|  |  |
| --- | --- |
| **khcnlogovnright** | **BỘ GIÁO DỤC VÀ ĐÀO TẠO**  **TRƯỜNG ĐẠI HỌC HOA SEN**  **KHOA KHOA HỌC VÀ CÔNG NGHỆ** |

**XÂY DỰNG KIẾN TRÚC**

**CỔNG THÔNG TIN TÌM VIỆC**

Giảng viên hướng dẫn : TS. Trần Vũ Bình

Nhóm sinh viên thực hiện : Lê Trung Hiếu (09015L)

Phùng Chí Nguyên (09023L)

Lê Dương Công Phúc (09025L)

Lớp : QL092L

**Tháng 12 năm 2011**

Table of Contents

[DANH MỤC HÌNH ẢNH 4](#_Toc310512110)

[DANH MỤC BẢNG 4](#_Toc310512111)

[DANH MỤC BIỂU ĐỒ 5](#_Toc310512112)

[THUẬT NGỮ 5](#_Toc310512113)

[1 Nhập đề 5](#_Toc310512114)

[2 Giới thiệu bài toán kiến trúc 6](#_Toc310512115)

[2.1 Bài toán 1: Tổ chức thông tin và sự linh hoạt của hệ thống so khớp 8](#_Toc310512116)

[2.2 Bài toán 2: Mức độ tương quan giữa những thẻ thông tin bất kỳ 8](#_Toc310512117)

[2.3 Bài toán 3: Phương pháp kết hợp việc so khớp và mức độ tương quan giữa các tag 8](#_Toc310512118)

[3 Lý thuyết về nền tảng kiến trúc 8](#_Toc310512119)

[3.1 Nghiên cứu khoa học 8](#_Toc310512120)

[3.1.1 Taxonomy 8](#_Toc310512121)

[3.1.2 Tag 10](#_Toc310512122)

[3.1.3 Kết hợp so khớp tag và độ tương quan giữa các tag 11](#_Toc310512123)

[3.1.4 Cây quyết định 11](#_Toc310512124)

[3.2 Kiến trúc phần mềm 28](#_Toc310512125)

[4 Giải quyết bài toán kiến trúc 28](#_Toc310512126)

[4.1 Áp dụng nghiên cứu khoa học vào bài toán 28](#_Toc310512127)

[4.2 Áp dụng kiến trúc phần mềm vào bài toán 28](#_Toc310512128)

[4.3 Đánh giá kiến trúc 29](#_Toc310512129)

[5 Áp dụng giải pháp vào bài toán Job Zoom 29](#_Toc310512130)

[5.1 Thực trạng các website tuyển dụng hiện nay 29](#_Toc310512131)

[5.2 Những vấn đề Job Zoom cần giải quyết 29](#_Toc310512132)

[5.2.1 Vấn đề 1: Hỗ trợ người dùng viết CV theo ngành nghề. 29](#_Toc310512133)

[5.2.2 Vấn đề 2: Hỗ trợ nhà tuyển dụng trong việc đăng tải yêu cầu công việc một cách chi tiết và có trọng số theo ngành nghề. 29](#_Toc310512134)

[5.2.3 Vấn đề 3: Matching tool 30](#_Toc310512135)

[5.3 Kết quả mong muốn 30](#_Toc310512136)

[6 Giải quyết những vấn đề trong Job Zoom 30](#_Toc310512137)

[6.1 Khái quát phương pháp giải quyết vấn đề 30](#_Toc310512138)

[6.2 Ứng dụng cây quyết định vào bài toán 31](#_Toc310512139)

[6.3 Phương pháp tiền xử lý dữ liệu khi ứng dụng cây quyết định 32](#_Toc310512140)

[7 Kiến trúc cổng thông tin tìm việc JobZoom 33](#_Toc310512141)

[7.1 Điều kiện ra đời 33](#_Toc310512142)

[7.2 Mô hình kiến trúc khắc phục những điểm yếu của các website tìm việc hiện có 33](#_Toc310512143)

[7.3 Kiến trúc framework 33](#_Toc310512144)

[7.3.1 Kiến trúc tổng quan framework 33](#_Toc310512145)

[7.3.2 Matching tool 33](#_Toc310512146)

[7.3.3 Data mining 34](#_Toc310512147)

[7.4 Mô hình kiến trúc Job Zoom có nhiều ưu điểm vượt trội 34](#_Toc310512148)

[7.4.1 Kiến trúc linh hoạt và khả năng mở rộng của framework 34](#_Toc310512149)

[7.4.2 Khả năng ứng dụng vào những ngành nghề khác nhau 34](#_Toc310512150)

[7.4.3 Khả năng ứng dụng vào những lĩnh vực khác 34](#_Toc310512151)

[7.4.4 Triển khai nhanh chóng 34](#_Toc310512152)

[7.4.5 Đảm bảo hiệu năng hệ thống 34](#_Toc310512153)

[7.5 So sánh Job Zoom với các website tìm việc hiện tại 35](#_Toc310512154)

[8 Đánh giá và hướng phát triển 35](#_Toc310512155)

[8.1 Những điểm làm được 35](#_Toc310512156)

[8.2 Những điểm hạn chế 35](#_Toc310512157)

[8.3 Hướng phát triển 35](#_Toc310512158)

[8.3.1 Xác thực độ tin cậy của CV và yêu cầu tuyển dụng 35](#_Toc310512159)

[8.3.2 Phát triển Semantic web 35](#_Toc310512160)

[8.3.3 Đánh trọng số cho các thuộc tính dựa vào độ sâu của taxonomy 35](#_Toc310512161)

[8.3.4 Phân cụm dữ liệu 35](#_Toc310512162)

[8.3.5 Phỏng vấn trực tuyến 35](#_Toc310512163)

[8.3.6 Đưa thông tin khách quan để training cho chương trình 35](#_Toc310512164)

[8.3.7 Thu thập thông tin việc làm tự động 35](#_Toc310512165)

[TÀI LIỆU THAM KHẢO 37](#_Toc310512166)

**DANH MỤC HÌNH ẢNH**

[Hình 1. Ví dụ về cây quyết định 11](#_Toc310172280)

[Hình 2. Phân chia bảng D thành những tập con Si 16](#_Toc310172281)

[Hình 3. Phân chia Si dựa vào phân lớp 17](#_Toc310172282)

[Hình 4. Độ lợi thông tin thuộc tính Travel cost/Km 18](#_Toc310172283)

[Hình 5. Độ lợi thông tin các thuộc tính còn lại trong Bảng 1 20](#_Toc310172284)

[Hình 6. Node gốc của cây quyết định sau lần lặp đầu tiên 21](#_Toc310172285)

[Hình 7. Bảng D được phân chia sau lần lặp đầu tiên 21](#_Toc310172286)

[Hình 8. Cây quyết định sau lần lặp đầu tiên 22](#_Toc310172287)

[Hình 9. Dữ liệu cho lần phân lớp thứ 2 22](#_Toc310172288)

[Hình 10. Tính Impurity degree cho lần phân lớp thứ 2 23](#_Toc310172289)

[Hình 11. Tính Impurity degree các thuộc tính cho lần phân lớp thứ 2 23](#_Toc310172290)

[Hình 12. Bảng dữ liệu sau khi chia theo phân lớp Gender 24](#_Toc310172291)

[Hình 13. Cây quyết định sau lần phân lớp thứ 2 24](#_Toc310172292)

[Hình 14. Cây quyết định đầy đủ sau 3 lần phân lớp 25](#_Toc310172293)

[Hình 15. Ví dụ về một taxonomy đơn giản 30](#_Toc310172294)

**DANH MỤC BẢNG**

[Bảng 1. Ví dụ bảng dữ liệu lựa chọn phương tiện di chuyển 10](#_Toc310172295)

[Bảng 2. Ví dụ về bảng dữ liệu cần dự đoán phương tiện di chuyển 1 11](#_Toc310172296)

