iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 3 **Issue:** VI **Month of publication:** June 2015

DOI:

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)

Ranking Approaches Based On Improving Aggregate Recommendation Diversity Algorithms

L.Pravin Kumar[1], K. Ramesh[2]
[1]U.G Scholar- Department of Computer Science and Engineering, Sri Venkateswara College of Engineering, Chennai, India.
[2] Ph.D Scholar - Department of Electronics and Communication Engineering, Easwari Engineering College, Chennai, India.

Abstract--Recommender systems are becoming increasingly important to individual users and businesses for providing personalized recommendations. However, while the majority of algorithms proposed in recommender systems literature has focused on improving recommendation accuracy, other important aspects of recommending quality, such as the diversity of recommendations, have often been overlooked. In this paper, we introduce and explore a number of item ranking approaches that can generate recommendations that have substantially higher aggregate diversity across all users while maintaining comparable levels of recommendation accuracy. Comprehensive empirical evaluation consistently shows the diversity gains of the proposed approaches using several real-world rating data sets and different rating prediction algorithms.

I. INTRODUCTION

In the current age of information overload, it is becoming increasingly harder to find relevant content. This problem is not only widespread but also alarming. Over the last 10- 15 years, recommender systems technologies have been introduced to help people deal with these vast amounts of information, and they have been widely used in research as well as e-commerce applications, such as the ones used by Amazon and Netflix.The most common formulation of the recommendation problem relies on the notion of ratings, i.e., recommender systems estimate ratings of items (or products) that are yet to be consumed by users, based on the ratings of items already consumed. Recommender systems typically try to predict the ratings of unknown items for each user, often using other users' ratings, and recommend top N items with the highest predicted ratings. Accordingly, there have been many studies on developing new algorithms that can enhance the predictive accuracy of the recommendations. While, the quality of recommendations can be estimated along a number of dimensions, and relying on the accuracy of recommendations alone may not be enough to find the most relevant items for each user. In particular, the importance of diverse recommendations has been previously emphasized in several studies. These studies argue that one of the goals of recommender systems is to provide a user with highly idiosyncratic or personalized items, and more diverse recommendations result in more opportunities for users to get recommended such items. With this motivation, some studies proposed new recommendation methods.That can increase the diversity of recommendation sets for a given individual user, often measured by an average dissimilarity between all pairs of recommended items, while maintaining an acceptable level of accuracy.

These studies measure recommendation diversity from an individual user's perspective (i.e., individual diversity). In contrast to individual diversity, which has been explored in a number of papers, some recent studies started examining the impact of recommender systems on sales diversity by considering the aggregate diversity of recommendations across all users. Note that high individual diversity of recommendations does not necessarily imply high aggregate diversity. For example, if the system recommends to all users the same five best-selling items that are not similar to each other, the recommendation list for each user is diverse (i.e., high individual diversity), but only five distinct items are recommended to all users and purchased by them (i.e., resulting in low aggregate diversity or high sales concentration). Before developing the tool it is necessary to determine the time factor, economy n company strength. Once these things r satisfied, ten next steps 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 taken into account for developing the proposed system.

A. Security A Major Concern

Security concerns arising because both customer data and program are residing in Provider Premises. Security is always a major concern in Open System architecture.

©IJRASET 2015: All Rights are Reserved

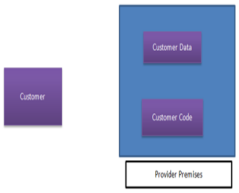


Fig 1. Data centre Security

Professional Security staff utilizing video surveillance, state of the art intrusion detection systems, and other electronic means. When an employee no longer has a business need to access datacenter his privileges to access datacentre should be immediately revoked. All electronic and physical access to data canters by employees should be logged and audited routinely. Audit tools so that users can easily determine how their data is stored, protected, used, and verify policy enforcement.. Data Location when user uses the cloud, user probably won't know exactly where your data is hosted, what country it will be stored in. Data should be stored and processed only in specific jurisdictions as define by user. Provider should also make a contractual commitment to obey local privacy requirements on behalf of their customers, Data-cantered policies that are generated when a user provides personal or sensitive information, which travels with that information throughout its lifetime to ensure that the information is used only in accordance with the policy.

