2015

Geographic Relevance for Travel Search: The 2014-2015 Harvey Mudd College Clinic Project for Expedia, Inc.

Hannah Long
Scripps College

Geographic Relevance for Travel Search:
The 2014-2015 Harvey Mudd College Clinic Project for Expedia, Inc.

By Hannah Long

Submitted to Scripps College
In Partial Fulfillment of
The Degree of Bachelor of Arts

Senior Thesis in Computer Science
May 2015

Table of Contents

Abstract

The purpose of this Clinic project is to help Expedia, Inc. expand the search capabilities it offers to its users. In particular, the goal is to help the company respond to unconstrained search queries by generating a method to associate hotels and regions around the world with the higher-level attributes that describe them, such as "family-friendly" or "culturally-rich." Our team utilized machine-learning algorithms to extract metadata from textual data about hotels and cities. We focused on two machine-learning models: decision trees and Latent Dirichlet Allocation (LDA). The first appeared to be a promising approach, but would require more resources to replicate on the scale Expedia needs. On the other hand, we were able to generate useful results using LDA. We created a website to visualize these results.

Introduction

Expedia is a leading company in the online travel industry, and the Expedia website offers services to book many types of travel arrangements, including hotels and air travel. The website currently uses a constrained search model, meaning that users are very restricted in their search capabilities. For instance, users searching for a hotel are prompted to enter very specific information, including the exact city they want to visit, their dates of travel, and the number of travellers in their party.

The alternative to constrained search is an unconstrained search model, which is used by search engines such as Google. With unconstrained search, users are able to enter free-form text that is less well defined and includes more qualitative terms. Examples of qualitative unconstrained search queries include "best family-friendly vacation spots in Europe" and "culturally-rich cities in South America." Google's unconstrained search model is able to process queries like these and return relevant results. In contrast, a constrained search model does not support these types of queries, because the search engine cannot process qualitative terms like "family-friendly" or "culturally-rich."

The distinction between constrained and unconstrained search is important, because many travellers don't know exactly where they want to go or what type of vacation they want to take when they set out to plan a trip. If they don't know their exact destination, the Expedia website is of little help to such users. Instead, users will often gravitate toward unconstrained search engines to do more exploratory research. Expedia would like to bridge this gap between constrained and unconstrained search by offering exploratory research capabilities and travel booking in one place.

Background

To fully understand this problem, it is necessary to examine the task of unconstrained search in more detail. An unconstrained search begins with a query from the user, such as "best family-friendly vacation spots in Europe." The ultimate goal of this search is to return relevant results to the user. In this case, that would be cities or regions in Europe that are particularly family-friendly, such as Rome and London. As with all search problems, the fundamental task is one of filtering from a large set of data to find the relevant results. In this case, Expedia has a lot of information about travel destinations in Europe; the goal is to infer what portions of this data correspond to regions that are particularly family-friendly.

Thus, the task of unconstrained search involves building associations between high-level concepts, such as "family-friendly", and travel data. These associations must be built for a number of higher-level attributes in which travellers might be interested, including family-friendly, cultural, wine and beach.

Problem Statement

The purpose of this project is to help Expedia respond to unconstrained search queries. In particular, our team was provided with the following problem statement: Generate a method to associate hotels and regions around the world with the qualitative attributes that describe them, such as “family-friendly” or “culturally-rich”.

General Approach and Objectives

Our general approach for this project was to collect external data about geographic destinations and then utilize offline machine learning techniques to extract high-level concepts from this data. Then these associations between qualitative attributes and travel data could be used in real time to filter during the unconstrained search process. See diagram 1 for a detailed overview of this process.

We defined three specific objectives to guide our progress:

1. Collect textual data about hotels and regions from various online travel sources.
2. Research data mining and experiment with various techniques to utilize machine learning for extracting high-level concepts from our data.
3. Report on the benefits and drawbacks of each approach we tried. In addition to a description of the process we developed, we also wanted to create a tool to visualize the results we obtained.

