Online Match-Making Recommendation using Case Based Reasoning and K-Nearest-Neighbors

by

Mehrnaz Bayaki
BSc., Azad University of Mashhad, 2009

A Project Report Submitted in Partial Fulfillment
of the Requirements for the Degree of

Master of Engineering

in the Department of Electrical and Computer Engineering

© Mehrnaz Bayaki, 2014
University of Victoria

All rights reserved. This thesis may not be reproduced in whole or in part, by photocopy or other means, without the permission of the author.
Supervisory Committee

Online Match-Making Recommendations using Case Based Reasoning and K-Nearest-Neighbors

by

Mehnaz Bayaki
BSc. Azad University of Mashhad, 2009

Supervisory Committee

Dr. Issa Traore, Department of Electrical and Computer Engineering
Supervisor

Dr. Kin Li, Department of Electrical and Computer Engineering
Departmental Member
Abstract

Supervisory Committee
Dr. Issa Traore, Department of Electrical and Computer Engineering
Supervisor

Dr. Kin Li, Department of Electrical and Computer Engineering
Departmental Member

In this project a new approach that uses case-based reasoning (CBR) in match-
making for online recommendations is developed. The proposed CBR model uses K-
nearest neighbor (KNN) classification for effective and efficient case retrieval. The
proposed recommendation system is used to match students with potential employees,
based on students’ qualifications and interests, and available projects. The project report
outlines the algorithm design, and corresponding architectural model and
implementation.
# Table of Contents

Supervisory Committee ................................................................. ii  
Abstract ................................................................................................... iii  
Table of Contents ....................................................................................... iv  
List of Tables .............................................................................................. v  
List of Figures ............................................................................................. vi  
Acknowledgments ....................................................................................... vii  
Dedication ..................................................................................................... viii  
1. Chapter 1 Introduction ........................................................................ 1  
   1.1 Context and Overview ................................................................. 1  
   1.2 Report Outline ........................................................................... 2  
2. Chapter 2 Background ......................................................................... 3  
   2.1 Overview of Case-Based Reasoning ........................................... 3  
   2.2 Background on KNN Classification ......................................... 3  
3. Chapter 3 Proposed Match-making Algorithm ................................... 6  
   3.1 Algorithm Design ....................................................................... 6  
   3.2 Initial Case Base and Dataset ..................................................... 11  
   3.3 Evaluation/ Testing: ................................................................. 13  
4. Chapter 4 System Design and Implementation ................................ 15  
5. Chapter 5 Conclusion ........................................................................ 23  
Bibliography ............................................................................................... 24
List of Tables

Table 2.1: Match making system .................................................................................. 7
Table 2.2: QueryList ................................................................................................... 8
Table 2.3: K-Nearest-Neighbors ................................................................................ 8
List of Figures

Figure 2.1 Match-making system software architecture .................................................. 15
Figure 2.2 Subsystems decomposition .............................................................................. 18
Figure 2.3 Package decomposition .................................................................................. 20
Figure 2.4 Package diagram for the match-making system .............................................. 21
I would like to express my honest gratitude to my advisor prof. Dr. Issa Traore for the constant support of my MEng thesis, for his motivation, guidance, and massive knowledge. His assistance directed me in all time of research, design and implementation of my project.

Besides my advisor, I would like to be grateful to Dr. Sherif Saad who significantly assisted me in the design stage of my thesis.

At last, I am deeply thankful to my lovely family, my parents, my brother, Dr. Ehsan Bayaki, and my sister Elnaz Bayaki for their endless compassion and support whom had an outstanding role during completion of my thesis and graduate studies.
Dedication

I would like to dedicate my thesis to my parents Hossein Bayaki and Marzieh Kashani for their endless love, support and encouragement in all stages of my life.
Chapter 1 Introduction

1.1 Context and Overview

Online recommendation systems play a critical role in the success of many current e-commerce organizations such as NetFlix, Amazon, and so on. The project presented in this report was a collaboration with a local startup company to develop an online recommendation system for match-making between companies and students looking for projects or internship positions.

