A Social Recommender System For Wines

Undergraduate Project by Peter Chamberlin
BSc Information Systems and Management
Department of Computer Science
Birkbeck, University of London

April 3, 2013

Abstract

This is the paper's abstract . . .

Contents

1 Introduction

Recommender Systems.

Since their origin in the mid-1990s with systems such as Tapestry [6] and GroupLens [9], recommender systems have become ubiquitous on the World Wide Web, being employed by some of the worlds largest online businesses as core parts of their offering.

It is the growth of the Web, which is now ubiquitous itself, that has given companies the ability to draw on unprecedented amounts of data about their users' preferences. At the same time the Web has made it easier than ever to reach their users with tailored suggestions.

Amazon's system of product recommendations using item-to-item collaborative filtering is regarded as a "killer feature"[7], and is one of the defining features of the Amazon brand experience. Amazon state their mission to be, "to delight our customers by allowing them to serendipitously discover great products"[7].

Netflix's movie recommender system *Cinematch* "Netflix Prize" competition

Social Recommender Systems.

In recent years social applications have dominated the Web. Networks like Facebook and Twitter have become massive global businesses as Internet users share more and more of their lives online. In such a context recommender systems are able to look beyond users' relationships to product and services, being able instead to interrogate qualitative data about the relationships between specific people. Facebook describe this as the "social graph".

CITATIONS NEEDED!

These systems, "a class of recommender systems that target the social media domain"[**?**], represent the current state of the art.

Recommending Wines.

Recommender systems for wines are not a new idea, being typical of the kind of item many systems are designed to recommend. Burke developed the VintageExchange FindMe recommender system in 1999[1], and there is at least one patent pending with the WIPO for a wine recommender system[**?**].

As a knowledge-based recommender, Burke's FindMe system, "required approximately one person-month of knowledge engineering effort"[2].

The Tetherless World Wine Agent (TWWA), by Patton and McGuinness[**?**]. The TWWA project is primarily concerned with knowledge representaion and the Semantic Web, presenting a common and collaborative ontology for wine with which users can share wine recommendations across their social networks[11]. The system does not automatically tailor recommendations to users, although this is stated as a target for future work[11].

Service Oriented Architecture.

I have chosen to implement my system as a service, such

Aims and Objectives.

I aim to produce a recommender system for wines which takes advantage of both ratings and tasting notes to suggest the most interesting wines and users. I intend the system to ignore wine attributes, such as grape variety and colour, preferring to make recommendations based on a hybrid of pure collaborative filtering and a content-based filtering approach using user-submitted and expert tasting notes.

Rather than implement a full graphical interface for the system I have chosen to develop an HTTP API.

In doing so I will explore the field of recommender systems,

Why are these systems interesting - Benefits - Challenges

Typical applications... - Movies - Products (i.e. Amazon)

Applications in wine domain - what's the same - what's different

What will this project do? - implement a recommender system for wines - exploration of techniques etc.

2 Literature Review

The term *recommender system* was coined by Resnick and Varian [10] to describe a system that “assists and augments” the “natural social process” of recommendation, with Resnick and Varian stating that they preferred it to the more narrow term “collaborative filtering” used by Goldberg et al. [6] to describe their Tapestry system.

The growth of the World Wide Web has seen recommender systems become a ubiquitous part of everyday life, with companies such as Amazon, Facebook, Twitter and Google using making recommendations to millions of us every day.

. . .

There are several main categories of filtering technique employed in recommender systems. Burke [4] presents five: *collaborative*, *content-based*, *demographic*, *utility-based* and *knowledge-based*. Table 1 details these methods and their properties.

The properties by which Burke, 2002 [4], classifies these systems are *background data*, which exists prior to recommendation, *input data*, the data that has been contributed by the user, and *process*, the method by which recommendations are arrived at by using the *background data* and *input data*.

Burke, 2002 [4], asserts that *collaborative filtering* is probably the most widely used and mature of these types, citing *GroupLens* [9] and *Tapestry* [6] as important examples of such systems, as well as several others.

Collaborative filtering a process whereby a user’s preferences for items are inferred by the comparison of their previous preferences with the preferences of others. In itself this is a fairly straighforward principle, the comparison of two vectors of items and ratings or preferences, but there are a plurality of approaches. Burke, 2002 [4],

Content-based, like *collaborative filtering*, builds up a long term profile of a user’s interests and preferences [4].