[Bảng 3. Ví dụ về bảng dữ liệu cần dự đoán phương tiện di chuyển 2 12](#_Toc310172297)

[Bảng 4. Lợi ích khi chia bảng D theo thuộc tính “Travel cost/km” 19](#_Toc310172298)

[Bảng 5. Kết quả độ lợi thông tin sau khi phân chia bảng D theo từng thuộc tính 21](#_Toc310172299)

[Bảng 6. Bảng dữ liệu cho lần phân lớp thứ 3 25](#_Toc310172300)

[Bảng 7. Dữ liệu của các thông tin tuyển dụng trước khi thực hiện Pivot trên cột Tag 28](#_Toc310172301)

[Bảng 8. Kết quả sau khi Pivot Transformation 28](#_Toc310172302)

**DANH MỤC BIỂU ĐỒ**

[Biểu đồ 1. Mối liên hệ giữa giá trị Entropy cực đại và số lượng phân lớp 14](#_Toc310172303)

[Biểu đồ 2. Mối liên hệ giữa giá trị Gini index cực đại và số lượng phân lớp 15](#_Toc310172304)

**THUẬT NGỮ**

|  |  |  |
| --- | --- | --- |
| STT | Thuật ngữ | Giải thích |
|  | Tag |  |
|  | Email |  |
|  | Forum |  |
|  | Blog |  |
|  | Social Network |  |
|  | Taxonomy |  |
|  | CV |  |
|  | Resume |  |
|  | Semantic web |  |
|  | Framework |  |
|  | Training set |  |
|  | Series of rules |  |
|  | Attribute |  |
|  | Class |  |
|  | Kiểu dữ liệu Binary |  |
|  | Kiểu dữ liệu Nominal |  |
|  | Kiểu dữ liệu Ordinal |  |
|  | Kiểu dữ liệu Continuos |  |
|  | Pivot table |  |

# Nhập đề

Các kiến trúc phần mềm ứng dụng hiện nay đều đi nhiều về giao diện, giao tiếp (communication) giữa người dùng, những kiến trúc dành riêng giải quyết bài toán nghiệp vụ ít được phổ biến. Bên cạnh đó, hệ thống kiến trúc hiện tại đều rời rạc và không thống nhất, những ứng dụng tìm việc, tìm bạn, tìm mặt hàng, … có chức năng tương đối giống nhau - người dùng đăng thông tin, tìm kiếm và so khớp - nhưng những ứng dụng này chưa có kiến trúc chung. Từ nhu cầu thông tin nhiều, đa dạng, cần so khớp nhiều và nhanh, nhóm chúng tôi muốn xây dựng một kiến trúc phần mềm giúp người dùng đăng tải và so khớp thông tin một cách dễ dàng, nhanh chóng và hiệu quả hơn.

Trong khuôn khổ thực hiện đề tài, chúng tôi muốn cung cấp những giải pháp nhằm xây dựng kiến trúc nói trên, đồng thời áp dụng giải pháp kiến trúc này vào lĩnh vực tuyển dụng. Chúng tôi tạm gọi kiến trúc được áp dụng vào lĩnh vực tuyển dụng này là Job Zoom framework. Từ việc áp dụng vào một lĩnh vực cụ thể đó, chúng tôi sẽ tiến hành đánh giá giải pháp của chúng tôi đưa ra liệu có hợp lý và giải quyết hiệu quả các vấn đề đưa ra hay không.

Phương pháp thực hiện của chúng tôi là phân tích nhu cầu người dùng, các hệ thống website có sự so khớp thông tin, đặc biệt các trang tìm việc làm trong và ngoài nước, cùng với việc thu thập sưu liệu các thông tin hiện có trên internet, các mẫu thiết kế và kiến trúc cơ bản, lý thuyết khai thác dữ liệu và phương pháp phân loại động (cụ thể là cây quyết định và taxonomy) để đánh giá điểm mạnh, điểm yếu của các hệ thống hiện có, từ đó đưa ra những đưa ra những giải pháp có tính hiệu quả cao cho đề tài.

Mục tiêu của nhóm chúng tôi khi thực hiện đề tài “**xây dựng kiến trúc hệ thống cổng thông tin tìm việc”** là cung cấp giải pháp kiến trúc nghiệp vụ linh hoạt giúp người dùng đăng tải thông tin và so khớp, cung cấp một framework đảm bảo tính linh hoạt khi áp dụng trong nhiều hệ thống ngành nghề khác nhau, hỗ trợ người tìm việc hoàn thiện CV tìm việc của mình và hỗ trợ nhà tuyển dụng trong việc đánh giá ứng viên. Đồng thời, doanh nghiệp cung cấp website tìm việc có thể dễ dàng xây dựng hoặc hoàn thiện website của mình dựa trên Job Zoom framework.

Với những ưu điểm của kiến trúc nghiệp vụ đăng thông tin và so khớp nói chung và Job Zoom framework nói riêng, chúng tôi hy vọng đề tài sẽ được ứng dụng và triển khai rộng rãi trong các website so khớp, đánh giá cũng như các website tìm việc trực tuyến, góp phần nâng cao hiệu quả và tiện ích cho người dùng đưa ra những quyết định hiệu quả và kịp thời.

# Giới thiệu bài toán kiến trúc

Ngày nay, Internet mang lại rất nhiều tiện ích hữu dụng cho người sử dụng, một trong các tiện ích phổ thông của Internet là hệ thống thư điện tử (email), trò chuyện trực tuyến (chat), máy truy tìm dữ liệu (search engine) và nhiều dịch vụ tiện ích khác. Những hệ thống này cung cấp một khối lượng thông tin và dịch vụ khổng lồ trên Internet. Đã 29 năm trôi qua kể từ ngày 30/8/1982[[1]](#footnote-1), ngày hệ thống liên lạc bằng thư điện tử (email) được cấp bản quyền, có thể nói đây là một phương tiện thông tin rất nhanh, vận tốc truyền thư điện tử chỉ vài giây đến vài phút và chi phí rất nhỏ không đáng kể so với gửi qua đường bưu điện. Việc gửi email cho một nhóm thật sự trở nên khó khăn khi phải nhập danh sách người nhận gây mất thời gian, việc email bị nhận nhầm là thư rác sẽ cao. Để giải quyết vấn đề này, hộp thư nhóm (group) đã ra đời nhằm giải quyết được khuyết điểm của email, việc gửi email cho một nhóm trở nên dễ dàng hơn. Chưa dừng lại ở đó, diễn đàn (forum) ra đời nhằm bổ sung những thiếu sót của group – việc trao đổi thông tin giữa một nhóm người được thực hiện một cách dễ dàng và nhanh chóng hơn, những bài viết cũng được phân chia thành từng chủ đề cụ thể và rõ ràng hơn. Tuy nhiên, việc phân loại bài viết trên forum cũng không phải là vấn đề đơn giản, một bài viết trong forum chỉ thuộc về duy nhất một chủ đề, do đó, khi một bài viết thuộc về nhiều chủ đề khác nhau, khi chúng ta đăng bài viết trong chủ đề này, thì khi vào chủ đề kia lại không thấy và ngược lại. Để giải quyết vấn đề này, thẻ (tag) đã ra đời. Việc ra đời của tag cùng với blog và mạng xã hội đã đánh dấu bước ngoặc phát triển mới của internet. Một bài viết sẽ được tag những tag liên quan, sự liên kết các tag này góp phần giải quyết cho việc phân loại bài viết trở nên dễ dàng hơn. Tag gợi ý cho chúng tôi thực hiện một hệ thống, mà trên đó *các thực thể khác nhau được định nghĩa thông qua tag, những tag được liên kết với nhau tạo thành một cây yêu cầu hay cây thông tin giúp dễ dàng phân loại và đánh giá đối tượng trở nên dễ dàng hơn*. Chẳng hạn khi bạn muốn mua một cái áo, bạn sẽ miêu tả những chi tiết mà bạn cần chẳng hạn như: sơ mi, tay dài, có khuyu, màu xanh,… bên cạnh đó, các công ty dệt may cũng sẽ mô tả sản phẩm của mình dưới dạng tag, việc so sánh giữa cây yêu cầu của người tiêu dùng và cây thông tin về sản phẩm của công ty dệt may sẽ gợi ý cho người tiêu dùng mua sản phẩm mình yêu thích và nhà sản xuất bán được sản phẩm.

