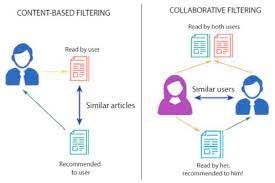
**CHAPTER 1**

**INTRODUCTION**

The highly competitive and dynamic nature of the job market as well as personal preferences and goals lead individuals to change their jobs at some point in their lives. Moving to a new job, however, is not an easy decision, which may depend on many factors, such as salary, job description, and geographical location. Making successful job transitions is essential for a successful professional career. In this work, we build an automated system that can recommend jobs to people based on their past job histories in order to facilitate the process of selecting a new job. We believe that such a system can successfully exploit the job transitions performed by other employees. That is, we propose recommending jobs to people based on inference from the job transition patterns observed in the past. These patterns may involve features extracted from the business profiles of employees (e.g., years of experience, educational degree, job title), the profiles of institutions1 (e.g., industry, type, size), and the job transitions themselves (e.g., frequency of transitions between jobs, average time spent in a job).



The framework we propose is based on K means clustering. Given an employee’s past job history, the objective of the learning model is to accurately predict the next institution that the employee will move to. The predicted institution can then be recommended to the employee as the next step in his/her career. To evaluate our framework, we use a large sample of job transitions extracted from the publicly available employee profiles in the Web. From this sample, we extract a number of features that we use to train and test our machine learning model. The results of our experiments demonstrate that the transition of an employee to an institution can be quite accurately predicted, significantly improving over a baseline predictor that always predicts the most frequent institution in the data. Our results indicate that the most important feature in predicting a job transition is the current institution of the employee.

**Problem Definition**

In existing job recommendation sites, job offers which are matching with user’s profile used to recommend user. To improve this recommendation we proposed machine learning algorithms which will find user’s preferences and recommend job offers as per the preferences, profile and requirements of the recruiters.

**Problem Objectives**

* To implement K means clustering algorithm
* To develop an online job recommendation application for job seekers
* To implement recommendation model

**CHAPTER 2**

**LITERATURE SURVEY**

The study of methods for job search and factors that determine the success of individuals in a labor market is an extensive topic of research in the area of labor economics. In this area, the analysis of social networks has played an important role in analyzing how people are searching for jobs. For instance, in his seminal work, Granovetter showed that weak ties are superior to strong ties for providing support ingetting a job [5]. This evidence has led to a number of the-  
oretical models in economics that explore the importance of social networks in labor markets (e.g., Calv ́o-Armengol [1], Calv ́o-Armengol and Zenou [2], Galeotti and Merlino [4]). Another well investigated scenario in assigning jobs to individuals is via the theory of matching [3, 7, 9]. In this scenario, many applicants rank a set of available job positions in terms of preferences, and similarly the employers in those positions rank the applicants. Then, an optimal  
matching is sought, in various technical notions of optimality. This setting is obviously very different than the problem we study in this project. More similar to our work, the problem of recommending jobs to individuals (as well as the dual problem of recommending applicants for job profiles) was studied by Malinowski et al. [8], who propose to learn a probabilistic model that estimates the probability that an applicant likes a job.  
  
This approach differs from our work in that it requires more information. It is assumed that not only applicant profiles are available, but also job opening profiles, and the goal is to match applicant to job opening profiles. In our work, we do not assume that we know the job opening profiles, instead we recommend job positions to applicants based only on the previous job history of a number of other employees. Another difference is that through the machine learning approach and the prediction methodology we follow, we offer a quantitative way of evaluation.  
Finally, a somewhat related concept is that of churn prediction [10]. Here, the challenge is to predict when a customer (and not an employee) will leave a company and possibly sign up with a competitor company. The churn prediction scenario is orthogonal to the problem we focus on,  
where we try to predict not when someone will leave, but what would be the next move.

