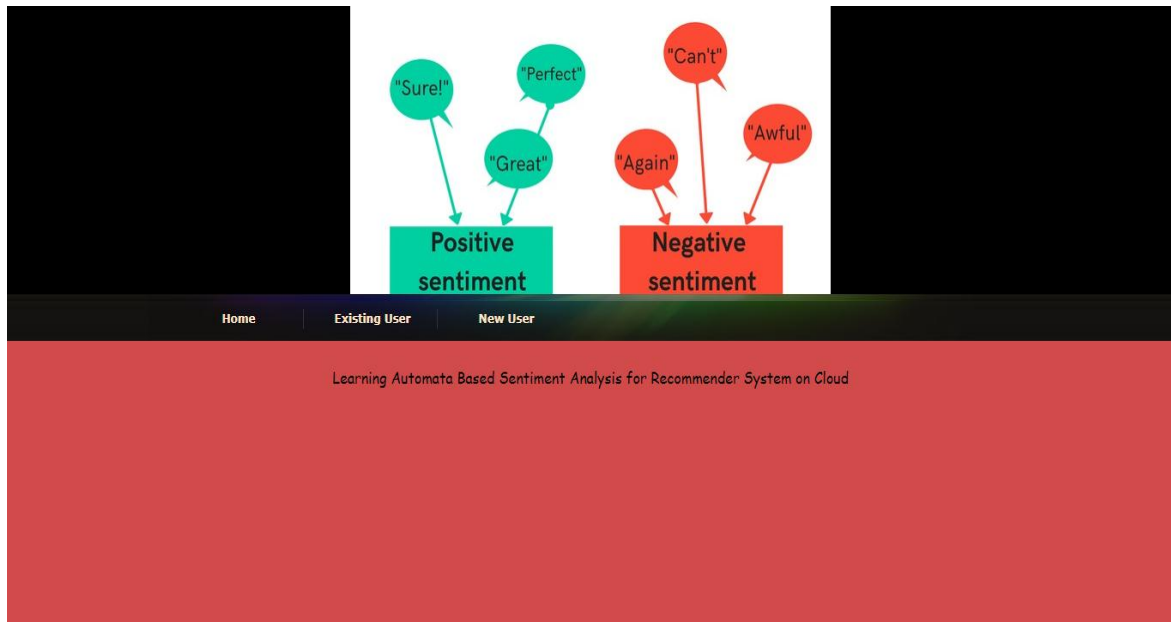
</html>

```

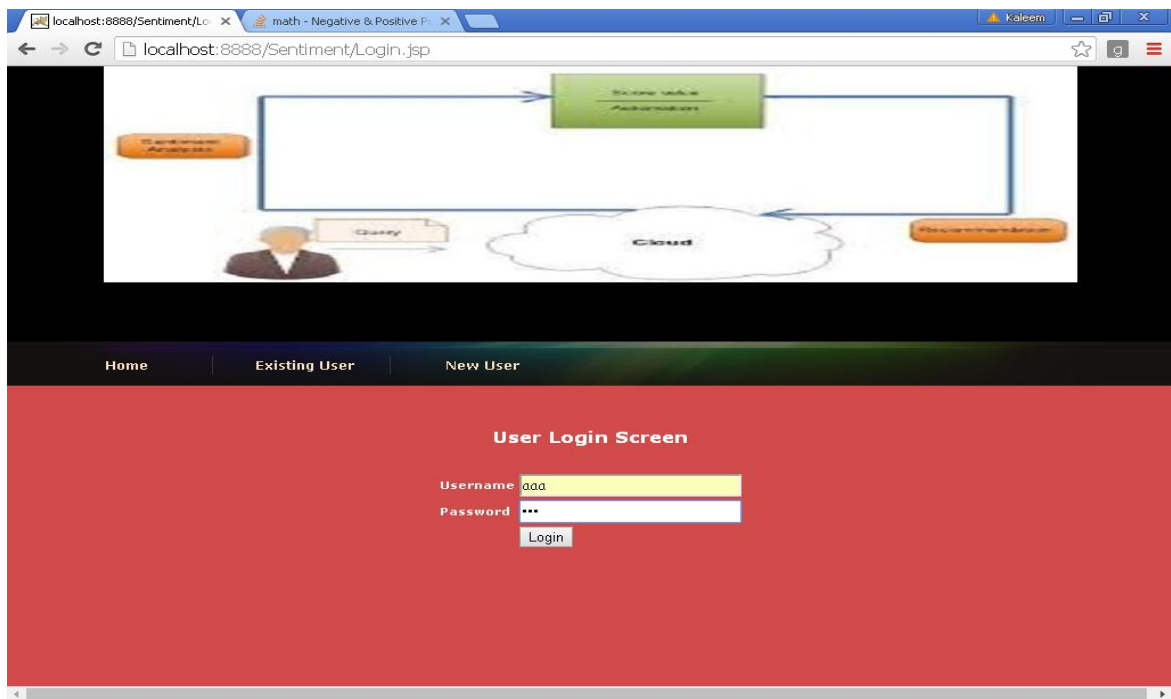
EXPERIMENTAL RESULTS

6.1 Output Screenshots

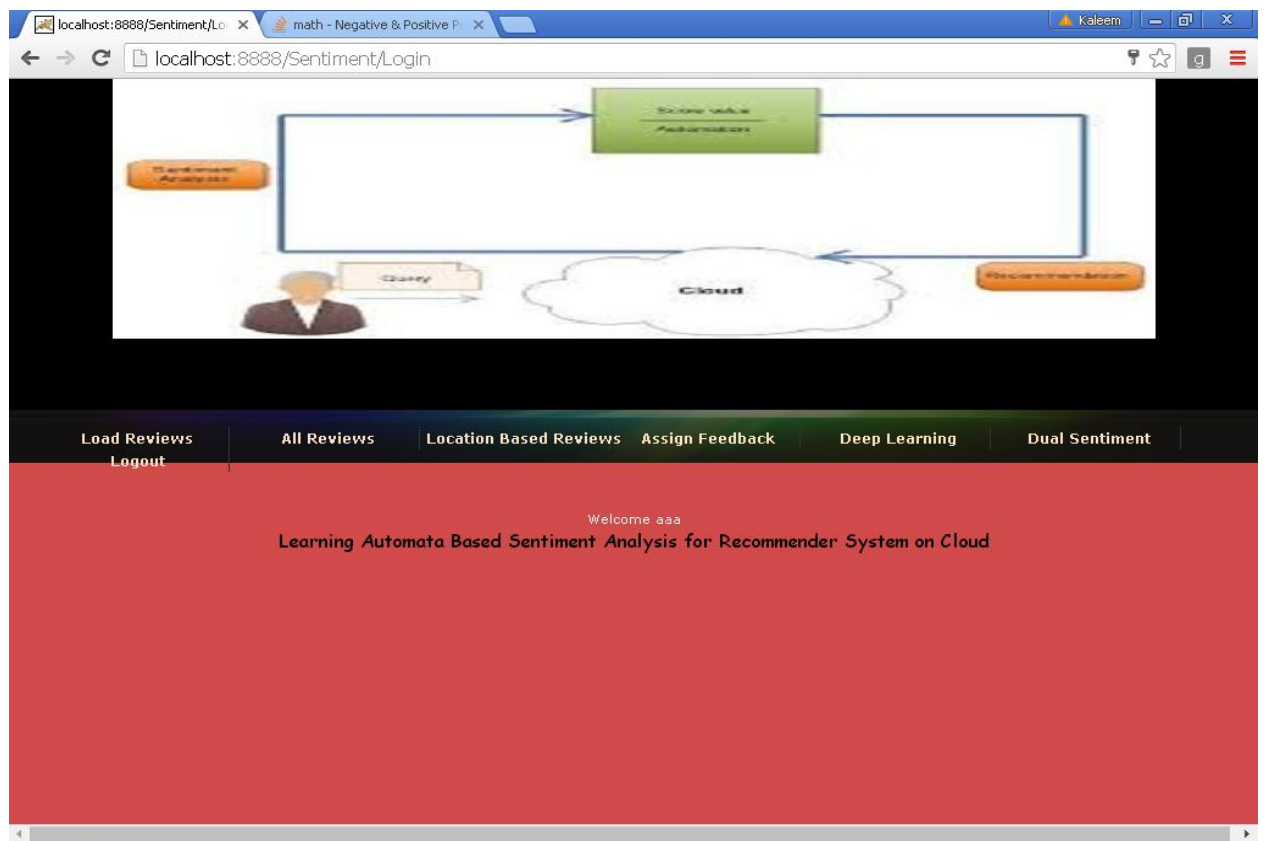
6.1.1 Home page



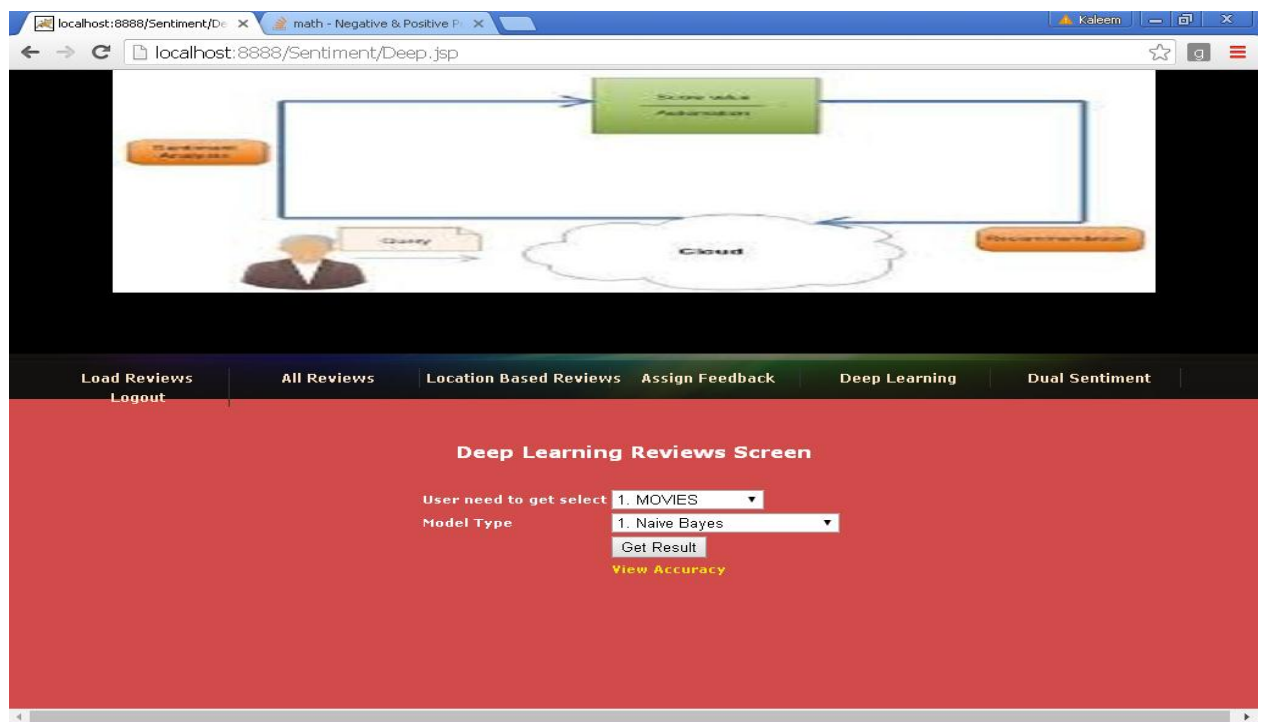
6.1.2 Index Page:



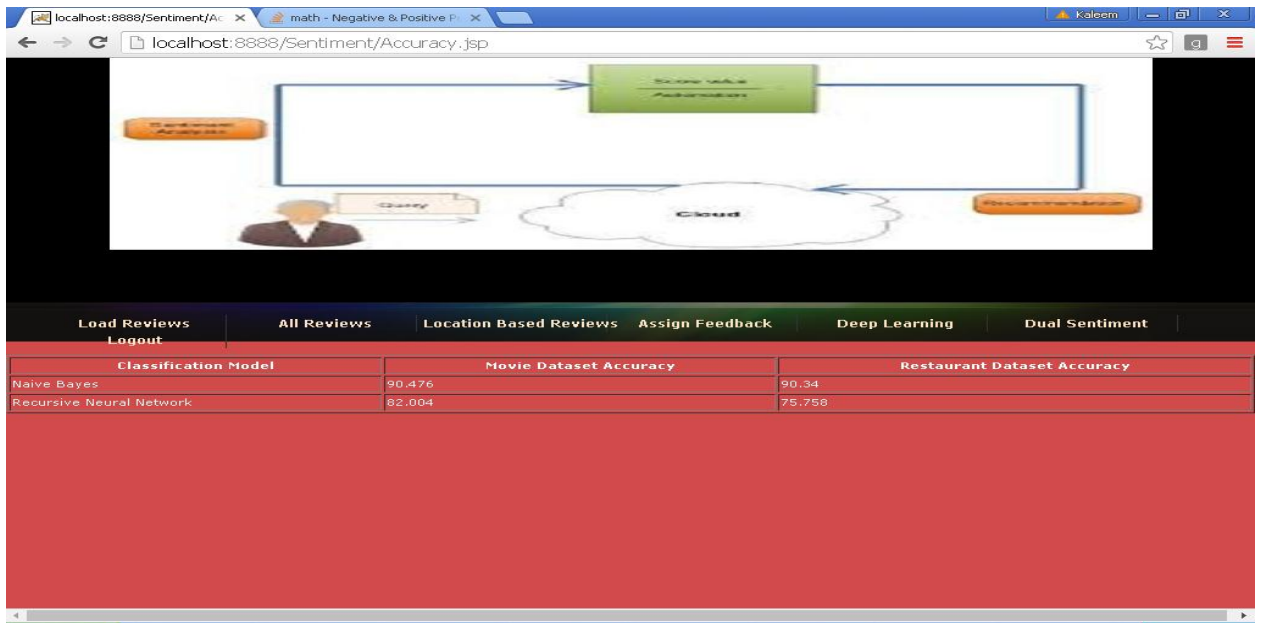
6.1.3 Load Reviews:



6.1.4 All Reviews:

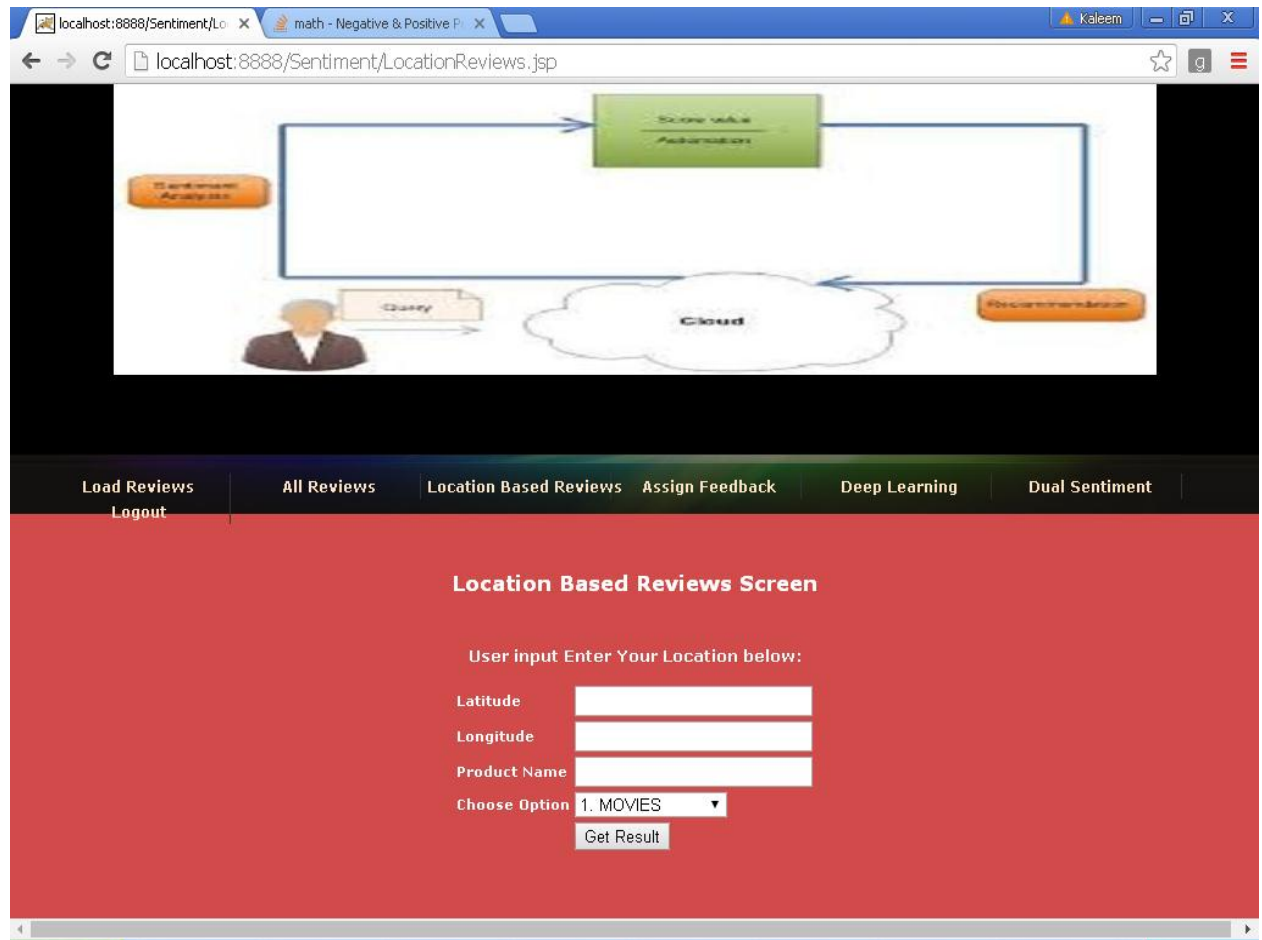


6.1.5 Go to “Location Based Reviews” Link:



Classification Model	Movie Dataset Accuracy	Restaurant Dataset Accuracy
Naive Bayes	90.476	90.34
Recursive Neural Network	82.004	75.758

6.1.6 Enter latitude, longitude and product name