Các kiến trúc phần mềm ứng dụng hiện nay đều đi nhiều về giao diện, giao tiếp (communication) giữa người dùng, những kiến trúc dành riêng giải quyết bài toán nghiệp vụ ít được phổ biến. Bên cạnh đó, hệ thống kiến trúc rời rạc và không thống nhất, những ứng dụng tìm việc, tìm bạn, tìm mặt hàng, … có chức năng tương đối giống nhau - người dùng đăng thông tin, tìm kiếm và so khớp - nhưng những ứng dụng này chưa có kiến trúc chung. Từ ***nhu cầu thông tin nhiều, đa dạng, cần so khớp nhiều và nhanh***, chúng tôi nghĩ về một kiến trúc phần mềm giúp người dùng đăng tải và so khớp thông tin một cách dễ dàng, nhanh chóng và hiệu quả. Kiến trúc này phải giải quyết được những bài toán sau:

## Bài toán 1: Tổ chức thông tin và sự linh hoạt của hệ thống so khớp

Sự phát triển của mạng xã hội, …

Phát sinh từ nhu cầu so khớp thông tin ngày càng nhiều: tìm việc, tìm bạn, tìm mặt hàng,… kiến trúc phần mềm của chúng tôi phải đáp ứng được sự linh hoạt của hệ thống so khớp đồng thời thông tin về một thực thể phải được thể hiện và phân loại một cách rõ ràng dưới dạng cây, dễ dàng trong việc tìm kiếm và đánh giá mức độ so khớp.

To chuc thong tin co the so sanh ma van dam bao kha nang matching

## Bài toán 2: Mức độ tương quan giữa những thẻ thông tin bất kỳ

Một số tag có mức độ tương quan giữa những tag khác trong hệ thống (ví dụ: cây xanh và thực vật có một mức độ tương quan với nhau), kiến trúc phần mềm phải hỗ trợ cho việc mô tả, đánh giá mức độ tương quan ấy.

Ho tro so khop, lay gia tri danh gia va ket noi su dung no, he thong co the lay muc do tuong quan giua 2 tag bat ky, cung cap thong tin cho he thong so khopyt

## Bài toán 3: Phương pháp kết hợp việc so khớp và mức độ tương quan giữa các tag

Kiến trúc phần mềm cung cấp thông tin về việc so khớp cho người dùng quyết định. Thông tin về việc so khớp sẽ được thể hiện thông qua những con số giúp người dùng có thể đưa ra những quyết định một cách hiệu quả và nhanh chóng.

# Lý thuyết về nền tảng kiến trúc

## Nghiên cứu khoa học

### Taxonomy

#### Tổng quan về taxonomy

Vào những năm 90 của thế kỷ XX, khái niệm taxonomy được sử dụng trong nhiều lĩnh vực khác nhau như khoa học xã hội, tâm lý học và công nghệ thông tin… để thiết lập sử trùng hợp giữa thuật ngữ của người sử dụng và thuật ngữ của hệ thống. Các chuyên gia đầu tiên phát triển cấu trúc hệ thống web đã dùng thuật ngữ taxonomy để nói đến việc tổ chức nội dung các trang web. Kể từ đó, khái niệm taxonomy được sử dụng rộng rãi với mục đích này.

Do được sử dụng trong nhiều lĩnh vực khác nhau, nên cũng có nhiều định nghĩ về taxonomy. Từ năm 2000 đến 2005 có hơn 36 định nghĩa khác nhau, dựa theo tài liệu của Hypertext[[2]](#footnote-2) chúng tôi định nghĩa taxonomy như sau:

***Taxonomy là sự phân loại toàn bộ thông tin trong một hế thống có phân cấp, sự phân loại này theo một mối quan hệ có trước của các thực thể trong thế giới thực mà nó biểu diễn.***

Một taxonomy thường được mô tả với gốc ở trên cùng, mỗi nút của taxonomy – bao gồm cả gốc – là một thực thể thông tin đại diện cho một thực thể trong thế giới thực. Giữa các nút trong taxonomy có một mối quan hệ đặc biệt gọi là ***is subclassification of*** nếu hướng liên kết từ nút con lên nút cha hoặc là ***is superclassification of*** nếu hướng liên kết từ nút cha xuống nút con. Đôi khi những quan hệ này được xác định một cách chặt chẽ hơn là ***is subclass of*** hoặc ***is superclass of***, nếu thực thể thông tin là một lớp đối tượng.

Ví dụ sau đây mô tả một taxonomy đơn giản gồm lớp Person, lớp con của nó là Employee, Manager; Lớp cha của Person là Agent. Khi đi lên từ gốc của taxonomy, các thực thể khái quát hơn. Khi đi xuống những lá ở cuối, thực thể xác định rõ ràng hơn. Ví dụ, Agent chung chung hơn Person, Employee cụ thể hơn Person.



Hình 15. Ví dụ về một taxonomy đơn giản

Taxonomy thể hiện ưu điểm trong việc phân lớp thực thể thông tin theo ngữ nghĩa, chúng thiết lập một quan hệ ngữ nghĩa đơn giản để phân biệt giữa các đối tượng trong một miền thông tin.

Taxonomy đóng vai trò rất quan trọng trong việc tổ chức thông tin và tổ chức tri thức. Nó được sử dụng chủ yếu để giúp cho việc tìm kiếm và duyệt thông tin thuận lợi và nhanh chóng hơn, đặc biệt khi chúng ta chỉ có những thông tin chung chung về vấn đề cần tìm kiếm. Khi tìm kiếm trên Internet, nếu sử dụng từ khoá để tìm kiếm thông tin, kết quả trả về có thể từ vài nghìn đến vài chục nghìn tài liệu về các chủ đề khác nhau. Sử dụng taxonomy để tìm kiếm và duyệt thông tin sẽ tiết kiệm được rất nhiều thời gian cho người dùng để tìm được thông tin cần thiết. Đồng thời, taxonomy cho phép các máy tìm kiếm và các ứng dụng có thể dễ dàng tìm được các thực thể thông tin nhanh và chính xác hơn nhiều.

#### Tag

**“Tag** là một từ khóa không có thứ tự hay một thuật ngữ nhằm chỉ một mẫu thông tin (như một bookmark trên internet, hình ảnh kỹ thuật số, hoặc một tệp tin máy tính). Loại dữ liệu biến đổi này giúp miêu tả một mục tin và cho phép người sử dụng tìm lại mục đó bằng cách trình duyệt hay dò tìm. ”[[3]](#footnote-3)

### Độ tương quan giữa các tag

### So khớp tag

### Kết hợp so khớp tag và độ tương quan giữa các tag

### Cây quyết định

#### Giới thiệu về cây quyết định

* Cây quyết định là một cây phân cấp có cấu trúc.
* Dùng để phân lớp đối tượng dựa vào dãy các luật (series of rules), các luật này được sinh ra từ tập dữ liệu (training set).
* Các thuộc tính phân lớp thường có kiểu dữ liệu là binary, nominal, ordinal, continuos.
* Ví dụ “Phương tiện di chuyển”: Cho tập dữ liệu (training set) như sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Thuộc tính (Attribute) | | | | Thuộc tính phân lớp (Class) |
| Gender | Car ownership | Travel cost ($/km) | Income level | Transportation mode |
| Male | 0 | Cheap | Low | Bus |
| Male | 1 | Cheap | Medium | Bus |
| Female | 1 | Cheap | Medium | Train |
| Female | 0 | Cheap | Low | Bus |
| Male | 1 | Cheap | Medium | Bus |
| Male | 0 | Standard | Medium | Train |
| Female | 1 | Standard | Medium | Train |
| Female | 1 | Expensive | High | Car |
| Male | 2 | Expensive | Medium | Car |
| Female | 2 | Expensive | High | Car |