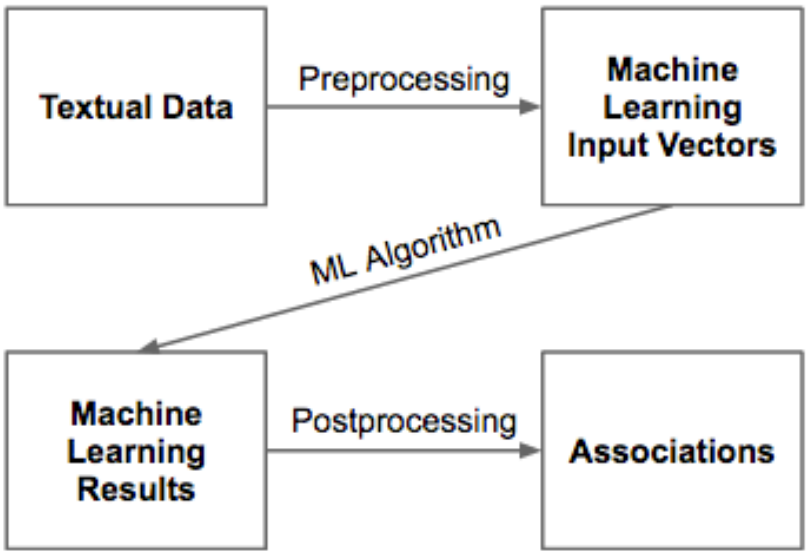

Diagram 1: Our model utilizes machine-learning algorithms to extract high-level associations from data.

Data Collection

The first phase of this project was to collect textual data from various online travel sources. At the suggestion of our liaisons at Expedia, we decided to focus our research on California as an example, and spent a portion of the first semester collecting data about regions and hotels in this area. The task of data collection involved three main components: finding sources, setting up a database framework, and "scraping" the data from various sources, which is the process of extracting data from human-understandable content on a website or in a program.

One of the specific tasks assigned to us by our liaisons at Expedia was exploring potential datasets to determine what types of data are particularly relevant or helpful to this project. For this reason, we spent a lot of time searching for sets of travel data and kept clear records of the results we found. Ultimately, we collected a list of 26 datasets. Although we are under agreement not to divulge this list, each dataset seemed to offer interesting data for this project. Some of them were structured in the form of travel guides, while others featured commercial descriptions and user reviews of hotels, cities and destinations.

Next, we ranked each source based on whether they included certain features, such as professional descriptions of hotels, user reviews of hotels, and lists of amenities. We also considered the issue of ease of navigating each site, as this would greatly influence our ability to easily scrape the data. Ultimately, we settled on three main sources of interest.

The next step in the data collection process was to develop a database infrastructure to store our data. We decided to use the open-source database management

system PostgreSQL[1]. We organized a hierarchical table structure to store both raw HTML[2] pages scraped from websites and parsed information from these pages, such as specific text fields or metadata tags. We wrote the majority of our code for this project in Python[3], so one of our team members devised a database interface to allow us to perform database queries from Python. This made accessing the information in the database much easier and more intuitive.

The final step in the data collection process was scraping the HTML pages from websites and parsing this raw data into a useable format. In general, the pages of interest to us listed information for specific points of interest, such as a given hotel or city. The scraping process involved a significant amount of work writing Python scripts to traverse each site, since we needed a way to automatically locate these pages of interest and extract their URLs. Once we found the URLs, we used a request library built into Python to perform HTTP GET requests, which ask for content from the server. Finally, we needed to parse the raw HTML documents to locate and extract the useful data. To do this, we used the Python library BeautifulSoup[4], which transforms HTML into a traversable tree data structure. From this tree structure, we were able to extract the data fields we wanted.

We completed this scraping and parsing process for each of the three sources of interest. In addition, Expedia provided us with a subset of its own hotel reviews. Here is a summary of the four datasets we collected:

1. 130 professional reviews of hotels.
2. 400 descriptions of cities.
3. 3,000 professional descriptions of hotels.

4. 1,600,000 user reviews of hotels.