Location Based Reviews Screen

User input Enter Your Location below:

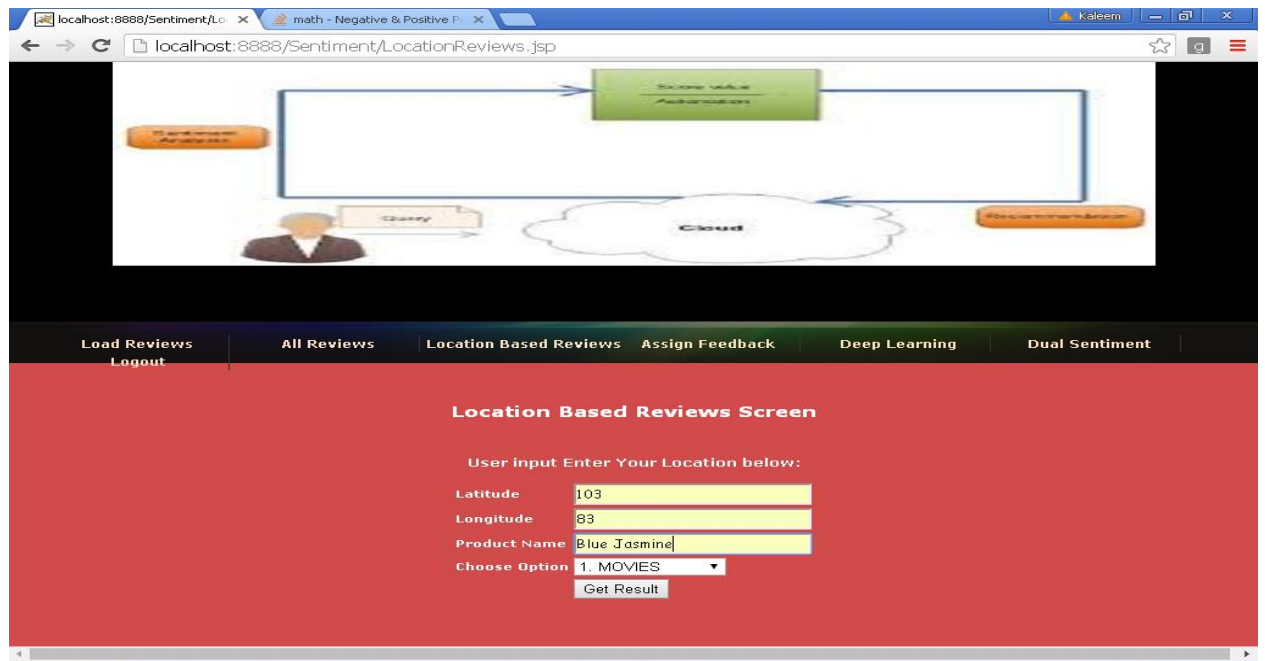
Latitude

Longitude

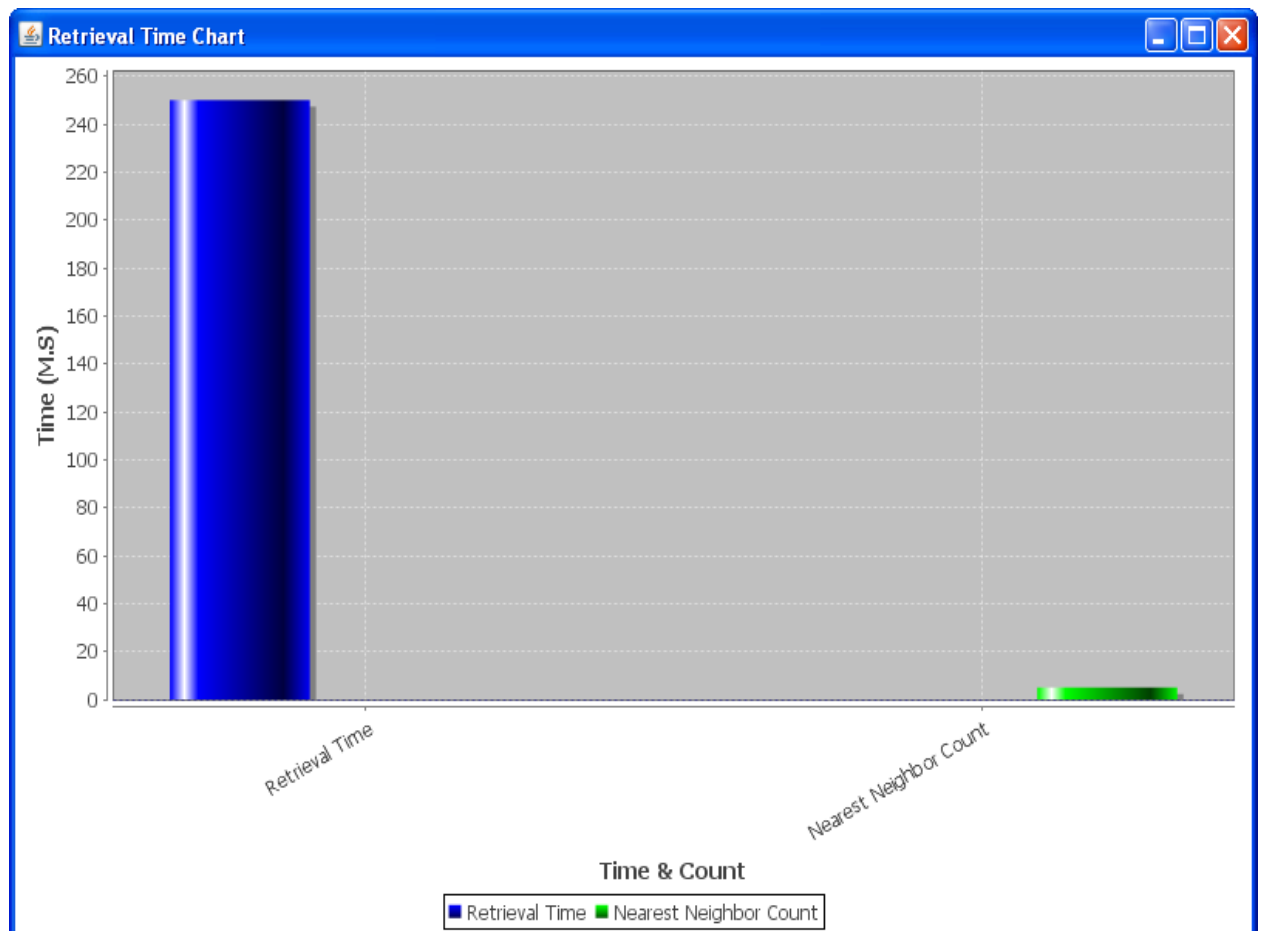
Product Name

Choose Option 1. MOVIES ▼

6.1.7 Graphical result for above query



6.1.8 Go to 'Dual Sentiment' Link to get below screen



6.1.9 Go to 'All Review' Link

localhost:8888/Sentiment/Dual.jsp

Score via Activation

Cloud

Load Reviews Logout | All Reviews | Location Based Reviews | Assign Feedback | Deep Learning | Dual Sentiment

Dual Learning Reviews Screen

User need to get select 1. MOVIES