**CHAPTER 3**

**SYSTEM ANALYSIS**

**3.1 PROBLEM STATEMENT:**

The highly competitive and dynamic nature of the job market as well as personal preferences and goals lead individuals to change their jobs at some point in their lives. Moving to a new job, however, is not an easy decision, which may depend on many factors, such as salary, job description, and geographical location. Making successful job transitions is essential for a successful professional career. In this

**3.2 SYSTEM REQUIREMENTS:**

Our project has some software requirements as follows:

**Eclipse Software:**

Eclipse is an integrated development environment (IDE) used in computer programming. It contains a base workspace and an extensible plug-in system for customizing the environment. We used this software to develop our project because of it’s compatibility with java programming language. The Eclipse SDK includes the Eclipse Java development tools (JDT), offering an IDE with a built-in Java incremental compiler and a full model of the Java source files. This allows for advanced refactoring techniques and code analysis. The IDE also makes use of a workspace, in this case a set of metadata over a flat file space allowing external file modifications as long as the corresponding workspace resource is refreshed afterward.  
Eclipse implements the graphical control elements of the Java toolkit called Standard  
Widget Toolkit (SWT), whereas most Java applications use the Java standard Abstract  
Window Toolkit (AWT) or Swing. Eclipse's user interface also uses an intermediate graphical  
user interface layer called JFace, which simplifies the construction of applications based on  
SWT.

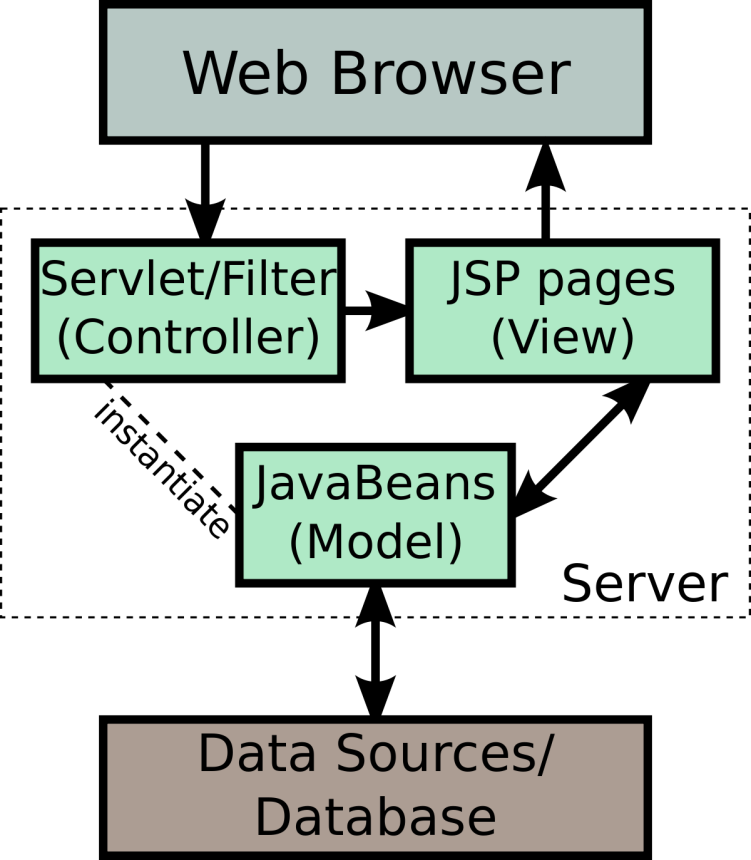
**MYSQL**

MySQL is an open-source relational database management system (RDBMS). A  
relational database organizes data into one or more data tables in which data types may be related to each other; these relations help structure the data. SQL is a language programmers use to create, modify and extract data from the relational database, as well as control user access to the database. In addition to relational databases and SQL, an RDBMS like MySQL works with an operating system to implement a relational database in a computer's storage system, manages users, allows for network access and facilitates testing database integrity and creation of backups

TECHNOLOGIES INVOLVED

**Jakarta Server Pages (JSP):**  
Jakarta Server pages is one of the original java web technology which is being widely  
used to create dynamic web pages that can connect to java backend. It is built on top of the  
Java Servlet specification. JSP may be viewed as a high-level abstraction of Java servlets.  
JSPs are translated into servlets at runtime, therefore JSP is a Servlet; each JSP servlet is  
cached and re-used until the original JSP is modified. Jakarta Server Pages can be used  
independently or as the view component of a server-side model–view–controller design,  
normally with JavaBeans as the model and Java servlets (or a framework such as Apache  
Struts) as the controller.