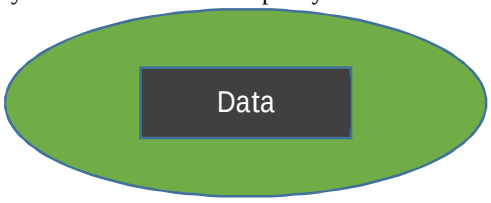


Fig 2.Backups of Data

Data store in database of provider should be redundantly store in multiple physical locations. Data that is generated during running of program on instances is all customer data and therefore provider should not perform backups.Control of administrator on databases.

1) Data Sanitization: Sanitization is the process of removing sensitive information from a storage device. What happens to data stored in a cloud computing environment once it has passed its user's "use by date" What data sanitization practices does the cloud computing service provider propose to implement for redundant and retiring data storage devices as and when these devices are retired or taken out of service.

2) Network Security: Denial of Service: where servers and networks are brought down by a huge amount of network traffic and users are denied the access to a certain Internet based service. Like DNS Hacking, Routing Table "Poisoning", XDoS attacks. QoS Violation: through congestion, delaying or dropping packets, or through resource hacking. Man in the Middle Attack: To overcome it always use SSL. IP Spoofing: Spoofing is the creation of TCP/IP packets using somebody else's IP address.
Solution: Infrastructure will not permit an instance to send traffic with a source IP or MAC address other than its own.

3) Information Security: Security related to the information exchanged between different hosts or between hosts and users. This issues pertaining to secure communication, authentication, and issues concerning single sign on and delegation. Secure communication issues include those security concerns that arise during the communication between two entities. These include confidentiality and integrity issues. Confidentiality indicates that all data sent by users should be accessible to only "legitimate" receivers, and integrity indicates that all data received should only be sent/modified by "legitimate" senders. Solution: public key encryption, X.509 certificates, and the Secure Sockets Layer (SSL) enables secure authentication and communication over computer networks.

B. System Study

1) Feasibility Study: The feasibility of the scheme is analyzed in this phase and business proposal is put forth with a very general plan for the scheme and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential. Three key considerations involved in the feasibility analysis are

i) Economic Feasibility: This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures

©IJRASET 2015: All Rights are Reserved

must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

ii) Technical Feasibility: This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

iii) Social Feasibility: The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

C. System Analysis

1) Problem Formulation: There is a growing awareness of the importance of aggregate diversity in recommender systems. Furthermore, while, as mentioned earlier, there has been a significant amount of work done on improving individual diversity, the issue of aggregate diversity in recommender systems has been largely untouched. It is becoming increasingly harder to find relevant content. This problem is not only widespread but also alarming.

II. PROPOSED SYSTEM

The objective of the scheme is to develop a system that automates the processes and activities of a travel and. The purpose is to design a system using which one can perform all operations related to traveling. In real world settings, recommender systems generally perform the following two tasks in order to provide recommendations to each user. First, the ratings of unrated items are estimated based on the available information (typically using known user ratings and possibly also information about item content or user demographics) using some recommendation algorithm. And second, the system finds items that maximize the user's utility based on the predicted ratings, and recommends them to the user. Ranking approaches proposed in this paper are designed to improve the recommendation diversity in the second task of finding the best items for each user. In particular, these approaches are extremely efficient, because they are based on scalable sorting-based heuristics that make decisions based only on the local data (i.e., only on the candidate items of each individual user) without having to keep track of the global information, such as which items have been recommended across all users and how many times.