Hybrid Systems.

— off the web (?) — on the web...
– What are the methods employed in recommender systems?
– Collaborative Filtering - User-based filtering - Item-based filtering
– Content-Based Filtering - Variants
– Characteristics of the domain

Table 1: Recommendation Techniques, reproduced from Burke, 2002 [4]

Technique	Backgroud	Input	Process
Collaborative	Ratings from U of items in I.	Ratings from u of items in I.	Identify users in U similar to u, and extrapolate from their ratings of i.
Content-based	Features of items in I.	u's ratings of items in I.	Generate a classifier that fits u's rating behaviour and use it on i.
Demographic	Demographic information about U and their ratings of items in I.	Demographic information about u.	Identify users that are demographically similar to u, and extrapolate from their ratings of i.
Utility-based	Features of items in I.	A utility function over items in I that describes u's preferences.	Apply the function to the items and determine i's rank.
Knowledge-based	Features of items in I. Knowledge of how these items meet a user's needs.	A description of u's needs or interests.	Infer a match between i and u's need.

— Cold start problem — Sparsity problem — ... etc.

3 Development Method

The two main aims of my project were to create a recommender system for wines, and to implement an API for that system which might act as a service back end for a web application.

Python

My first enquiries into recommender systems included looking at Segaran's code examples in *Collective Intelligence* [?], where the language he uses for his implementations is Python. Other authors on recommender systems also use Python, such as [?]. Looking into the language further it became apparent that there are many tools available to the Python programmer that are particularly useful for this kind of system, such as the libraries Scipy[?], Numpy[?] and, for natural language processing, NLTK[?].

For the most part my system would suit the stateless, non-persistent nature of a Python web application. The only concern in this regard would be that I would need to recreate objects in memory from scratch with each request rather than persist them as I might using another language, such as using Java with the JPA[?]. Should the lack of persistence prove problematic down the line I looked into the possibility of using a persistence mechanism such as Memcache[?] to serve this purpose, and found that there is wide support for such a solution using Python and Flask[?].

With Python there is also a solid heritage of web application frameworks, such as Django[?] which is widely used for enterprise websites. I felt that Django might be a bit too fully featured for my purposes however [WHY?], and instead decided to use the framework Flask[?], which is developed with small services in mind and would be ideally suited to my purposes.

MySQL

Originally I had envisaged a system backed by a PostgreSQL[?] RDBMS, but having received the Decanter.com data as a MySQL[?] database it did not seem, comparing the two systems, that there would be any significant benefit migrating the data to PostgreSQL. Both are widely used in production, and have similar feature sets. For a short time I considered using a “NoSQL” database such as MongoDB[?] for my project, but decided against such a solution, recognising that such document-oriented systems are not ideal when joining between tables in the way that I would need to for my wines and tasting notes. It seemed that an RDBMS was ideally suited to the purpose, and there was no reason why that shouldn't be MySQL.

GitHub

In order to back-up and version my code I chose to use GitHub[?], which is a web service providing Git version control. I chose to use GitHub for my notes and project files also, so that my project was stored and versioned in its entirety in a private repository on GitHub.

Methodology

It was clear early in the planning for this project that I would be doing a large amount of experimental programming, with the recommendations the system makes being a work in progress.

4 The Sommelier System

Data Cleanup

!!! Priorities: retain as many ratings and notes as possible. Don't worry about attributes.

The data source I have used for my project is the wines database belonging to Decanter.com[?]. The database contains nearly 40,000 professional ratings and tasting notes for wines from as far back as 1986, featuring vintages as far back as 1917.

The original database is highly inconsistent, displaying a mixture of design approaches and a variable quality of data. This is consistent with the fact that the database has been developed over a long period of time by a number of different developers with varying levels of skill, and that wine journalists making entries into the database have taken a number of idiosyncratic approaches to data entry.

Nevertheless I considered there to be a great deal of useful and interesting information in the database, with it to contain usable ratings and/or tastings for over 33,000 wines.