JSP allows Java code and certain predefined actions to be interleaved with static web  
markup content, such as HTML. The resulting page is compiled and executed on the server to  
deliver a document. The compiled pages, as well as any dependent Java libraries, contain Java  
bytecode rather than machine code. Like any other .jar or Java program, code must be  
executed within a Java virtual machine (JVM) that interacts with the server's host operating  
system to provide an abstract, platform-neutral environment. JSPs are usually used to deliver HTML and XML documents, but through the use of OutputStream, they can deliver other types of data as well. The Web container creates JSP implicit objects like request, response, session, application, config, page, pageContext, out and exception. JSP Engine creates these objects during translation phase. Architecturally, JSP may be viewed as a high-level abstraction of Java servlets. JSPs are translated into servlets at runtime, therefore JSP is a Servlet; each JSP servlet is cached and re-used until the original JSP is modified. JSP can be used independently or as the view component of a server-side modelâ€“viewâ€“controller design, normally with JavaBeans as the model and Java servlets as the controller.



JSP technology is the extension to Servlet technology. The main features of JSP technology are as follows:  
❖ A language for developing JSP pages, which are text-based documents that describe how to process a request and construct a response  
❖ An expression language for accessing server-side objects  
❖ Mechanisms for defining extensions to the JSP language  
Servlets provide URL mapping and request handling capabilities in your Java web applications. Request handling is the bread and butter of Java web application development. In order to respond to requests from the network, a Java web application must first determine what code will respond to the request URL, then marshal a response. Every technology stack has a way of accomplishing request-response handling. In Java, we use servlets (and the Java Servlet API) for this purpose. Think of a servlet as a tiny server whose job is to accept requests and issue responses.

Bootstrap:  
Bootstrap is the most popular open-source framework full of useful and common classes to use in any project. It helps to develop responsive and mobile-first websites faster and easier. It is known for its faster and effortless responsive web development assistance, Bootstrap web design methodology utilize HTML and CMS based templates for user interface components like forms, navigations, alerts, buttons, typography in addition to optional JavaScript extensions. Bootstrap is a web framework that focuses on simplifying the development of informative web pages (as opposed to web apps). The primary purpose of adding it to a web project is to apply Bootstrap's choices of color, size, font and layout to that project. As such,  
the primary factor is whether the developers in charge find those choices to their liking. Once added to a project, Bootstrap provides basic style definitions for all HTML elements. The result is a uniform appearance for prose, tables and form elements across web browsers. In addition, developers can take advantage of CSS classes defined in Bootst rap to further customize the appearance of their contents. For example, Bootstrap has provisioned for light-and dark-colored tables, page headings, more prominent pull quotes, and text with a highlight. Bootstrap also comes with several JavaScript components in the form of jQuery plugins. They provide additional user interface elements such as dialog boxes, tooltips, and carousels. Each Bootstrap component consists of an HTML structure, CSS declarations, and in some cases accompanying JavaScript code. They also extend the functionality of some existing interface elements, including for example an auto-complete function for input fields. The most prominent components of Bootstrap are its layout components, as they affect an entire web page. The basic layout component is called "Container", as every other element in the page is placed in it. Developers can choose between a fixed-width container  
and a fluid-width container. While the latter always fills the width of the web page, the former uses one of the four predefined fixed widths, depending on the size of the screen showing the page:

Smaller than 576 pixels  
576–768 pixels  
768–992 pixels  
992–1200 pixels Larger than  
1200 pixels  
Once a container is in place, other Bootstrap layout components implement a CSS Flexbox layout through defining rows and columns.  
A precompiled version of Bootstrap is available in the form of one CSS file and three JavaScript files that can be readily added to any project. The raw form of Bootstrap, however, enables developers to implement further customization and size optimizations. This raw form  
is modular, meaning that the developer can remove unneeded components, apply a theme and modify the uncompiled Sass files