System architecture

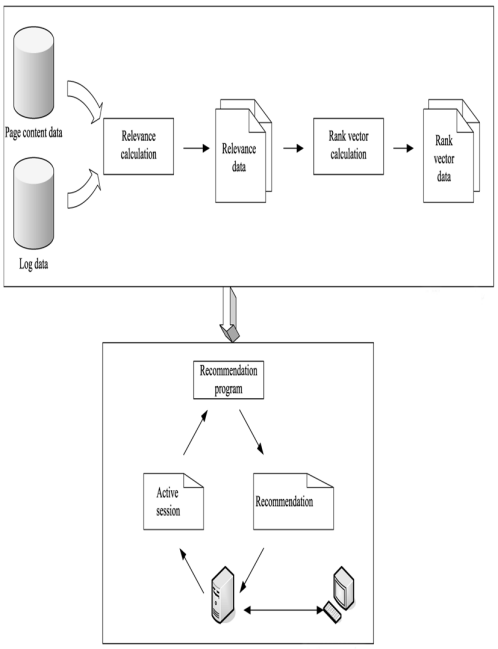


Fig 3. Recommendation algorithm

©IJRASET 2015: All Rights are Reserved

There exist multiple variations of neighbourhood-based CF approaches. In this paper, to estimate $R^*(u, i)$, i.e., the rating that user u would give to item i, we first compute the similarity between user u and other users u' using a cosine similarity metric. Where $I(u, u')$ represents the set of all items rated by both user u and user u'. Based on the similarity calculation, set $N(u)$ of nearest neighbours of user u is obtained. The size of set $N(u)$ can range anywhere from 1 to $|U|$-1, i.e., all other users in the dataset.

Then, $R^*(u, i)$ is calculated as the adjusted weighted sum of all known ratings $R(u', i)$ Here $R(u)$ represents the average rating of user u. A neighbourhood-based CF technique can be user-based or item-based, depending on whether the similarity is calculated between users or items, the user-based approach, but they can be straightforwardly rewritten for the item-based approach because of the symmetry between users and items in all neighbourhood-based CF calculations. In our experiments we used both user-based and item-based approaches for rating estimation.

A. *Posting the opinion*

In this module, we get the opinions from various people about business, e-commerce and products through online. The opinions may be of two types. Direct opinion and comparative opinion. Direct opinion is to post a comment about the components and attributes of products directly. Comparative opinion is to post a comment based on comparison of two or more products. The comments may be positive or negative.

B. *Recommendation technique*

However, the quality of recommendations can be evaluated along a number of dimensions, and relying on the accuracy of recommendations alone may not be enough to find the most relevant items for each User, these studies argue that one of the goals of recommender systems is to provide a user with highly personalized items, and more diverse recommendations result in more opportunities for users to get recommended such items.

With this motivation, some studies proposed new recommendation methods that can increase the diversity of recommendation sets for a given *individual* user. They can give the feedback of such items.

C. *Rating prediction*

First, the ratings of unrated items are estimated based on the available information (typically using known user ratings and possibly also information about item content) using some recommendation algorithm. Heuristic approaches typically calculate recommendations based directly on the previous user activities (e.g., transactional data or rating values). For each user, ranks all the predicted items according to the predicted rating value ranking the candidate (highly predicted) items based on their predicted rating value, from lowest to highest (as a result choosing less popular items.

D. *Ranking approach*

Ranking items according to the rating variance of neighbours of a particular user for a particular item. There exist a number of different ranking approaches that can improve recommendation diversity by recommending items other than the ones with topmost predicted rating values to a user. A comprehensive set of experiments was performed using every rating prediction technique in conjunction with every recommendation ranking function on every dataset for different number of top-N recommendations.

DATAFLOW DIAGRAM

©IJRASET 2015: All Rights are Reserved

Volume 3 Issue VI, June 2015
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