Bảng 1. Ví dụ bảng dữ liệu lựa chọn phương tiện di chuyển

* Tập dữ liệu trên mô tả 10 đối tượng, mỗi đối tượng được miêu tả bằng 4 thuộc tính là “Gender” (kiểu dữ liệu binary), “Car ownership” (quantitative interger), “Travel cost/km” (ordinal), “Income level” (ordinal) và thuộc tính phân loại – category attribute – “Transportation mode” (ordinal).
* Từ tập dữ liệu trên, chúng ta có thể tạo ra cây quyết định như sau:



Hình 1. Ví dụ về cây quyết định

* Trong cây quyết định trên, thuộc tính “Income level” không xuất hiện trong cây; vì dựa vào Bảng 1 (trang 8), thuộc tính “Travel cost/Km” sẽ sinh ra cây quyết định dùng để phân lớp tốt hơn “Income level”

#### Sử dụng cây quyết định để dự đoán lớp các dữ liệu chưa biết

* Mục đích chính của cây quyết định là dùng để xác định lớp hay nói khác đi là dự đoán lớp của các dữ liệu chưa biết dựa vào cây quyết định được sinh ra từ tập dữ liệu đào tạo (training data)
* Ví dụ: dựa vào ví dụ ở phần 3.1.1 Giới thiệu về cây quyết định (trang 8). Cho tập dữ liệu cần dự đoán sau đây:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Gender | Car ownership | Travel cost ($/km) | Income level | Transportation mode |
| Ngoc | Female | 1 | Cheap | High | ? |
| Hieu | Male | 0 | Standard | High | ? |
| Phuc | Male | 1 | Cheap | Medium | ? |
| Nguyen | Male | 2 | Expensive | High | ? |

Bảng 2. Ví dụ về bảng dữ liệu cần dự đoán phương tiện di chuyển 1

* Dựa vào cây quyết định (Hình 1, trang 9), cây quyết định sẽ được duyệt từ nút gốc “Travel cost/km”, dãy các luật sau sẽ được sinh ra:
* Nếu “Travel cost/Km” là **Expensive** thì người đó sẽ chọn phương tiện di chuyển là **car**.
* Nếu “Travel cost/Km” là **Standard** thì người đó sẽ chọn phương tiện di chuyển là **train**
* Nếu “Travel cost/Km” là **Cheap**, chúng ta sẽ xem xét thuộc tính “Gender”:
  + Nếu “Gender” là **Male**, người đó sẽ chọn phương tiện là **bus**
  + Nếu “Gender là **Female**, thì xem người đó sở hữu bao nhiêu xe hơi (thuộc tính “Car ownership”). Nếu số xe sở hữu là **0**, thì người đó sẽ chọn phương tiện di chuyển là **bus**; ngược lại, nếu số xe sở hữu lớn hơn hay bằng **1**, thì người đó sẽ chọn phương tiện di chuyển là **train**.
* Bảng 2 sẽ được dự đoán như sau:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Gender | Car ownership | Travel cost ($/km) | Income level | Transportation mode |
| Ngoc | Female | 1 | Cheap | High | Train |
| Hieu | Male | 0 | Standard | High | Train |
| Phuc | Male | 1 | Cheap | Medium | Bus |
| Nguyen | Male | 2 | Expensive | High | Car |

Bảng 3. Ví dụ về bảng dữ liệu cần dự đoán phương tiện di chuyển 2

* **Một số chú ý khi sử dụng cây quyết định:**
* Phụ thuộc rất nhiều vào training data, tập dữ liệu training data càng lớn thì cây quyết định sẽ đáng tin cậy hơn.
* Không thể nói cây quyết định được sinh ra từ cây quyết định trên là tập luật tốt nhất
* Có nhiều thuật toán phân lớp: ID3, J48, C4.5, C5, CART (Classification and Regression Tree), … Việc lựa chọn thuật toán phụ thuốc vào rất nhiều yếu tố, trong đó yếu tố cấu trúc dữ liệu ảnh hưởng rất nhiều đến kết quả của thuật toán. Chẳn hạn, thuật toán ID3 và CART hiệu quả cho việc phân lớp đối với các dữ liệu số (quantitative value), trong khi đó, thuật toán J48, C4.5 có hiệu quả hơn đối với dữ liệu Qualititive value (ordinal, binary, nominal)

#### Một số độ đo thông dụng

* Cho bảng dữ liệu bao gồm các thuộc tính và thuộc tính phân lớp, chúng ta có thể đo được tính đồng nhất hay không đồng nhất thông qua thuộc tính phân lớp. Bảng dữ liệu có tính đồng nhất nếu nó có duy nhất một phân lớp. Ngược lại, nếu nó có nhiều phân lớp khác nhau, thì bảng dữ liệu có tính không đồng nhất hay tính pha trộn. Chúng ta có thể đo được mức độ pha trộn (Impurity Degree); Entropy, độ đo Gini và classification error là những cách tính mức độ pha trộn thông dụng nhất.

*Pj: xác suất xảy ra phân lớp j*

* Ví dụ: Chúng ta cùng nhìn lại Bảng 1. Ví dụ bảng dữ liệu lựa chọn phương tiện di chuyển. “Transportation mode” có 3 nhóm Bus, Car và Train. Bảng 1 có 10 dòng dữ liệu, trong đó “Transportation mode” có 4 buses, 3 cars, 3 trains (4B, 3C, 3T).
* Xác suất để xảy ra cho từng phân lớp là:

##### Entropy

Khi bảng dữ liệu có duy nhất 1 phân lớp thì Entropy sẽ bằng 0 vì xác suất là 1 và . Entropy sẽ đạt giá trị cực đại khi tất cả thuộc tính phân lớp có xác suất xảy ra bằng nhau. Biểu đồ dưới đây thể hiện giá trị cực đại của Entropy sẽ thay đổi phụ thuộc vào số lượng thuộc tính phân lớp n, trong trường hợp xác xuất tất cả thuộc tính phân lớp thì . Giá trị của Entropy sẽ lớn hơn 1 khi số lượng thuộc tính phân lớp nhiều hơn 2



Biểu đồ 1. Mối liên hệ giữa giá trị Entropy cực đại và số lượng phân lớp

##### Gini index

* Một cách khác để tính “Impurity degree”.
* Chúng ta cùng quay lại ví dụ ở trên
* Khi bảng dữ liệu có duy nhất 1 phân lớp thì Gini index sẽ bằng 0 vì xác suất bằng 1 và . Cũng giống như Entropy, Gini index sẽ đạt giá trí cực đại khi tất cả thuộc tính phân lớp có xác suất xảy ra bằng nhau. Biểu đồ dưới đây thể hiển giá trị cực đại của Gini index sẽ khác nhau phụ thuộc vào số lượng phân lớp n, khi xác xuất tất cả thuộc tính phân lớp



Biểu đồ 2. Mối liên hệ giữa giá trị Gini index cực đại và số lượng phân lớp

##### Classification error

* Trong ví dụ trên
* Cũng giống như Entropy và Gini index, Classification error sẽ bằng 0 khi bảng dữ liệu có duy nhất 1 phân lớp vì xác suất bằng 1 và .