**JavaScript**

JavaScript is one of the core technologies of the WWW (World Wide Web). It enables interactive web pages and is an essential part of web applications. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM). Almost all the websites and web browser uses JavaScript engines to execute client side page behavior. JavaScript engines were  
originally used only in web browsers, but they are now embedded in some servers, usually via Node.js. They are also embedded in a variety of applications created with frameworks such as Electron and Cordova

**Spring MVC**

Spring's Web MVC framework is designed around a DispatcherServlet that dispatches requests to handlers, with configurable handler mappings, view resolution, locale and theme resolution as well as support for upload files. The default handler is a very simple Controller interface, just offering a ModelAndView handleRequest(request,response) method. This can already be used for application controllers, but you will prefer the included implementation hierarchy, consisting of, for example AbstractController, AbstractCommandController and SimpleFormController. Application controllers will typically be subclasses of those. Note that you can choose an appropriate base class: if you don't have a form, you don't need a form controller. This is a major difference to Struts. Spring Web MVC allows you to use any object as a command or form object - there is no need to implement a framework-specific interface or base class. Spring's data binding is highly flexible: for example, it treats type mismatches as validation errors that can be evaluated by the application, not as system errors. All this means that you don't need to duplicate your business objects' properties as simple, untyped strings in your form objects just to be able to handle invalid submissions, or to convert the Strings properly. Instead, it is often preferable to bind directly to your business objects. This is another major difference to Struts which is built around required base classes such as Action and ActionForm.

Compared to WebWork, Spring has more differentiated object roles. It supports the notion of a Controller, an optional command or form object, and a model that gets passed to the view. The model will normally include the command or form object but also arbitrary reference data; instead, a WebWork Action combines all those roles into one single object. WebWork does allow you to use existing business objects as part of your form, but only by making them bean properties of the respective Action class. Finally, the same Action instance that handles the request is used for evaluation and form population in the view. Thus, reference data needs to be modeled as bean properties of the Action too. These are (arguably) too many roles for one object.

Spring's view resolution is extremely flexible. A Controller implementation can even write a view directly to the response (by returning null for the ModelAndView). In the normal case, a ModelAndView instance consists of a view name and a model Map, which contains bean names and corresponding objects (like a command or form, containing reference data). View name resolution is highly configurable, either via bean names, via a properties file, or via your own ViewResolver implementation. The fact that the model (the M in MVC) is based on the Map interface allows for the complete abstraction of the view technology. Any renderer can be integrated directly, whether JSP, Velocity, or any other rendering technology. The model Map is simply transformed into an appropriate format, such as JSP request attributes or a Velocity template model.

**CHAPTER 4**

**SYSTEM ARCHITECTURE**

The highly competitive and dynamic nature of the job market as well as personal preferences and goals lead individuals to change their jobs at some point in their lives. Moving to a new job, however, is not an easy decision, which may depend on many factors, such as salary, job description, and geographical location. Making successful job transitions is essential for a successful professional career. In this

**CHAPTER 5**

**IMPLIMENTATION AND RESULT**

The highly competitive and dynamic nature of the job market as well as personal preferences and goals lead individuals to change their jobs at some point in their lives. Moving to a new job, however, is not an easy decision, which may depend on many factors, such as salary, job description, and geographical location. Making successful job transitions is essential for a successful professional career. In this

**CHAPTER 6**

**CONCLUSION AND FUTURE SCOPE**

The highly competitive and dynamic nature of the job market as well as personal preferences and goals lead individuals to change their jobs at some point in their lives. Moving to a new job, however, is not an easy decision, which may depend on many factors, such as salary, job description, and geographical location. Making successful job transitions is essential for a successful professional career. In this

**REFERENCES**

The highly competitive and dynamic nature of the job market as well as personal preferences and goals lead individuals to change their jobs at some point in their lives. Moving to a new job, however, is not an easy decision, which may depend on many factors, such as salary, job description, and geographical location. Making successful job transitions is essential for a successful professional career. In this