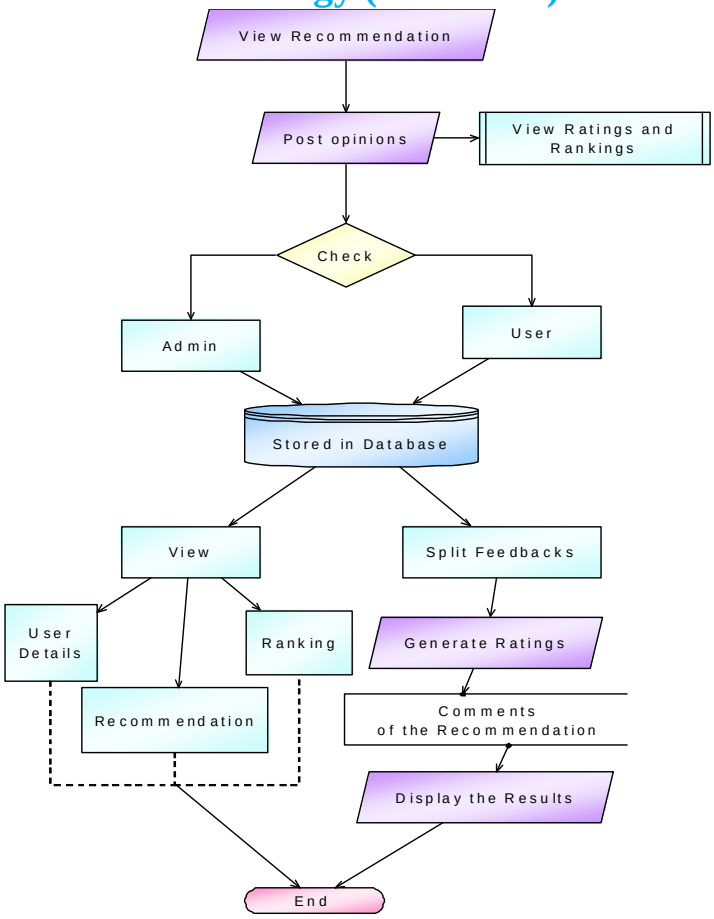
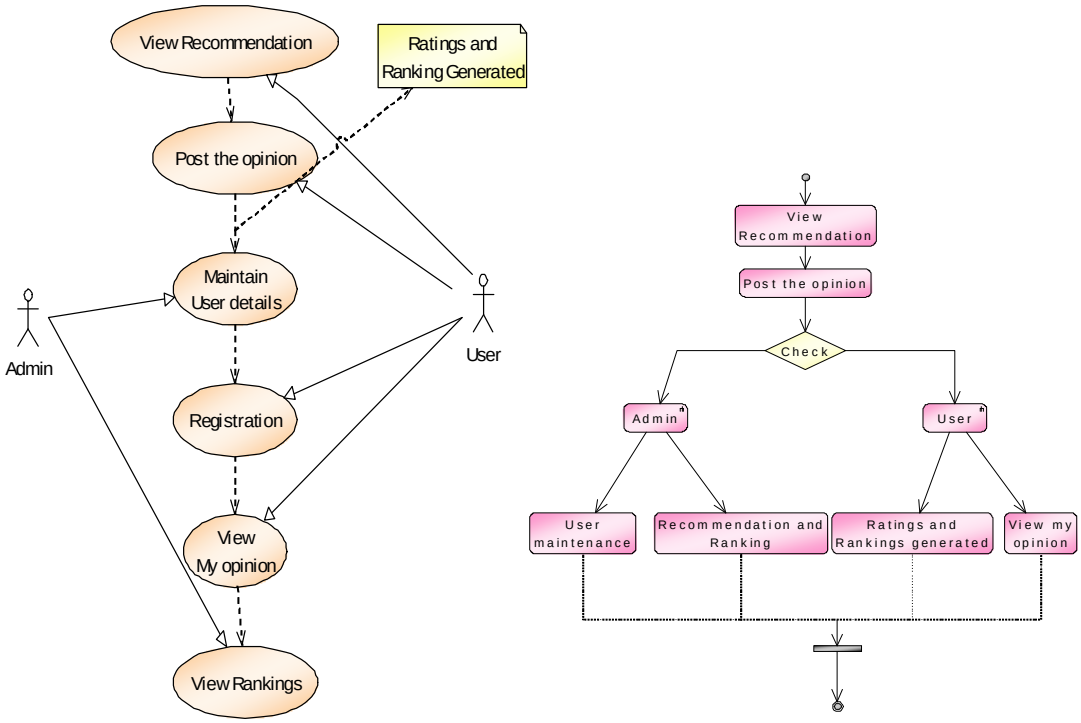
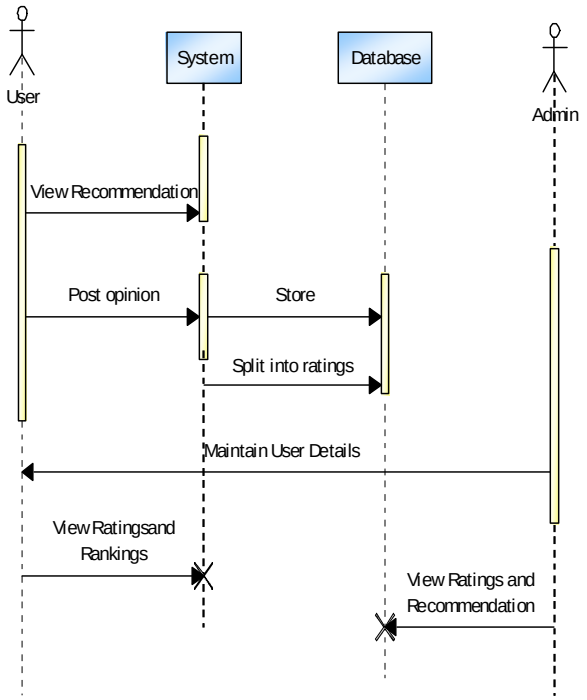