Machine Learning Overview

After collecting our data, the next step was to brainstorm methods to extract high-level concepts from this data. Machine learning seemed like an appropriate approach to this problem, because machine-learning algorithms can learn from data. In particular, these types of algorithms define a model that can find similarities within input data or predict information about new data. In the case of this problem, our input is textual data collected about regions and hotels, while the predictions we hope to discover are metadata tags regarding particular categories.

As this is a research project, we wanted to explore a variety of types of machine learning techniques. Machine learning algorithms can be broken down into two main classes: supervised and unsupervised. We experimented with both types of approaches.

Supervised Learning

Supervised machine learning techniques input labeled training data and build a model to predict similar labels for unseen data. For instance, given a set of 1,000 hotel reviews and a tag of "family-friendly" or "not family-friendly" for each one, a supervised machine learning algorithm could develop a model that assigns to a new review a label of either family-friendly or not. That is exactly how we wanted to utilize this type of algorithm.

We decided to focus on decision trees, a particular type of supervised machine learning model (Flach). The decision tree model builds a tree-like structure, wherein each node is a question and each branch represents an answer to that question. Leaves of the tree correspond to labels. For instance, for a tree predicting whether a hotel review expresses family friendliness, a node might ask the question "Does the review contain the word 'child'?" The right branch might correspond to an answer of yes, and so this leaf would label the review as family-friendly. Meanwhile, the left branch could correspond to an answer of no, and that leaf would label the review as not family-friendly.

Decision trees are a powerful tool because they provide a very clear visualization of the factors utilized by the model to make predictions. However, one drawback is that, like all supervised learning models, training decision trees requires labeled data, which can be difficult to obtain in large quantities.

We initially ran a decision tree algorithm on our two datasets of professional descriptions of hotels, since we were able to obtain labels for both. For the set of 130 hotel reviews, we read each review and hand-labeled them as family-friendly or not. Although this is not a very scalable approach, it was manageable for this small set of

reviews. We trained a decision tree on two-thirds of these reviews, and then tested on the rest. This testing showed about 60% accuracy, meaning that the tree correctly predicted the label for 60% of the reviews on which we tested it. Although this is slightly better than prediction at random, it is less than we would like. We think the main reason that this tree was not very accurate was because our dataset was simply too small, so the tree did not have enough reviews to develop a well-trained model.

Next, we tried decision trees with the dataset of 3,000 professional descriptions of hotels. We utilized the presence of a family-friendly tag on the webpages for these hotels as our label. We also made a few enhancements to our approach. First, we used the random forest method (Flach), which builds multiple decision trees and combines the results when making predictions. We also set a maximum depth for our trees. Both of these enhancements are designed to build trees that avoid over-fitting. The problem of over-fitting occurs when a tree is well suited to the reviews on which it was trained but doesn't generalize well to other reviews. With this approach trained on the text from the 3,000 descriptions, we were able to achieve 74% accuracy, a significant improvement.

From this work with decision trees, we concluded that supervised learning, and particularly the decision tree model, is a promising approach for this problem. We think that our results would have continued to improve with even more data, as 3,000 data points is still rather small for a machine learning problem. Obtaining a larger set of labeled data proved difficult for a team of our size, so at this point we decided to focus our attention on unsupervised learning techniques. However, we think that Expedia would likely find success with this approach if they were to invest sufficient resources to collect more labeled data.

Unsupervised Learning

Unsupervised learning techniques input training data without labels and attempt to extract similarities from the data. For instance, given a set of hotel reviews, an unsupervised learning algorithm might group these reviews into a number of clusters based on similarities between the reviews. Although the algorithm does not provide a reason for assigning the groups, one can subsequently examine each group and try to assign it a higher-level meaning or label.

There are two main benefits to unsupervised learning techniques. The first is that they don't require any labeled training data. This means that they can easily be run on a large text corpus, and we don't need to take the time to assign any labels to the text. The second is that they scale very well to many different categories. Instead of running a separate decision tree model for each category that interests us, such as "family-friendly," "culturally-rich," and "good wine," we can simply run an unsupervised learning algorithm once and expect that the resulting model includes clusters for each of these high-level attributes.