The previous online matchmaking platform allowed collecting information related to individuals (i.e. students) and organizations (e.g. companies) using traditional search engines. The collected information was then manually analyzed by experts to extract entities (i.e. company or students) profiles and then perform matchmaking between students and companies. Students can follow and request to do a project with a specific company and an email will be sent out to that company to tell them to join the matchmaking site and fill out their profile and submit projects descriptions for interested students.

The goal of the MENG project has been to fully automate the previous platform by researching and developing efficient data extraction and structuring mechanisms and the matchmaking algorithms that will match users to one another based on data inputted through their profiles.

In this project, a case-based reasoning model based on the K-nearest-neighbor (KNN) algorithm is used in order to recommend suitable job opportunities to students. This is a decision making system which uses KNN classification algorithm for pattern classification.
1.2 Report Outline

The rest of the report is structured as follows.

Chapter 2 summarizes background knowledge on case-based reasoning and KNN classification.

Chapter 3 introduces the proposed match-making algorithm.

Chapter 4 presents the algorithm design and implementation.

Chapter 5 makes some concluding remarks.
Chapter 2 Background

2.1 Overview of Case-Based Reasoning

Case-Based Reasoning (CBR) is the procedure of solving a problem based on the solutions of similar past problems. A case typically comprises of two main parts: the existing problem description and a solution to solve it. In addition to that, the case might indicate the consequences of the given solution (success or failure). The CBR process is defined as following [3]:

1. Retrieve: Given a case, CBR system tries to retrieve the most similar cases to the problem case.
2. Reuse: Mapping a solution from the previous similar cases to the objective problem.
3. Revise: Test the new solution for the target problem and review it to adapt to the target problem if required.
4. Retain: After the successful adjustment of solution to the objective problem, store the case as a new case in the memory.

CBR performance is dependent on two factors: precision and swiftness of retrieval and reuse algorithms.

2.2 Background on KNN Classification

Machine learning techniques are categorized into two groups: supervised and unsupervised methods [7, 8]. Supervised learning is the procedure of learning a mapping between a set of input variables and an output variable and applying this mapping to predict the outputs for unseen data. Supervised learning is the most common methodology in machine learning. Classification and regression data mining methods are classified as supervised machine learning procedures. Unsupervised learning attempts to find unseen structure in unmarked data. Clustering and association mining are placed in this group [1, 2].
Classification is a classic data mining technique based on machine learning.Mainly, classification is used to categorize each item in a set of data into one of predefined, distinct set of classes or groups. As it is a supervised method, the categories (or classes) need to be previously defined and known for the data that is used as the training set (i.e. each instance of data needs to be associated with a “class label”, representing which class the instance belongs to).

We used in this project a CBR model that is based on KNN, which is a popular classification algorithm. KNN is one of the simplest lazy learning algorithms. It has been frequently used in pattern classification due to its simplicity and effectiveness [4]. Let us assume we have $N$ training instances which are labeled $\{X_i, C_i\}$. The algorithm stores all of the training instances in the memory. Class prediction for a new instance $x_0$ is done by finding its $k$ nearest neighbors based on a similarity metric, and then assigning a label using majority vote of its $k$ nearest neighbor class labels. Typically, Euclidean distance is used as the similarity metric for measuring the distances between instances. According to the studies on pattern classification algorithms, the performance of KNN is comparable with most sophisticated algorithms in this area. Although KNN is moderately precise and simple for implementation, it may result in low efficiency if some algorithm parameters are selected inadequately. Likewise, determining an optimum value for $k$ is a great challenge since the efficiency of the algorithm is directly affected by the value selected for $k$.

Normally, the performance of KNN is affected by the following parameters and factors: number of nearest neighbors ($k$), size of training instances ($N$) and selection of similarity metric. The choice of $k$ is very critical in this algorithm. Classically, picking larger values for $k$ can be more resilient to noise, but that is more expensive in terms of computation cost. Specifying the value of $k$ is still a challenge and determining optimum values for $k$ is dissimilar in different applications [5, 6].