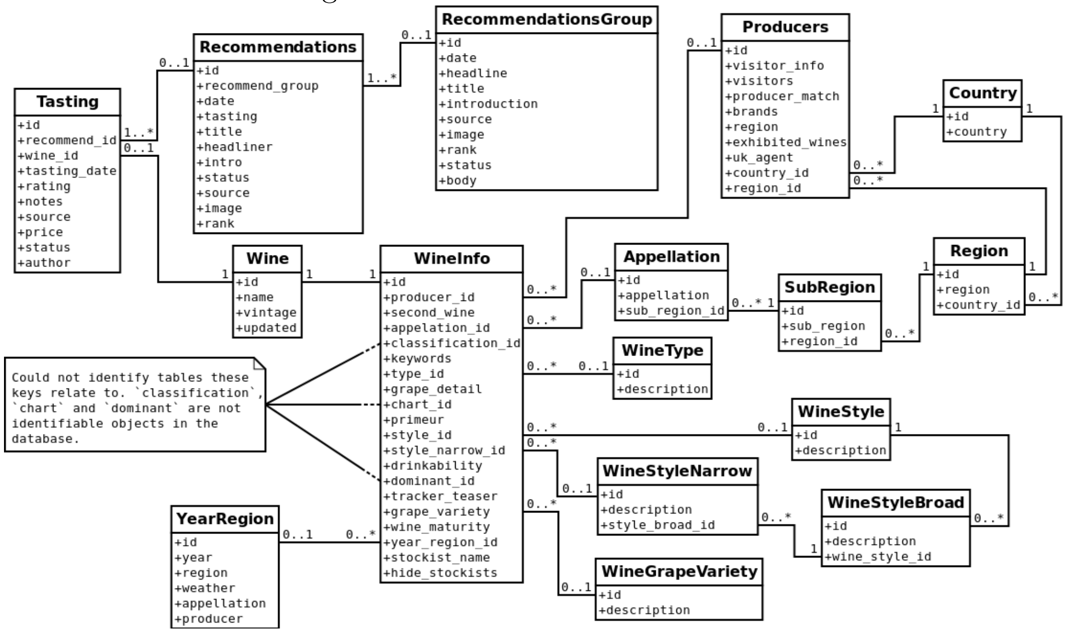
The Decanter Database

In the original database the WineInfo table is a mixture of foreign keys joining to very small tables, such as WineInfo.type_id joining to WineType.id where WineType is a table with only two attributes. This approach, stiving for a high degree of normalisation, contrasts with the fact that the same table also has the attribute second_wine, as a string which only holds data in 450 of the 38762 entries in the table.

Creating The Sommelier Dataset

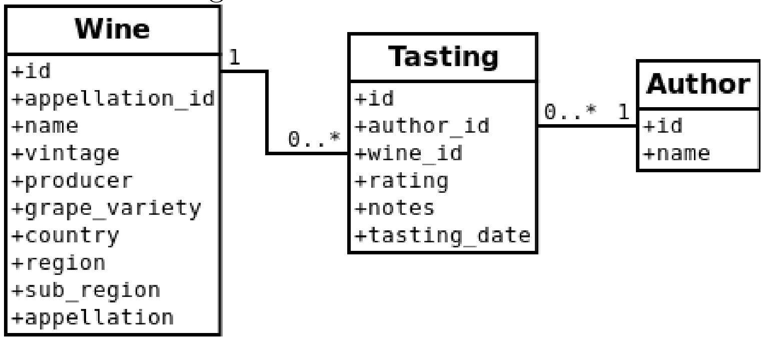
For the new Sommelier database, Figure 2, I decided to denormalise [explanation/citation needed!] the wine data. This enables the data to be queried with fewer joins, maximising the simplicity and execution speed of the queries [citation needed] used for data access. Denormalization makes data integrity more difficult to maintain however, as there are potentially a large number of records to update for any change in a duplicated value. In

Figure 1: Decanter.com Database



Could not identify tables these keys relate to. `classification`, `chart` and `dominant` are not identifiable objects in the database.

Figure 2: Sommelier Database



this case an appellation or sub-region name changing might require thousands of records to be amended. Creating and editing wines is not a requirement of my system though, so for the purposes of this project the wine and tasting data is static and will not be subject to updates. For this reason the duplication of data within the Wine records is not problematic. In a real world setting this issue may need to be revisited.

Much of the data from the original database was disregarded entirely.