In researching various unsupervised machine-learning techniques, we learned that a common approach used in the context of textual data mining is Latent Dirichlet Allocation, or LDA (Blei). LDA is a generative model that attempts to discover the underlying topics that the author of a piece of text had in mind when writing the text. When run on a large corpus of text documents, the result of LDA is a list of topics that are relevant to the corpus. Each of these topics is defined by a list of words. Each word in each topic is assigned a score specifying how applicable that word is to that topic.

Additionally, each text document in the corpus is given a score for each topic, corresponding to how much that topic makes up the document.

We ran LDA on our two remaining datasets, the hotel reviews from Expedia and the descriptions of cities. We utilized a number of preprocessing techniques on the data to improve our results. One of these was stemming, which reduces a word to its root stem. For instance, both "family" and "families" are reduced to "famili," and so they are considered the same word. Other preprocessing steps included considering important bigrams, i.e. two consecutive words that appear together often throughout the text, and removing common stop words such as "the" and "at," which add little meaning to the documents. Because unsupervised techniques output groupings rather than the specific predictions provided by supervised models, it is more difficult to assess the accuracy of these models. However, we found that LDA produced useful and interesting topics, which we were able to process into very relevant results.

Results

The results we obtained from running LDA were lists of topics, each of which was composed of a list of words. We had two such sets of lists, one from the Expedia data and one from the dataset of city descriptions. The next step was to process these results, as our ultimate goal was to assign scores to each region or hotel for a certain set of high-level categories, including "family-friendly" and "beachy".

We developed two separate scoring techniques to process the LDA results into these associations. The first approach was to examine the list of topics, and see if we could assign qualitative category tags to any of them. We did this using the list of words of which each topic was comprised. For instance, if a given topic contained the words "child," "pool," and "fun," we might label that topic as family-friendly. Then, because the LDA output specifies how closely each topic is associated to each region or hotel, we could use these percentages to score the destinations based on our labeled topics. When scoring the Expedia data, this approach worked fairly well for four topics: "family-friendly", "beachy", "good food", and "good wine". However, there were a number of drawbacks to this approach. First, we can only assign category tags if they are seen clearly in the topics; this is why the Expedia results only yielded four categories. Additionally, this approach requires a significant amount of human work to examine each and every topic in the LDA output; thus, this approach is not easily scalable. For these reasons, we also experimented with another approach.

The second technique to score regions and hotels using the LDA output is to search each topic for a predefined list of keywords. Because each topic is composed of a list of words, if we are interested in a particular word, such as "beach," we can search

through all of the words for each topic and see where the word appears. Each word relates to a given topic with some percentage, and each topic relates to a given hotel or region with some percentage. Thus, we can multiply those percentages and sum over all occurrences of the given word to assign a region or hotel a score for that word. Similarly, if we define a category to be a list of keywords, we can search for each of the keywords in the topic lists and combine the results to give each destination a score for that category. For example, if we define a beach topic to be composed of the words "beach", "sand", "surf", and "coast", we can search for all of those words and use their percentages to assign a beach score to each hotel or region. We can also include weightings for each word in the list to achieve more precise results.

Testing of the second approach produced very promising results. This approach is much more scalable than the first, because it simply requires defining word lists for each category of interest; the remainder of the process is automated. Thus, using the keyword search approach, we were able to extract scores for each region and hotel for a number of qualitative attributes.

Website

Although the main purpose of this project was to report on the benefits and drawbacks of the machine learning techniques we implemented, we also wanted to develop a tool to visualize the results we obtained. Because this project is geographical in nature, we endeavored to display our results on a map. We decided to build a website for this purpose. The website is built using HTML, JavaScript[5] and CSS[6]. All of our data, including information about the regions and hotels in California and scores and topic information for these destinations is stored in JSON[7] files that can be read into JavaScript.