In KNN, selecting a high value for the number of instances ($N$) can result in better performance of the algorithm. For the reason that picking large quantities for $N$ helps
finding more similar $k$ nearest neighbors for each new instance, although a large training set can be costly in terms of computation and memory space [5, 6, 9].
Chapter 3 Proposed Match-making Algorithm

3.1 Algorithm Design

The goal of the project is to develop a match-making algorithm that connects individuals (i.e. students) and organizations based on their preferences in collaborating on specific projects. For recommending appropriate matches (organizations and their projects) to students, we have made use of a CBR system which employs KNN as its retrieval algorithm. We use case-based reasoning to retrieve previously successful matches. A successful match here is defined as following: the organization was satisfied by the student performance, the student has gained experience and knowledge and the project was successful. This is a classification algorithm in which our instances are stored in the case base and are labeled with two classes: positive feedback and negative feedback. Positive feedback indicates a successful match, while negative feedback specifies a failed match between the student, organization and the project.

Our goal is to recommend a suitable match to the student based on previous successful matches. The match making procedure will consist of comparing a new case, defined in terms of the student, the organization and the project, to previous cases in the CBR system. As mentioned above, we use KNN as retrieval algorithm in the CBR system.
In order to give an overview of our proposed match making system, let us consider the example database depicted by Table 2.1. This involves lists of students, organizations, projects, and a sample case base. Now, let us say we want to find a job opportunity for the student S2 in the student table. For doing that, we generate a job opportunity query list as follows. For each organization and project pair in the Organization/Project table we create a JOB Opportunity Query that consists of the student, the organization and project pair. In Table 2.2 below, we have created the job opportunity query list for student S2:
In this part, we want to predict which job opportunity will most probably have a positive feedback (i.e. will be a positive experience for the student and the company). For doing that, we will make our decision based on the previous cases stored in our case base. For each query in the query list (in this case Q1, Q2, Q3... Q7) we will search the case-base looking for the $k$ nearest/similar cases. This $k$ should be an odd number (e.g. 1, 3, 5, 7, 9, etc.). Then we predict the expected feedback for the query using majority voting over the $k$ nearest/similar cases. To illustrate, let us assume that we are using $k = 5$ and after running the KNN search over the case-base, we got the 5 similar cases to each query listed in Table 2.3.

Table 2.2: Query List Example

<table>
<thead>
<tr>
<th>Query ID</th>
<th>Student</th>
<th>Organization</th>
<th>Project</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>S2</td>
<td>O1</td>
<td>P1</td>
</tr>
<tr>
<td>Q2</td>
<td>S2</td>
<td>O2</td>
<td>P7</td>
</tr>
<tr>
<td>Q3</td>
<td>S2</td>
<td>O1</td>
<td>P3</td>
</tr>
<tr>
<td>Q4</td>
<td>S2</td>
<td>O1</td>
<td>P2</td>
</tr>
<tr>
<td>Q5</td>
<td>S2</td>
<td>O3</td>
<td>P6</td>
</tr>
<tr>
<td>Q6</td>
<td>S2</td>
<td>O1</td>
<td>P4</td>
</tr>
<tr>
<td>Q7</td>
<td>S2</td>
<td>O2</td>
<td>P5</td>
</tr>
</tbody>
</table>

Table 2.3: K-Nearest-Neighbors Examples

<table>
<thead>
<tr>
<th>Query ID</th>
<th>Similar Cases</th>
<th>Predicted Feedback</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>3, 4, 9, 11 and 14</td>
<td>Negative</td>
</tr>
<tr>
<td>Q2</td>
<td>1, 3, 4, 9 and 14</td>
<td>Negative</td>
</tr>
<tr>
<td>Q3</td>
<td>1, 4, 5, 8 and 9</td>
<td>Negative</td>
</tr>
<tr>
<td>Q4</td>
<td>3, 4, 8, 11 and 14</td>
<td>Negative</td>
</tr>
<tr>
<td>Q5</td>
<td>2, 4, 5, 6, 9 and 11</td>
<td>Negative</td>
</tr>
<tr>
<td>Q6</td>
<td>3, 5, 6, 11 and 14</td>
<td>Positive</td>
</tr>
<tr>
<td>Q7</td>
<td>3, 6, 7, 10 and 11</td>
<td>Positive</td>
</tr>
</tbody>
</table>
Based on the \((k = 5)\) similar cases to each job opportunity query, the most likely job opportunity suitable for student S2 are the ones in Q6 and Q7. This is because 3 similar cases to Q6 had positive feedback and 4 similar cases to Q7 had positive feedback. The system at the end should recommend the student to apply to work for organization O2 on project P5.