Get Result

ID	Product ID	Product Name	Summary	Result
3	B00006HAXW	Blue Ruin	a rock n roll history lesson	Positive
3	B00006HAXW	Blue Ruin	a musthave video if you grew up in the s or s	Neutral
3	B00006HAXW	Blue Ruin	if you like doowop you gotta have this dvd	Neutral
3	B00006HAXW	Blue Ruin	i expected more	Neutral
3	B00006HAXW	Blue Ruin	professional excellence	Neutral
3	B00006HAXW	Blue Ruin	marvelous just marvelous	Positive
3	B00006HAXW	Blue Ruin	pittsburgh home of the oldies	Neutral
3	B00006HAXW	Blue Ruin	they sang in the subway in halls	Neutral
2	B00006HAXW	Blue Ruin	doowop recorded history a must have item	Neutral
6	B00006HAXW	Blue Ruin	rock rythm and doowop	Neutral
7	B00006HAXW	Blue Ruin	unbelievable best concert	Positive
3	B00006HAXW	Blue Ruin	another outstanding performance and concert	Positive
3	B00006HAXW	Blue Ruin	outstanding whether it is either on vhs or dvd	Positive
3	B00004CQT3	The Book Thief	far from home the adventures of yellow dog	Neutral
4	B00004CQT3	The Book Thief	spectacular and suspenseful family film	Positive
4	B00004CQT3	The Book Thief	a truly wonderful family film	Positive
4	B00004CQT3	The Book Thief	excellent family movie	Positive