Fig 4. Use case diagram

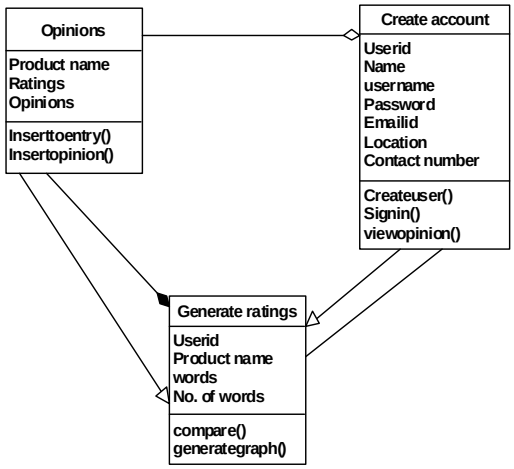


©IJRASET 2015: All Rights are Reserved

E. Activity diagram



Sequence diagram



Class diagram

F. Features of. Net

Microsoft .NET is a set of Microsoft software technologies for rapidly building and integrating XML Web services, Microsoft Windows-based applications, and Web solutions. The .NET Framework is a language-neutral platform for writing programs that can easily and securely interoperate. There's no language barrier with .NET: there are numerous languages available to the developer including Managed C++, C#, Visual Basic and Java Script. The .NET framework provides the foundation for components to interact seamlessly, whether locally or remotely on different platforms. It standardizes common data types and communications protocols so that components created in different languages can easily interoperate. ".NET" is also the collective name given to various software components built upon the .NET platform. These will be both products (Visual

©IJRASET 2015: All Rights are Reserved

Studio.NET and Windows.NET Server, for instance) and services (like Passport, .NET My Services, and so on).

THE .NET FRAMEWORK

The .NET Framework has two main parts:

1. The Common Language Runtime (CLR).
2. A hierarchical set of class libraries.

The CLR is described as the "execution engine" of .NET. It provides the environment within which programs run. The most important features are conversion from a low-level assembler-style language, called Intermediate Language (IL), into code native to the platform being executed on. Memory management, notably including garbage collection. Checking and enforcing security restrictions on the running code. Loading and executing programs, with version control and other such features.

The following features of the .NET framework are also worth description:

G. Managed Code

The code that targets .NET, and which contains certain extra Information - "metadata" - to describe itself. Whilst both managed and unmanaged code can run in the runtime, only managed code contains the information that allows the CLR to guarantee, for instance, safe execution and interoperability.

H. Managed Data

With Managed Code comes Managed Data. CLR provides memory allocation and Deal location facilities, and garbage collection. Some .NET languages use Managed Data by default, such as C#, Visual Basic.NET and JScript.NET, whereas others, namely C++, do not. Targeting CLR can, depending on the language you're using, impose certain constraints on the features available. As with managed and unmanaged code, one can have both managed and unmanaged data in .NET applications - data that doesn't get garbage collected but instead is looked after by unmanaged code.

I. Common Type System

The CLR uses something called the Common Type System (CTS) to strictly enforce type-safety. This ensures that all classes are compatible with each other, by describing types in a common way. CTS define how types work within the runtime, which enables types in one language to interoperate with types in another language, including cross-language exception handling. As well as ensuring that types are only used in appropriate ways, the runtime also ensures that code doesn't attempt to access memory that hasn't been allocated to it.

J. Common Language Specification

The CLR provides built-in support for language interoperability. To ensure that you can develop managed code that can be fully used by developers using any programming language, a set of language features and rules for using them called the Common Language Specification (CLS) has been defined. Components that follow these rules and expose only CLS features are considered CLS-compliant.