At this time, the proof-of-concept software implementation is designed by using \(k = 1\). Hence for each query in the query list the similar cases to the query are retrieved from the case-base, however, the feedback of the most similar case among similar cases will be assigned to the feedback feature of the query. For more illustration, assume that we are searching the case base using \(KNN\) algorithm, we will find out that Q6 has 5 similar cases (3, 5, 6, 11 and 14), but case 3 has the highest similarity to Q6 compared to the 4 other cases. So, the feedback for Q6 would be positive. This indicates we can recommend to student S2 to work for organization O1 on project P4.

The steps of our proposed match-making algorithm are defined as follows:

1. Create \textit{JobOpportunities} as an empty List.
2. For each \textit{organization} with some \textit{open project} in the system database (DB) read the organizations and their open projects.
   
   2.1. Extract useful organization and project features/attributes and create organization and project objects.
   
   2.2. Set each organization object and project object as a job opportunity JO.
   
   2.3. Add JO to the JobOpportunities list.
3. From \textit{DB} read one student profile.
4. Extract useful student feature/attributes from the student’s profile and create a student object.
5. Create \textit{Q} as an empty queries list.
6. For each \textit{JO} in the \textit{JobOpportunities}:
6.1. If the \textit{JO} not suitable for the student drop this \textit{JO}.

6.2. Create a query \textit{q}, where \textit{q} consists of the student object and the \textit{JO}.

6.3. Add \textit{q} to the \textit{Q} list.

7. Create \textit{R} as an empty result list.

8. For each \textit{q} in the \textit{Q} list:

   8.1. Use KNN search to find the nearest \textit{K} cases to \textit{q} in the case base.

   8.2. Use majority voting with \textit{K} voters to predict the feedback (good or not good) of \textit{q}.

   8.3. Add \textit{q} and the result to the Results list

9. From the Result list select the top \textit{m} positive feedback and display it to the students
3.2 Initial Case Base and Dataset

The sample dataset provided by the company was stored in PostgreSQL database containing different tables storing information for students, businesses and projects. The number of the records for students, businesses, and projects are 12, 10, and 41, respectively. The considerable point to mention is that each organization has one or more projects. As per of the company’s request, some of the attributes for each of the students, businesses and projects were taken from the dataset tables to use for matching.

The attributes that were taken for matching from the student_profiles table in the dataset are the following:

1. Student year of study
2. Student major
3. Student education level id

Student’s industry was taken from sectors table that was inner join to the sectorizations table in the database using a SQL script.

Student’s skills were taken form taggings table which was inner join to the tags table in the database using a SQL script.

The attributes that were taken for matching organizations from the business_profiles table in the database are:

1. OrgTypeId: indicates the organization type
2. OrgSizeId: indicates organization size.

Project attribute that was taken from the projects table in the database is:

1. Project Funding available

Project (required) skills are taken from sectors and sectorizations table, and represented as a set of strings.

Project industry is an array of strings that indicates the industry areas of the project and is taken from taggings and tags table.

Other tables were also defined in the database. The tables that were used for this project from the dataset are:

1. student_profiles
2. business_profiles
3. projects
4. sectorizations  
5. sectors  
6. taggings  
7. tags

We created the initial case for the system based on expert advice and the business practice of the sponsoring company. The case base is written as an XML file containing 10 cases. Each case contains student, organization, and project information.

**Student Attributes**

The student information are major which is a string value, year of study, which is an integer value and education_level_id as an integer; education_level_id can have the following values:

- 1 for a high school student  
- 2 for an undergraduate student  
- 3 for a master student  
- 4 for a PhD student  
- 5 for a post-doc fellow.