The tables WineStyleNarrow and WineStyleBroad contained generic text descriptions for wine ("rich and creamy", "crisp and tangy" etc.). I initially considered this to have potential for migration into tag data which I could reuse as part of my filtering. Unfortunately less than 6435 of the records in WineInfo had non-null values for their style_narrow_id field, and only 3397 of these had corresponding records in the Tasting table. This figure was only around 10% of the number of wines I expected the Sommelier database to contain so I decided that the WineStyle* tables were probably not worthwhile to migrate.

The WineType table was ignored because no wines corresponded to it; no WineInfo.type_id record matched any WineType.id.

The TasterRating and TastingId tables were discarded because they only referenced 1158 of the records in Tasting, too small a proportion of the Tasting table to make it worthwhile migrating them into the dataset.

The Author Problem

The biggest shortcoming of the dataset is that the author of a tasting note is often not recorded. The number of wines with notes and known authors is only 1401, with there being 18 named authors on the system.

Table 2 shows the distribution of tastings amongst authors, only 5 of which have tasted and rated more than 100 wines in the database.

In some cases an author's initials or full name are recorded within the text of a tasting note. I decided that extracting and making use of these was impractical given the time constraints of this project.

DESCRIBE DATA SETS BEFORE AND AFTER

THE SOMMELIER DATASET

Having analysed the dataset and conceived an ideal schema, I needed to decide what the criteria to apply when extracting my new dataset from the source data.

Given that the purpose of the dataset is social recommendations, the first decision I made was to discard any wines without both tasting notes and a rating, whether.

Making Recommendations

Table 2: Authors of tasting notes and ratings

Author	Wines tasted	Wines also tasted by another
Amy Wislocki	28	1
Andrew Jefford	105	38
Beverley Blanning MW	13	-
Carolyn Holmes	1	-
Christelle Guibert	119	9
Clive Coates MW	6	-
David Peppercorn	44	-
Gerald D Boyd	7	-
Harriet Waugh	250	23
James Lawther MW	226	21
John Radford	2	-
Josephine Butchart	24	1
Norm Roby	4	-
Rosemary George MW	6	-
Serena Sutcliffe	31	15
Stephen Brook	19	3
Steven Spurrier	497	53

Wine attributes, vintage, grape variety, region etc., are of limited use for making recommendations. Take a user who has rated a red Bordeaux wine highly. It is not interesting to that user for a system to simply recommend them other highly-rated red Bordeaux wines, which is what a system will do if it looks for items with similar attributes. It is likely that any drinker who enjoys Bordeaux wines will recognise that such wines are of a type, with geography, grape varieties and production processes in common. Thus the user will be capable of “recommending” Bordeaux wines to themselves, and will not have very much difficulty sourcing ones which are highly rated. They do not need a recommender system for that.

It is recommendations in spite of the attributes of the subject item that are of real interest. Recommending a Syrah from Chile’s Colchagua Valley to someone who rated a red Bordeaux wine highly might be of more interest. It is relatively more likely that a user is unaware of the fine Syrah wines from

Table 3: Matrix of authors with wines tasted in common

Author	SS	JL	JB	SB	CG	SS	HW	AJ	AW
Steven Spurrier (SS)	-	6	1	1	7	0	15	30	1
James Lawther MW (JL)	6	-	0	0	0	15	0	0	0
Josephine Butchart (JB)	1	0	-	0	0	0	0	0	0
Stephen Brook (SB)	1	0	0	-	0	0	1	1	0
Christelle Guibert (CG)	7	0	0	0	-	0	1	5	0
Serena Sutcliffe (SS)	0	15	0	0	0	-	0	0	0
Harriet Waugh (HW)	15	0	0	1	1	0	-	10	0
Andrew Jefford (AJ)	30	0	0	1	5	0	10	-	0
Amy Wislocki (AW)	1	0	0	0	0	0	0	0	-

that region, and that makes it a much more interesting recommendation; potentially a good one.

Similarly a recommendation of another Bordeaux red wine might be good, as long as it were possible to establish a commonality of appeal for that particular drinker other than that the wine is similar in attributes. For example there are many delicious Pauillacs from 2005, made of the same grape varieties in the same manner; finding the one which is most interesting to a given user is not aided by looking at its grape variety or appellation as they are identical. The appeal and interest of a wine recommendation lies in qualities beyond the item's attribute profile.