K. The class library

.NET provides a single-rooted hierarchy of classes, containing over 7000 types. The root of the namespace is called System; this contains basic types like Byte, Double, Boolean, and String, as well as Object. All objects derive from System. Object. As well as objects, there are value types. Value types can be allocated on the stack, which can provide useful flexibility. There are also efficient means of converting value types to object types if and when necessary. The set of classes is pretty comprehensive, providing collections, file, screen, and network I/O, threading, and so on, as well as XML and database connectivity. The class library is subdivided into a number of sets (or namespaces), each providing distinct areas of functionality, with dependencies between the namespaces kept to a minimum.

L. Languages supported by .net

The multi-language capability of the .NET Framework and Visual Studio .NET enables developers to use their existing programming skills to build all types of applications and XML Web services. The .NET framework supports new versions of Microsoft's old favorites Visual Basic and C++ (as VB.NET and Managed C++), but there are also a number of new additions to the family. Visual Basic .NET has been updated to include many new and improved language features that make it a powerful object-oriented programming language. These features include inheritance, interfaces, and overloading, among others. Visual Basic also now supports structured exception handling, custom attributes and also supports multi-threading.

Visual Basic .NET is also CLS compliant, which means that any CLS-compliant language can use the classes, objects, and

©IJRASET 2015: All Rights are Reserved

components you create in Visual Basic .NET. Managed Extensions for C++ and attributed programming are just some of the enhancements made to the C++ language. Managed Extensions simplify the task of migrating existing C++ applications to the new .NET Framework. C# is Microsoft's new language. It's a C-style language that is essentially "C++ for Rapid Application Development". Unlike other languages, its specification is just the grammar of the language. It has no standard library of its own, and instead has been designed with the intention of using the .NET libraries as its own. Microsoft Visual J# .NET provides the easiest transition for Java-language developers into the world of XML Web Services and dramatically improves the interoperability of Java-language programs with existing software written in a variety of other programming languages. Active State has created Visual Perl and Visual Python, which enable .NET-aware applications to be built in either Perl or Python. Both products can be integrated into the Visual Studio .NET environment. Visual Perl includes support for Active State's Perl Dev Kit.

Fig: .Net Framework

ASP.NET XML WEB SERVICES	Windows Forms
Base Class Libraries	
Common Language Runtime	
Operating System	

TABLE: A database is a collection of data about a specific topic.

C#.NET is also compliant with CLS (Common Language Specification) and supports structured exception handling. CLS is set of rules and constructs that are supported by the CLR (Common Language Runtime). CLR is the runtime environment provided by the .NET Framework; it manages the execution of the code and also makes the development process easier by providing services.

C#.NET is a CLS-compliant language. Any objects, classes, or components that created in C#.NET can be used in any other CLS-compliant language. In addition, we can use objects, classes, and components created in other CLS-compliant languages in C#.NET .The use of CLS ensures complete interoperability among applications, regardless of the languages used to create the application.

M. Constructors and destructors

Constructors are used to initialize objects, whereas destructors are used to destroy them. In other words, destructors are used to release the resources allocated to the object. In C#.NET the sub finalize procedure is available. The sub finalize procedure is used to complete the tasks that must be performed when an object is destroyed. The sub finalize procedure is called automatically when an object is destroyed. In addition, the sub finalize procedure can be called only from the class it belongs to or from derived classes.

N. Garbage collection

Garbage Collection is another new feature in C#.NET. The .NET Framework monitors allocated resources, such as objects and variables. In addition, the .NET Framework automatically releases memory for reuse by destroying objects that are no longer in use. In C#.NET, the garbage collector checks for the objects that are not currently in use by applications. When the garbage collector comes across an object that is marked for garbage collection, it releases the memory occupied by the object.

O. Overloading

Overloading is another feature in C#. Overloading enables us to define multiple procedures with the same name, where each procedure has a different set of arguments. Besides using overloading for procedures, we can use it for constructors and properties in a class.

P. Multithreading

C#.NET also supports multithreading. An application that supports multithreading can handle multiple tasks simultaneously, we can use multithreading to decrease the time taken by an application to respond to user interaction.

Q. Structured exception handling

C#.NET supports structured handling, which enables us to detect and remove errors at runtime. In C#.NET, we need to use

©IJRASET 2015: All Rights are Reserved

Try…Catch…Finally statements to create exception handlers. Using Try…Catch…Finally statements, we can create robust and effective exception handlers to improve the performance of our application.