Furthermore, student_industry, which is a string type, specifies student’s industry and at last student_skill, that is an array of strings represents student’s skills.

**Organization Attributes**

Organization information consists of Org_type_id and Org_size_id.

Org_type_id is an integer, which can have the following different values:

- 0 for private  
- 1 for government  
- 2 for non-profit.  
- 3 for educational

Org_size_id is an integer that refers to the number of employees in the organization; it can have the following possible values:

- 0 for the size of 1-5  
- 1 for the size of 6-20  
- 2 for the size of 20-50  
- 3 for the size of 50+
Project attributes

Project attributes are skills, industry and Funding_available. Funding_available is defined as follows:

\( \text{Funding\_available} \): a Boolean value that specifies whether the project has fund (true value) or no fund (false value) for the student that will work on it.

3.3 Evaluation/Testing:

The number of queries is 240. The system failed to find matches for 4 queries out of 240.

We evaluated the system using the test samples provided by the sponsoring company.

The similar cases to each query were printed on the screen using KNN algorithm (as per of the company request here \( k = 1 \), and it might be larger later, but an odd number should be taken for the value of \( k \)). In order to calculate similarity between each query and all the 10 cases, an instance of the caseRetriever class is created to do the comparison. The caseRetriever class will create an instance of similarityMetric class to do the calculation according to the formulas shown below. The final similarity is the sum of similarities between students, organizations and projects. The following formulas are shown for illustration:

\[
\text{similarity} = \text{studentSimilarity} + \text{organizationSimilarity} + \text{projectSimilarity}
\]

\[
\text{studentSimilarity} = \text{similarity in yearOfStudy} + \text{similarity in major} + \text{similarity in education\_level\_id} + \text{similarity in skills} + \text{similarity in industry}
\]

\[
\text{organizationSimilarity} = \text{similarity in organization type} + \text{similarity in organization size}
\]

\[
\text{projectSimilarity} = \text{similarity in project skills} + \text{similarity in project industry} + \text{similarity in available funding}
\]

If the similarity is more than a specific threshold, then that case would be counted as a similar case to our query and will be stored along with the similarity measure and the value of its feedback which is either positive or negative. In case, none of the cases were
similar to our query, a message will be printed on the screen: “No similar cases found for this query”. Then the similarity measures between the different cases for each query are compared, and the case with the highest similarity will be selected ($k = 1$) as the nearest neighbor to our query and if the feedback is positive, then the query feedback is marked as positive (shows a suitable match) otherwise it will be marked as negative (specifies that the matching between this student, organization and project will not be successful).

At last we only printed out the matches with positive feedback (successful matches to recommend to the students) for each student in the queryList.
Chapter 4 System Design and Implementation

We designed and documented the match-making software architecture using the UML. Figure 2.1 illustrates the UML class diagram for the system.

Figure 2.1 Match-making system software architecture
The following brief descriptions of the different classes involved in the match-making system are provided:

1. **Student**: This class describes the student object.
2. **Organization**: This class describes the organization object.
3. **Project**: This class describes the project object.
4. **Case**: This class describes the case object in the case base. Each case object consists of a student object, an organization object, a project object and a variable of string type named *feedback* that can have two values: positive and negative. The feedback variable indicates the relation between the student, organization and project. If the feedback is positive, it shows that the experience of doing the project by the student for the organization was satisfying for both the student and the organization. Otherwise, if the experience was not satisfying, the feedback variable has negative value. The case class has these attributes: case id (integer number), student object, organization object, project object and feedback variable of type string.
5. **Query**: This class describes the query we want to answer; the attributes of this class are Query ID, student object, organization object and project object.
6. **JobOpportunity**: This class is used for matching the organizations with their projects by using business_id attribute common in business_profiles table and projects table. The attributes of this class are organization object and project object that defines a job opportunity.
7. **DataBaseIO**: This class is used for interacting with the database. We use this class to read data from the database.
8. **CaseIO**: This class is used for reading and writing data from/to the case-base.
9. **CaseRetriever**: This class is used for our search process; it implements the search algorithm to find the similar cases in the case-base for our query in the Query list table. In this system *KNN* algorithm is used for finding the nearest neighbours in the case-base.
10. **SimilarityMetric**: This class encapsulates the similarity metric which is used by the CaseRetriever class. Different similarity metrics can be used depending on the type and characteristic of the data.
11. **SimilarCases**: This class describes a similar case to the query. The attributes of this class are similarity (Integer number) that shows the value of the similarity between the case and the query in the Query list, and relation (String attribute), which indicates the feedback related to this case.