#### Nguyên tắc hoạt động của một số thuật toán xây dựng cây quyết định thường dùng

* Các thuật toán xây dựng cây quyết định thường được sử dụng nhất là ID3, C4.5 và CART (classification and regression trees).
* Nhìn chung, các thuật toán xây dựng cây quyết định đều được xây dựng trên quy tắc đệ quy. Ví dụ, Hunt là thuật toán dựng cây quyết định, đệ quy theo nút của cây, bắt đầu từ nút gốc. Mặc dù kết quả đạt được từ thuật toán Hunt không được tối ưu, nhưng đây là một trong những một trong những thuật toán dùng để xây dựng cây quyết định sớm nhất.

##### Lần lặp đầu tiên

* Giả sử, chúng ta có một bảng dữ liệu chứa các thuộc tính và thuộc tính phân lớp, tạm gọi bảng này là D. Từ bảng D, chúng ta sẽ lấy ra từng cột thuộc tính trong bảng để đối chiếu với các giá trị của thuộc tính phân lớp. Nếu chúng ta có p cột dữ liệu, và lấy từng phần tử p là tập con của D, tạm gọi từng tập hợp này là Si. Bảng D là tập hợp những Si và thuộc tính phân lớp.



Hình 2. Phân chia bảng D thành những tập con Si

* Sau khi phân chia bảng dữ liệu D như trên, chúng ta sẽ tính mức độ pha trộn (tham khảo cách tính tại phần 3.1.3 Một số độ đo thông dụng)
* Ví dụ, dựa vào Bảng 1. Ví dụ bảng dữ liệu lựa chọn phương tiện di chuyển. chúng ta có thể tính “Impurity degree” dựa vào thuộc tính phân lớp “Transportation mode”. “Transportation mode” có 4 busses, 3 cars và 3 trains (4B, 3C, 3T)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Thuộc tính (Attribute) | | | | Thuộc tính phân lớp (Class) |
| Gender | Car ownership | Travel cost ($/km) | Income level | Transportation mode |
| Male | 0 | Cheap | Low | Bus |
| Male | 1 | Cheap | Medium | Bus |
| Female | 1 | Cheap | Medium | Train |
| Female | 0 | Cheap | Low | Bus |
| Male | 1 | Cheap | Medium | Bus |
| Male | 0 | Standard | Medium | Train |
| Female | 1 | Standard | Medium | Train |
| Female | 1 | Expensive | High | Car |
| Male | 2 | Expensive | Medium | Car |
| Female | 2 | Expensive | High | Car |

4B, 3C, 3T

Entropy 1.571

Gini index 0.660

Classification index 0.600

* Sau khi phân chia bảng D thành từng tập con Si, chúng ta tiếp tục phân chia Si để tính “Impurity degree” cho từng giá trị thuộc tính trong từng tập Si



Hình 3. Phân chia Si dựa vào phân lớp

* Tiếp tục ví dụ trên, thuộc tính “Travel cost/km” có 3 giá trị, đó là “Cheap”, “Standard” và “Expensive”.



Hình 4. Độ lợi thông tin thuộc tính Travel cost/Km

* **Độ lợi thông tin (Information Gain):**
  + Có sự khác nhau giữa các cách tính mức độ pha trộn “Impurity degree” giữa bảng D và các tập con Si, chúng ta sẽ tiến hành so sánh “Impurity degree” trước và sau khi phân chia bảng thành những tập con (ví dụ như phân chia D thành từng tập con Si). Information gain là một phương pháp để đo lường sự khác nhau này. Chúng ta sẽ so sánh lợi ích khi chia bảng dữ liệu theo các giá trị của thuộc tính, từ đó chọn ra thuộc tính tối ưu để phân chia
* Xét ví dụ trên, bảng D có 3 phân lớp 4B, 3C và 3T và có Entropy là 1.571. Bây giờ chúng ta sẽ sử dụng thuộc tính “Travel cost/km” để chia bảng thành 3 phần:
* “Travel cost/km” có giá trị là Cheap, thuộc tính phân lớp có 4B và 1T

* “Travel cost/km” có giá trị là Standard, thuộc tính phân lớp có 2T, vì lúc này chỉ có duy nhất 1 thuộc tính phân lớp)
* “Travel cost/km” có giá trị là Expensive, thuộc tính phân lớp có 3C, vì lúc này chỉ có duy nhất 1 thuộc tính phân lớp)
* Tương tự chúng ta sẽ tính được độ đo Gini index và Classification error:

|  |  |
| --- | --- |
| **Lợi ích khi chia theo thuộc tính “Travel cost/km” dựa vào** | |
| Entropy | 1.210 |
| Gini index | 0.500 |
| Classification error | 0.500 |

Bảng 4. Lợi ích khi chia bảng D theo thuộc tính “Travel cost/km”

* Chúng ta thực hiện lần lượt cho các thuộc tính còn lại của bảng D: “Gender”, “Car ownership” và “Income level”



Hình 5. Độ lợi thông tin các thuộc tính còn lại trong Bảng 1

* Bảng dưới đây sẽ cho chúng ta thấy độ lợi thông tin cho tất cả 4 thuộc tính trong bảng D. Chúng ta không cần tính “Impurity degree” dựa trên cả 3 độ đo Entropy, Gini index và Classification error, chỉ cần chọn lựa 1 trong 3 độ đo trên.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Kết quả của lần lặp đầu tiên** | | | | |
| **Độ lợi** | **Gender** | **Car ownership** | **Travel cost/Km** | **Income level** |
| Entropy | 0.125 | 0.534 | **1.210** | 0.695 |
| Gini index | 0.060 | 0.207 | **0.500** | 0.293 |
| Classification error | 0.100 | 0.200 | **0.500** | 0.300 |

Bảng 5. Kết quả độ lợi thông tin sau khi phân chia bảng D theo từng thuộc tính

* Sau khi tính được độ lợi thông tin cho từng thuộc tính, chúng ta sẽ lựa chọn thuộc tính có độ lợi thông tin cao nhất để làm node gốc cho cây quyết định. Bảng trên cho thấy, độ lợi thông tin khi phân lớp theo thuộc tính “Travel cost/km” là cao nhất nên chúng ta sẽ chọn “Travel cost/km” là node gốc của cây.



Hình 6. Node gốc của cây quyết định sau lần lặp đầu tiên

* Bảng D sẽ được phân chia theo thuộc tính “Travel cost/km” như sau:



Hình 7. Bảng D được phân chia sau lần lặp đầu tiên



Hình 8. Cây quyết định sau lần lặp đầu tiên

##### Những lần phân lớp tiếp theo

Trong phần này sẽ sử dụng lại ví dụ ở phần 3.1.4.1

###### Lần phân lớp thứ 2

* Sau lần lặp đầu tiên, chúng ta cần cập nhật lại bảng dữ liệu. Khi chúng ta phân chia bảng D theo node gốc “Travel cost/km”, những dòng dữ liệu có giá trị thuộc tính “Travel cost/km” là Expensive và Standard đã là phân lớp thuần khiết, chúng ta không cần sử dụng lại những dòng dữ liệu này để phân lớp. Trong lần lặp này, “Travel cost/km” chỉ còn lại giá trị Cheap, chúng ta sẽ loại bỏ thuộc tính này



Hình 9. Dữ liệu cho lần phân lớp thứ 2

* Tính “Impurity degree” cho bảng chính



Hình 10. Tính Impurity degree cho lần phân lớp thứ 2

* Tính “Impurity degree” cho các thuộc tính của bảng chính



Hình 11. Tính Impurity degree các thuộc tính cho lần phân lớp thứ 2

* Độ lợi thông tin khi phân lớp theo thuộc tính “Gender” lớn nhất. Chúng ta sẽ phân chia bảng dữ liệu như sau:



Hình 12. Bảng dữ liệu sau khi chia theo phân lớp Gender

* Cây quyết định sau lần phân lớp thứ 2:



Hình 13. Cây quyết định sau lần phân lớp thứ 2

###### Lần phân lớp thứ 3

Tương tự sau lần phân lớp thứ 2, bảng dữ liệu cho lần phân lớp thứ 3 như sau:



Bảng 6. Bảng dữ liệu cho lần phân lớp thứ 3

Nếu chia bảng trên theo “Car ownership” hay “Income level” thì bảng dữ liệu được chia đều có duy nhất 1 phân lớp. Chúng ta có thể chia bảng dữ liệu theo 1 trong 2 thuộc tính này.