The .net framework

The .NET Framework is a new computing platform that simplifies application development in the highly distributed environment of the Internet.

Objectives of .net framework

1. To provide a consistent object-oriented programming environment whether object codes is stored and executed locally on Internet-distributed, or executed remotely.
2. To provide a code-execution environment to minimizes software deployment and guarantees safe execution of code.
3. Eliminates the performance problems.

There are different types of application, such as Windows-based applications and Web-based applications.

Features of sql-server

The OLAP Services feature available in SQL Server version 7.0 is now called SQL Server 2000 Analysis Services. The term OLAP Services has been replaced with the term Analysis Services. Analysis Services also includes a new data mining component. The Repository component available in SQL Server version 7.0 is now called Microsoft SQL Server 2000 Meta Data Services. References to the component now use the term Meta Data Services. The term repository is used only in reference to the repository engine within Meta Data Services. SQL-SERVER database consist of six type of objects,They are TABLE,QUERY,FORM, REPORT,MACRO.

Views of table

We can work with a table in two types,Design View Datasheet View

Design View

To build or modify the structure of a table we work in the table design view. We can specify what kind of data will be hold.

Datasheet View

To add, edit or analyses the data itself we work in tables datasheet view mode.

Query

A query is a question that has to be asked the data. Access gathers data that answers the question from one or more table. The data that make up the answer is either dynaset (if you edit it) or a snapshot (it cannot be edited).Each time we run query, we get latest information in the dynaset. Access either displays the dynaset or snapshot for us to view or perform an action on it, such as deleting or updating.

System testing

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent.

Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items:

©IJRASET 2015: All Rights are Reserved

Valid Input : identified classes of valid input must be accepted.
Invalid Input : identified classes of invalid input must be rejected.
Functions : identified functions must be exercised.
Outpu : identified classes of application outputs must be exercised.
Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

All field entries must work properly. Pages must be activated from the identified link.. The entry screen, messages and responses must not be delayed. Features to be tested. Verify that the entries are of the correct format No duplicate entries should be allowed. All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any scheme and requires significant participation by the end user.

It also ensures that the system meets the functional requirements. All the test cases mentioned above passed successfully. No defects encountered.

III. CONCLUSION

Recommender systems have made significant progress in recent years and many approaches have been proposed to improve the recommendation quality. However, in most cases, new approaches are designed to improve the accuracy of the recommendations, whereas the recommendation diversity has often been overlooked. It tends to perform poorly with respect to recommendation diversity. Therefore, in this paper, we proposed a number of recommendation ranking approaches that can provide significant improvements in recommendation diversity with only a small amount of accuracy loss. In addition, these ranking approaches offer flexibility to system designers, since they are parameterizable and can be used in conjunction with different rating prediction algorithms (i.e., they do not require the designer to use only some specific algorithm). They are also based on scalable sorting based heuristics and, thus, are extremely efficient. We provide a comprehensive empirical evaluation of the proposed approaches and obtain consistent and robust diversity improvements across multiple real-world datasets and

©IJRASET 2015: All Rights are Reserved

using different rating prediction approaches.