12. **ResultList**: This class describes the matches found for each query. The attributes of this class are Result ID, student object, organization object, project object, feedback of type string (that indicates that the experience of getting this student to work on the specified project in the company would be a positive experience or a negative experience); the similarity of the nearest neighbour if $k = 1$ is used. If the feedback for this query is calculated according to the number of positive feedbacks and negative feedbacks of the query nearest neighbours, the similarity attribute is null and presents 0.

13. **Main**: This is the main (or process) class that manages the match-making system.

14. **studentSkill**: This class is used to capture student’s skills.

15. **studentIndustry**: This class is used to capture industry interest for a student.

16. **organizationIndustry**: This class is used to capture industry information for an organization.

17. **projectSkill**: This class is used for creating skills needed for a particular project.

18. **projectIndustry**: This class is used to capture industry for a project.
(a) Student subsystem  (b) Organization subsystem  (c) Project subsystem  (d) Query subsystem

(e) Case subsystem  (f) Retrieve subsystem  (g) DatabaseIO subsystem  (h) Result subsystem

Figure 2.2 Subsystems decomposition
As can be seen from Figure 2.1, the class diagram is really large, so we revisit the class structure by grouping them in subsystems, and the subsystems into packages. Finally, the whole system is shown by the relationships between packages which help to analyze the system architecture from a higher-level perspective. We consider 8 subsystems depicted by Figure 2.2.
Figure 2.3 Package decomposition
The subsystems are structured into 4 packages depicted by Figure 2.3, and defined as follows:

1. Query package: This package consists of student, Organization, Project and Query subsystems with their relationships in the class diagram.
2. CBR package: This package consists of case and retrieve subsystems.
3. DataBaseIO package: This package contains DataBaseIO class.
4. Result package: This package contains ResultList class.

The Package diagram of the whole system is depicted by Figure 4.

![Figure 2.4 Package diagram for the match-making system](image)

The following relationships are defined for the packages listed in Figure 2.4:

1) Query and CBR: Since some of the class elements in Query package have aggregation or association relationships with some class diagrams in CBR
package, there is an access relationship from Query package to CBR package. In contrast, CBR package has also the same relationships with Query Package, hence there is another access relationship from CBR package to Query package.

2) DataBaseIO and Query: DataBaseIO package consists of one class and has association relationship with some of the classes in Query package. This can specify an access relation from DataBaseIO to Query.

3) Result and Query: Result package which contains ResultList class has aggregation relation with one class in Query package; therefore we define an access relationship from Result to Query package.

4) Result and CBR: Result package which contains ResultList class has aggregation relation with one class in CBR package, this defines access relationship from Result to CBR package.

The above system architecture was implemented in Java and deployed on the web portal of the industry partner involved in the project.
Chapter 5 Conclusion

This report presents a match making system using case based reasoning. The case-based reasoning system uses a KNN classifier as its retrieval algorithm in matching students and organizations on projects.

The project was carried out in order to fulfill the requirements set by a Victoria-based e-commerce company. One of the company’s critical success factors is their ability to match students with organizations (like what online dating does for singles). However, their previous online matchmaking scheme relied on human experts to manually analyze information from students and companies in order to find adequate matches. Not only our proposed match-making algorithm embodies the existing human expertise, but more importantly, it allows automating the match-making process in more effective way. The proposed algorithm has been fully implemented, and is currently deployed and operating at the company’s site.