Hình 14. Cây quyết định đầy đủ sau 3 lần phân lớp

#### Đánh giá cây quyết định trong lĩnh vực khai thác dữ liệu

##### Điểm mạnh

* **Cây quyết định sinh ra các quy tắc hiểu được**

Cây quyết định có thể sinh ra các quy tắc được chuyển đổi sang tiếng Anh hay câu lệnh SQL. Thậm chí đối với những tập dữ liệu lớn làm cho hình dáng cây quyết định lớn và phức tạp, việc duyệt cây cũng rất dễ dàng. Bất cứ một sự phân lớp hay dự đoán nào đều tương đối minh bạch. Đây là ưu điểm nổi bật của cây quyết định.

* **Cây quyết định có thể thực thi trong những lĩnh vực hướng quy tắc**

Cây quyết định là sự lựa chọn hoàn hảo cho những lĩnh vực có quy tắc, từ lĩnh vực di truyền đến các những quá trình công nghiệp chứa các quy tắc ẩn, không rõ ràng (underlying rules), dữ liệu lỗi chưa được tiền xử lý khá phức tạp và tối nghĩa. Cây quyết định là một lựa chọn tối ưu khi cần tìm ra những quy tắc ẩn, không rõ ràng trong kho dữ liệu (data warehouse)

* **Dễ dàng tính toán trong khi phân lớp**

Những thuật toán xây dựng cây quyết định thường tạo ra cây với số phân nhánh thấp và kiểm tra đơn giản tại từng node. Những thuật toán này thường kiểm tra bằng cách so sánh số, xem xét phần tử của một tập hợp hay các phép nối đơn giản, những thao tác này sẽ được chuyển thành các toán hàm logic và số nguyên, đây là những toán hạng thực thi nhanh và chi phí tối ưu. Trong môi trường thương mại, các mô hình dự đoán này được sử dụng để phân lớp hàng triệu, thậm chí hàng tỷ bản ghi, bởi vậy có thể nói đây là một ưu điểm quan trọng của cây quyết định.

* **Cây quyết định xử lý với cả thuộc tính liên tục và thuộc tính rời rạc**

Các thuộc tính liên tục hay rời rạc đều có thể xử lý bằng cây quyết định. Tuy nhiên, thuộc tính liên tục cần nhiều tài nguyên tính toán hơn, được phân chia bằng việc chọn ra một ngưỡng trong tập các giá trị đã được sắp xếp của thuộc tính đó.

* **Thể hiện rõ ràng những thuộc tính quan trọng nhất cho việc dự đoán phân lớp**

Các thuật toán xây dựng cây quyết định chỉ ra những thuộc tính dùng để phân chia tốt nhất tập dữ liệu đào tạo bắt đầu từ node gốc của cây. Chính vì vậy, chúng ta có thể thấy được thuộc tính nào là quan trọng, có mức độ phụ thuộc lớn cho việc dự đoán hay phân lớp

##### Điểm yếu

Mặc dù có những điểm mạnh nổi bật trên, cây quyết định vẫn không tránh khỏi những điểm yếu. Cây quyết định không thích hợp lắm với những bài toán mục tiêu, như là dự đoán giá trị của thuộc tính liên tục: thu nhập, huyết áp, lãi suất ngân hàng… Bên cạnh đó, cây quyết định cũng khó giải quyết với những dữ liệu thời gian liên tục.

* **Cây quyết định dễ xảy ra lỗi khi có nhiều phân lớp**

Một số thuật toán chỉ tao tác với những lớp giá trị nhị phân dạng “có/không” hay “đồng ý/từ chối”. Số khác có thể chỉ định các bản ghi vào một số lớp bất kỳ nhưng dễ xảy ra lỗi khi ứng với một phân lớp có số lượng dữ liệu đào tạo nhỏ.

* **Chi phí tính toán để đào tạo cao**

Điểm này nghe có vẻ mâu thuẩn với điểm mạnh phía trên. Quá trình xây dựng cây quyết định khác đắt về mặt tính toán. Điều này cũng dễ hiểu vì *cây quyết định có nhiều node trong trước khi đi đến node lá cuối cùng*, ứng với từng node, chúng ta cần tính một độ đo (hay tiêu chuẩn phân chia) trên từng thuộc tính. Bên cạnh đó, đối với thuộc tính liên tục, chúng ta còn phải thêm thao tác sắp xếp lại dữ liệu theo thứ tự giá trị của thuộc tính đó. Quá trình này chọn ra những thuộc tính để phân lớp tốt nhất. Một số thuật toán còn sử dụng tổ hợp các thuộc tính kết hợp với nhau có trọng số để phát triển cây. Quá trình cắt tỉa cây chi phí cũng khá cao vì trong quá trình cắt tỉa, nhiều cây con sẽ được tạo ra và so sánh.

## Kiến trúc phần mềm

[Các design pattern dc áp dụng]

API: **Composition** & inheritance

Chuyển tag vào API -> tạo cây…

Tại sao **Composition hiệu quả hơn inheritance**

# Giải quyết bài toán kiến trúc

## Áp dụng nghiên cứu khoa học vào bài toán

Kết hợp decision tree vao hệ thống thông qua sql. Cấu trúc kết quả như thế nào

Các tổ chức và lưu trữ dữ liệu

## Áp dụng kiến trúc phần mềm vào bài toán

## Đánh giá kiến trúc

# Áp dụng giải pháp vào bài toán Job Zoom

## Thực trạng các website tuyển dụng hiện nay

## Những vấn đề Job Zoom cần giải quyết

### Vấn đề 1: Hỗ trợ người dùng viết CV theo ngành nghề.

**Bạn là người tìm việc?** Có thể bạn đã từng gặp khó khăn trong việc viết sơ yếu lý lịch (CV hay còn gọi là resume) để tạo ấn tượng cho nhà tuyển dụng, tiếp thị cho bản thân mình. Bên cạnh đó, mẫu CV sẽ khác nhau đối với từng ngành nghề khác nhau. Để lọt vào danh sách top ten ứng viên của nhà tuyển dụng, bạn cần phải có một sơ yếu lý lịch ngắn gọn, súc tích nhưng cần rõ ràng và đầy đủ về quá trình làm việc, trình độ chuyên môn, học vấn hay các kỹ năng mềm, việc viết CV này thật sự trở nên khó khăn khi bạn vẫn loay hoay chưa biết phải trình bày hay liệt kê những thông tin như thế nào. Những ứng dụng tìm việc hiện tại không thể gợi ý cho bạn tạo một CV chuẩn cho một ngành nghề cụ thể hay nói khác đi là không hỗ trợ người dùng tạo CV tìm việc đầy đủ và rõ ràng đối với ngành nghề mà người tìm việc muốn apply.

Khi người tìm việc muốn tìm một công việc nào đó, bên cạnh những thông tin cá nhân, ứng dụng sẽ gợi ý một CV chuẩn về ngành nghề, người dùng sẽ tạo ra sơ yếu lý lịch cho mình dựa vào các thuộc tính chương trình gợi ý. Ngoài ra, khi cần apply vào một ví trí của một nhà tuyển dụng nào, ứng dụng sẽ gợi ý người dùng những thông tin cần bổ sung cho vị trí tìm việc.

### Vấn đề 2: Hỗ trợ nhà tuyển dụng trong việc đăng tải yêu cầu công việc một cách chi tiết và có trọng số theo ngành nghề.