4	B00004CQT3	The Book Thief	awesome dog movie	Positive
4	B00004CQT3	The Book Thief	a movie for all ages	Positive
5	B00004CQT3	The Book Thief	great american movie	Positive
5	B00004CQT3	The Book Thief	uh oh	Neutral
5	B00004CQT3	The Book Thief	amazing	Positive

Waiting for localhost...

6.2 Performance Metric

This process includes analyzing the user's previous checkins on the places from the resultant data sets. If the user has already checked-in more than once and provided positive feedback then the reward probability will be added to the score. However, if the user has checked-in before, and has given the negative response then the penalty probability will be added to the score. Thus, the score will get more and more efficient based on the LA, which will provide recommendation based on the personal experience.

TABLE I EXPERIMENTAL SETTINGS/TOOLS

Cloud Service	Phpfog-Amazon Web Services
Data Sets	Foursquare
Network Analyzer Tool	Pingdom
Sentiment Analysis classifier	Viralheat
Visualization	HighCharts

To test the proposed system, we used foursquare, social networking site data-sets which include user's (nodes) checkins at the different places with their feedback on the same. In the system setup, the cloud service is provided by 'phpfog' which is the platform-as-a-service hosted on Amazon Web Services. Table 1 describes the experimental settings/tools that are used in testing the proposed framework. We evaluated the performance of LASA and compared our results with sentiment analysis. During the experiment, we have taken the real time data from various places all over the country with the help of *foursquare APIs* and using Network Analyzer Tool, we tested the response time and other factors. The Viralheat API's used in the system helps in getting the sentiment value of the user's response.