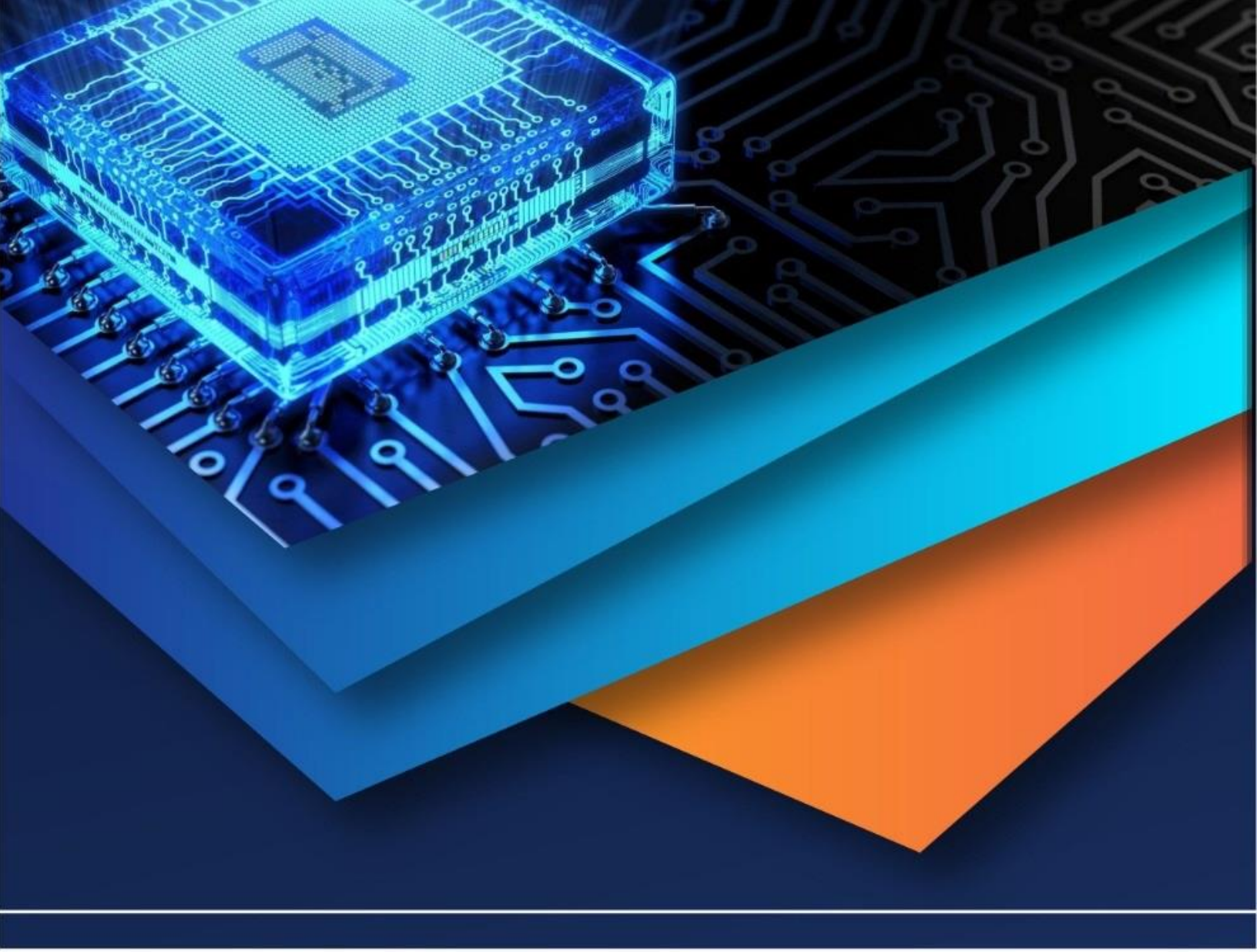
IV. ACKNOWLEDGMENT

The authors thank the anonymous reviewers for their comments, which were invaluable in improving in improving this brief.

REFERENCES

[1] B. Liu, "Sentiment Analysis and Subjectivity," Handbook of Natural language Processing, 2nd ed., N. Indurkhya and F.J. Damerau, eds., Chapman & Hall, 2010, pp. 627–666.
[2] B. Pang and L. Lee, "Opinion Mining and Sentiment Analysis," Foundations and Trends in Information Retrieval, vol. 2, nos. 1–2, 2008, pp. 1–135.
[3] J. Wiebe et al., "Learning Subjective Language," Computational Linguistics, vol. 30, Sept. 2004, pp. 277–308.
[4] M. Hu and B. Liu, "Mining and Summarizing Customer Reviews," Proc. ACM SIGKDD Conf. Knowledge Discovery and Data Mining (KDD), ACM Press, 2004, pp. 168–177.
[5] B. M. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Analysis of Recommender Algorithms for E-Commerce," ACM E-Commerce 2000 Conf., pp.158-167, 2000.
[6] User Interfaces in C#: Windows Forms and Custom Controls by Matthew MacDonald.
[7] Applied Microsoft® .NET Framework Programming (Pro-Developer) by Jeffrey Richter.
[8] Practical .Net2 and C#2: Harness the Platform, the Language, and the Framework by Patrick Smacchia.
[9] Data Communications and Networking, by Behrouz A Forouzan.
[10] Computer Networking: A Top-Down Approach, by James F. Kurose.Operating System Concepts, by Abraham Silberschatz.

©IJRASET 2015: All Rights are Reserved



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)