**Bạn là nhà tuyển dụng?** Các website tìm việc hiện nay, đều cho nhà tuyển dụng đăng thông tin dưới dạng văn bản thô, không đưa ra gợi ý cho bạn về các ứng viên có khả năng đáp ứng yêu cầu công việc dựa vào các trọng số các yêu cầu của bạn đưa ra. Nhà tuyển dụng khi muốn đăng thông tin tuyển dụng, các website tìm việc hiện tại không thể đánh giá mức độ đáp ứng của những ứng viên dựa vào những thông tin nhà tuyển dụng cung cấp, không thể biết ứng viên đáp ứng được bao nhiêu phần trăm những yêu cầu này, nhà tuyển dụng phải tự đánh giá bằng việc đọc những CV của ứng viên sau đó lựa chọn, việc này gây mất nhiều thời gian và chi phí.

Với Job Zoom framework, nhà tuyển dụng có thể đánh giá trọng số các thông tin yêu cầu về công việc. Ứng dụng đánh giá mức độ phù hợp của ứng viên và liệt kê những ứng viên “tiềm năng” dựa vào thông tin của nhà tuyển dụng cung cấp.

### Vấn đề 3: Matching tool

Đối với người tìm việc: Matching tool gợi ý người dùng hoàn thiện, bổ sung những thiếu sót của CV, tăng cơ hội apply vào một vị trí tại một công ty cụ thể.

Đối với nhà tuyển dụng: Matching tool đánh giá ứng viên xin apply vào công việc, hỗ trợ cho nhà tuyển dụng trong việc đánh giá ứng viên

## Kết quả mong muốn

* Xây dựng kiến trúc phần mềm đảm bảo tính linh hoạt khi áp dụng trong nhiều hệ thống ngành nghề khác nhau.
* Kết hợp được với các phương pháp khai thác dữ liệu cơ bản tạo nền tảng cải thiện khả năng thích ứng của hệ thống với từng loại ngành nghề.
* Xây dựng được kiến trúc phần mềm đảm bảo hiệu năng của hệ thống với số lượng người sử dụng lớn

# Giải quyết những vấn đề trong Job Zoom

## Khái quát phương pháp giải quyết vấn đề

[sửa lại – ghi nhận áp dụng giải quyết bài toán phía trên vào đây]

Sau thời gian nghiên cứu và đánh giá một số phương pháp khai thác dữ liệu, chúng tôi thống nhất sử dụng ***cây quyết định*** trong việc giải quyết vấn đề 1, từ đó tạo ra cây thuộc tính những yêu cầu công việc chung nhất ứng với từng ngành nghề cụ thể. Cây quyết định này sẽ được sử dụng, nhằm gợi ý người dùng viết một CV đơn giản, phù hợp với công việc hiện tại mà người tìm việc mong muốn apply. Để đáp ứng điều này, nhóm cần phải phân loại dữ liệu theo ngành nghề và vị trí làm việc, nói khác hơn chính là việc ***liên kết các cây quyết định*** lại với nhau nhằm tạo ra cây quyết định lớn, đảm bảo trong việc khai thác dữ liệu từ cây để gợi ý cho người tìm việc một mẫu CV cơ bản đối với từng ngành nghề cụ thể. Ngoài ra, đối với người tìm việc đã liệt kê những kỹ năng kinh nghiệm làm việc của mình mà chưa biết phải apply vào công việc nào, việc áp dụng cây quyết định này cũng có thể đáp ứng được việc gợi ý công việc cho người dùng apply

Đối với vấn đề 2 và 3, nhóm chúng tôi sử dụng ***taxonomy*** để đăng tải CV cũng như yêu cầu công việc của người dùng và nhà tuyển dụng dưới dạng cây. Việc so sánh những cây CV người tìm việc cung cấp và cây yêu cầu công việc của nhà tuyển dụng, chương trình sẽ đánh giá mức độ đáp ứng của ứng viên dựa vào việc so sánh các cây dữ liệu này. Ngoài ra, taxonomy còn được sử dụng trong việc gợi ý cho người dùng bổ sung những thông tin còn thiếu để hoàn thiện CV khi xin apply vào một vị trí làm việc nào đó tại công ty tuyển dụng.

Việc ứng dụng cây quyết định và taxonomy lại phát sinh vấn đề đảm bảo hiệu năng hệ thống khi ứng dụng cây quyết định và taxonomy… Để giải quyết vấn đề này, Job Zoom framework sẽ tạo view và khai thác dữ liệu tự động trên cụm máy chủ khác, sau khi có kết quả, Job Zoom framework sẽ lưu dữ liệu đã khai thác vào cơ sở dữ liệu chính của chương trình. Quá trình này được thực hiện hoàn toàn độc lập sau một khoảng thời gian hay dữ liệu phát triển lên đến một độ lớn nhất định.

## Ứng dụng cây quyết định vào bài toán

Đối với bài toán của chúng tôi, thuộc tính phân lớp “Is Approve” (thuộc tính xác định ứng viên có được nhà tuyển dụng nhận hay không sau khi apply) đóng vai trò là thuộc tính phân lớp.

Như vậy, khi khai thác dữ liệu, hệ thống sẽ sinh ra rất nhiều cây quyết định theo từng vị trí làm việc, những cây này sẽ được gom lại giúp việc truy vấn kết quả trở nên dễ dàng hơn. Node gốc của cây là lĩnh vực ngành nghề, đối với từng lĩnh vực ngành nghề sẽ có nhiều vị trí công việc khác nhau.

Kết quả khi tạo cây quyết định thể hiện chúng ta thấy mức độ phụ thuộc, tầm quan trọng của các thuộc tính trong việc lựa chọn ứng viên vào một ví trí cụ thể. Những thuộc tính này gợi ý cho người dùng những kỹ năng, kiến thức người tìm việc nên có đối với vị trí làm việc mà họ đang tìm. Như vậy, khi người dùng sử dụng chương trình, chương trình sẽ đặt câu hỏi và liệt kê các thuộc tính (kỹ năng, kinh nghiệm) có mức độ phụ thuộc cao vào xác suất ứng viên được nhận, ứng viên có thể thay đổi CV cho phù hợp với vị trí mình đang cần ứng tuyển. Nếu một trong những thuộc tính mà Job Zoom gợi ý người tìm việc không thể đáp ứng được, thì Job Zoom có thể gợi ý cho ứng viên những thuộc tính khác hoặc ứng viên có thể học tập, rèn luyện để bổ sung kịp thời những thuộc tính đó nhằm tăng xác suất và tính cạnh tranh của họ khi apply vào công việc.

Thông qua cây quyết định và việc gom nhóm cây, bài toán 1 của chúng tôi đã được giải quyết. Tuy nhiên, mức độ giải quyết này đáp ứng cho những ứng viên muốn ứng tuyển vào một vị trí công việc bất kỳ, xác suất của ứng viên được nhận cao khi ứng viên apply vào nhiều công ty tuyển dụng tương ứng với vị trí đó. Việc ứng dụng taxonomy được đề cập sau đây sẽ giải quyết phần còn lại của bài toán 1, khi ứng viên apply vào vị trí công việc tại một công ty cụ thể, thì những thuộc tính nào giúp ứng viên apply vào vị trí đó có xác suất được nhận cao nhất.

## Phương pháp tiền xử lý dữ liệu khi ứng dụng cây quyết định

Để việc khai thác dữ liệu bằng cây quyết định thực hiện dễ dàng hơn, chúng tôi tiến hành xử lý dữ liệu được lưu trữ theo dạng Tagging (Taxonomy) bằng kỹ thuật Pivot Transformation trong SQL Server.