So, by disregarding the attributes of the wines it may be possible to make more interesting recommendations.

A corrollary benefit of this disregard for wine attributes is that I am able to make use of far more of the Decanter.com data, many of the wines within which have incomplete attributes.

In Chapter 2 of Collective Intelligence[**?**], Segaran details basic methods for user- and item-based collaborative filtering. Following the guidelines from this chapter I recreated Segaran's recommendation methods and applied them to my dataset.

TODO Comparison of Segaran and Movielens datasets and Sommelier - why the results no good using straightforward methods?

TODO appendix of data

1. select a.name, count(t.id) from author a join tasting t on a.id = t.author_id where 0 ¡ (select count(*) from tasting t2 where t2.winei_id = t .wine_id and t2.author_id ¡¿ 0 and t2.author_id ¡¿ t.author_id) group by a.id; +——————————+————————+ — name — count(t.id) — +——————————+————————+ — Steven Spurrier — 53 — — James Lawther MW — 21 — — Josephine Butchart — 1 — — Stephen Brook — 3 — — Christelle Guibert — 9 — — Serena Sutcliffe — 15 — — Harriet Waugh — 23 — — Andrew Jefford — 38 — +——————————+————————+

2. select a.name, count(t.id) from author a join tasting t on a.id = t.author_id where 0 ¡ (select count(*) from tasting t2 where t2.wine_id = t.wine_id and t2.author_id = N) group by a.id;

5 Testing and Evaluation

How well does the system work? Details of testing and evaluation of the system…

6 Conclusion

Was the project successful?

7 Review

Review / reflections of the project on a personal level. What has been achieved? What were the problems, and how were they overcome?

Lessons learnt…- Data cleanup very time consuming - Literature vast -¿ plural techniques for recommendation: very difficult to work out what strategy is best for given situation.

References

[1] Burke, R., *The Wasabi Personal Shopper: A Case-Based Recommender System*, 1999. Submitted to the 11th Annual Conference on Innovative Applications of Artificial Intelligence.

[2] Burke, R., *Integrating Knowledge-Based and Collaborative-Filtering Recommender Systems*, 1999. In: Artificial Intelligence for Electronic Commerce: Papers from the AAAI Workshop (AAAI Technical Report WS-99-0 1), pp.69-72.

[3] Burke, R., *Knowledge-Based Recommender Systems*, Encyclopedia of Library and Information Systems, 2000. Marcel Dekker.

[4] Burke, R., *Hybrid Recommender Systems: Survey and Experiments*, User Modeling and User-Adapted Interaction, Volume 12 Issue 4, November 2002, Pages 331 - 370. Kluwer Academic Publishers: Hingham, MA, USA

[5] Debnath, Souvik and Ganguly, Niloy and Mitra, Pabitra, *Feature weighting in content based recommendation system using social network analysis*, Proceedings of the 17th international conference on World Wide Web, WWW '08, 2008, Beijing, China, Pages 1041 - 1042. ACM: New York, NY, USA,

[6] Goldberg, D. Nichols, D., Oki, B. M., and Terry, D., *Using collaborative filtering to weave an information tapestry*, Commun. ACM 35, 12 (Dec. 1992), 61–70.

[7] Mangalindan, J. P., *Amazon's Recommendation Secret*, July 2012. URL: http://tech.fortune.cnn.com/2012/07/30/amazon-5/

[8] Patton, E., McGuinness, D., *Scaling the Wall: Experiences Adapting a Semantic Web Application to Utilize Social Networks on Mobile Devices*, 2010. In: Proceedings of the WebSci10: Extending the Frontiers of Society On-Line, April 26-27th, 2010, Raleigh, NC: US.

[9] Resnick, P., Iacovou, N., Sushak, M., Bergstrom, P., Riedl, J., *GroupLens: An open architecture for collaborative filtering of netnews*, 1994 ACM Conference on Computer Supported Collaborative Work, 1994. Association of Computing Machinery, Chapel Hill, NC.

[10] Resnick, P., Varian, H. R., *Recommender Systems*, 1997. Communications of the ACM, 40 (3), 56-58. Association of Computing Machinery, Chapel Hill, NC.

[11] http://wineagent.tw.rpi.edu/index.php