Ví dụ dữ liệu của thông tin tuyển dụng gồm vị trí đăng tuyển (Job Title), tên công ty (Company), và một tập hợp các yêu cầu/ tiêu chí, với mỗi hàng là một tiêu chí cụ thể.

|  |  |  |
| --- | --- | --- |
| **Usage** | **SetKey** | **PivotKey** |
| **Column Name** | **Job Title** | **Tag** |
| **Data**  **Records** | Developer | OOP |
| Developer | Software design |
| Developer | DBMS |
| Developer | .Net Framework |
| Tester | Testing Technique |
| Tester | .Net Framework |
| Tester | Automation Testing |
| Tester | DBMS |

Bảng 7. Dữ liệu của các thông tin tuyển dụng trước khi thực hiện Pivot trên cột Tag

Bảng sau thể hiện kết quả sau khi Pivot Transformation

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Column Name** | **Job Title** | **OOP** | **Software design** | **DBMS** | **.Net Framework** | **Testing Technique** | **Automation Testing** |
| Data  Records | Developer | True | True | True | True | False | False |
| Tester | False | False | True | True | True | True |

Bảng 8. Kết quả sau khi Pivot Transformation

# Kiến trúc cổng thông tin tìm việc JobZoom

## Điều kiện ra đời

Ngày nay, internet đã dần chiếm hữu cuộc sống hiện tại, hết thảy mọi việc đều được “số hoá” cũng là lúc tuyển dụng và người tìm việc đều chọn các website tìm việc làm cầu nối cho mình.

Theo số liệu thống kê của tổ chức Internet Usage World Stats[[4]](#footnote-4), tính đến nay có hơn 2,1 tỉ người trên thế giới sử dụng Internet. Bên cạnh đó, nghiên cứu về người tiêu dùng do Mintel International Group Ltd. (tổ chức quốc tế chuyên nghiên cứu về thị trường và người tiêu dùng) tiến hành cho thấy có đến 30% số người dùng Internet đã tham khảo các website tuyển dụng trực tuyến để tìm kiếm thông tin về việc làm. Chúng ta có thể thấy xu hướng tìm việc trực tuyến đang ngày càng phát triển với tốc độc rất nhanh.

Việt Nam là quốc gia đang phát triển với dân số trên 90 triệu người với trên 28,6 triệu người sử dụng internet chiếm khoảng 31,6% dân số (2011)[[5]](#footnote-5). Các trang web về lao động việc làm ra đời đã ngày càng đem lại nhiều thông tin tổng hợp, đa dạng về công việc, mở ra những cơ hội lớn hơn cho người tìm việc, đồng thời giúp doanh nghiệp giải quyết nhu cầu nhân sự nhanh chóng và hiệu quả hơn. Tuy nhiên, thực trạng chung của các website tìm việc hiện nay chưa ứng dụng triệt để khai thác dữ liệu trong việc việc gợi ý người dùng viết CV hay hỗ trợ doanh nghiệp trong việc đánh giá ứng viên thông qua thông tin ứng viên cung cấp. Để góp phần đáp ứng những yêu cầu thực tiễn này, đề tài “**xây dựng kiến trúc hệ thống cổng thông tin tìm việc**” được ra đời.

JobZoom framework được phát triển dựa theo phần 3 - Nền tảng kiến trúc.

## Mô hình kiến trúc khắc phục những điểm yếu của các website tìm việc hiện có

Job Zoom framework giải quyết những điểm yếu nào của các website tìm việc hiện tại

## Kiến trúc framework

### Kiến trúc tổng quan framework

### Matching tool

+

+ Cách tính điểm

### Data mining

+ Decision tree

+ Đảm bảo hiệu năng hệ thống.

Hệ thống tự động tạo view, khai thác dữ liệu và lưu dữ liệu vào kho để truy xuất nhanh chóng và dễ dàng. Quá trình này được thực hiện độc lập trên một máy chủ khác …

## Mô hình kiến trúc Job Zoom có nhiều ưu điểm vượt trội

### Kiến trúc linh hoạt và khả năng mở rộng của framework

#### Interface minh chứng CV

#### Interface minh chứng yêu cầu công việc

#### Interface semantic web

#### Interface data mining

Tuỳ thuộc vào điều kiện sẽ có cách mining khác nhau

### Khả năng ứng dụng vào những ngành nghề khác nhau

### Khả năng ứng dụng vào những lĩnh vực khác

### Triển khai nhanh chóng

Người dùng không cần thay đổi cấu trúc bảng trong database.

Cách triển khai: …

### Đảm bảo hiệu năng hệ thống

ADO.NET Entity Framework tốc độ so với Linq

## So sánh Job Zoom với các website tìm việc hiện tại

# Đánh giá và hướng phát triển

## Những điểm làm được

## Những điểm hạn chế

Cây đa cấp

## Hướng phát triển

### Xác thực độ tin cậy của CV và yêu cầu tuyển dụng

Các đối tượng tham gia vào những cộng đồng trực tuyến thường là các đối tượng “ảo” và thường không được xác định tính chính xác, tính xác thực của các thông tin đăng tải. Việc xác thực độ tin cậy của CV và yêu cầu tuyển dụng sẽ giúp hệ thống vận hành tốt hơn, tránh rủi ro và tạo uy tín với người sử dụng.

### Phát triển Semantic web

Semantic web của kiến trúc vẫn còn ở mức cơ bản cần được phát triển một cách hoàn thiện hơn.

### Đánh trọng số cho các thuộc tính dựa vào độ sâu của taxonomy

Dựa vào semantic web, những thuộc tính có liên quan với những tag khác, khi matching sẽ được một số điểm tương ứng, số điểm này được dựa vào độ sâu của tag.

Ví dụ:

### Phân cụm dữ liệu

### Phỏng vấn trực tuyến

Hỗ trợ người dùng thêm khâu phỏng vấn trực tuyến

### Đưa thông tin khách quan để training cho chương trình

### Thu thập thông tin việc làm tự động

**TÀI LIỆU THAM KHẢO**

[1] Lior Rokach & Oded Maimon, Data Mining with Decision Trees: Theory and Applications, 2008.

[2]Jamie MacLennan, ZhaoHui Tang & Bogdan Crivat, *Data Mining with Microsoft SQL Server 2008*, Wiley Publishing Inc., 2008

[3] Lan H. Witten, Eibe Frank and Mark A. Hall, *Data Mining: Practical Machine Learning Tools and Techniques - Third Edition*, Elsevier, 2011

[4] Rafael Olivas, *Decision Trees - A Primer forDecision-making Professionals,* 2007

[5] Scott A. Golder & Bernardo A. Huberman, *The Structure of Collaborative Tagging Systems*

[6] Adam Freeman & Steven Sanderson, *Pro ASP.NET MVC 3 Framework – Third Edition*, Apress

[7] Tomislav Piasevoli, *MDX with Microsoft SQL Server 2008 R2 Analysis Services: Cookbook,* Packt Publishing, 2011

[8] Arttennick, *Practical MDX Queries for Microsoft SQL Server Analysis Serivces 2008*, McGraw-Hill Companies

[9] http://bis.net.vn/forums/p/378/661.aspx

[10] Nguyễn Thị Thùy Linh, *Thuật toán phân lớp cây quyết định*, Khóa luận tốt nghiệp đại học, Trường Đại học Công nghệ, 2005.

[11]

[12]

[13]

[14]

[15]

[16]

[17]

[18]

[19]

[20]

[21]

[22]

[23]

[24]

1. Nguồn: http://www.pcworld.com.vn/articles/tieu-dung/song-cong-nghe/2011/08/1227918/email-hanh-trinh-29-nam-lich-su/ [↑](#footnote-ref-1)
2. http://www.hipertext.net/english/pag1011.htm#origenNota2 [↑](#footnote-ref-2)
3. Nguồn: http://vi.wikipedia.org/wiki/Tag [↑](#footnote-ref-3)
4. http://www.internetworldstats.com/stats.htm [↑](#footnote-ref-4)
5. http://www.internetworldstats.com/asia.htm#vn [↑](#footnote-ref-5)