The website serves two main functions. The first is to illustrate how our results could be incorporated into the Expedia website; in this sense it could be seen as a product prototype. The second function is to provide an easy way for Expedia engineers to examine our results, since they will likely be building off of our work in the future. We created two web pages on the site, each to focus on one of these goals.

The user page of the site contains two main features: a single-word keyword search and a ranking system for a number of higher-level categories. A user can specify a single search term of interest. The user can also specify a ranking of "Not Important," "Somewhat Important," and "Very Important" for a number of predefined categories. After choosing preferences, a user can hit a button to display the results of the search on a map.

To map the user's results, we must filter from the entire lists of regions and hotels to find the ones most relevant. To implement the keyword search, we utilize the method described in the results section: we look through the word lists for each topic and see

when the keyword appears. We use the word and topic percentages to assign each region and topic a score for the search term. Next, we filter based on the rankings for each category by looking at the score that each hotel and region has for each category. Using these scores and the ranking, combined with the result from the keyword search, we devise an overall score that specifies how closely a region or hotel matches the input data from the user. Finally, we map the top results, and the user can specify exactly how many regions and hotels are to be shown.

For the analysis page, which is targeted towards Expedia developers, we use a very similar process. However, this page contains some extra features to allow for more precise and in-depth analysis of the data. First, the user can input more precise rankings for each category. Second, the user can choose to filter regions based on their area, as well as what type of region they are (such as city or neighborhood). Finally, instead of a simple single-word search, users can specify a list of terms that interest them, along with a weight for each word. To implement this feature, we search the topic lists for each of the words, and use the weights specified for each word to combine the results.

We believe that this website serves as a useful tool to better analyze our data. The user page illustrates how our results could be incorporated into the Expedia website, while the analysis page offers a number of options to filter our data and examine the results we obtained.

Conclusion

The main purpose of this project was to generate a method to associate hotels and regions around the world with the higher-level attributes that describe them. To develop this process, our team worked on a number of different tasks. First, we collected and examined possible data sources, and ultimately gathered four useful datasets. Next, we researched machine learning as a tool for data mining, and implemented a number of machine learning algorithms.

The most important part of our project was an analysis of the benefits and drawbacks of each approach we tried. Ultimately, we came to the conclusion that supervised learning, in particular the decision tree model, is a promising approach that could return great results if used on enough labeled data. We also found that LDA, a type of unsupervised learning, returned topic lists that we were successfully able to process into qualitative metadata for geographical regions. Finally, the website we developed serves as a useful tool to analyze the associations we extracted from these results.

Ultimately, our results outline a number of techniques that can be used to associate geographical travel data with higher-level, qualitative attributes. This work is an important contribution that advances Expedia's goal of expanding into unconstrained search.

Acknowledgements

I would like to thank Brooke Cowan, Jean-Cedric Desrochers, Yanick Duchesne, Hervé Hacot, Ondrej Linda, and Michael Mai, our liaisons at Expedia, for presenting us with this project and for offering immensely helpful knowledge and suggestions. I would also like to acknowledge Professor Robert Keller, our advisor, for his unending support and guidance. I want to thank Professor Winston Ou for advising me on this report. Finally, I would like to acknowledge my teammates Chris Brown, Ben Leader, and Nabil Zaman.

Glossary

1. **PostgreSQL**: an open source, object-relational database management system.
2. **HTML** (HyperText Markup Language): the standard markup language used to create web pages.
3. **Python**: a widely used, high-level programming language that emphasizes code readability.
4. **BeautifulSoup**: a Python package used to parse HTML and XML documents.
5. **JavaScript**: a programming language used in web browsers.
6. **CSS** (Cascading Style Sheets): a language used to provide specifications for the look and format of web pages.
7. **JSON** (JavaScript Object Notation): a format that transmits data as human-readable text in attribute-value pairs.

References

Blei, David M., Andrew Y. Ng, and Michael I. Jordan. 2003. “Latent Dirichlet Allocation.” *The Journal of Machine Learning Research*. 3:993-1022.

Flach, Peter. 2012. *Machine Learning: The Art and Science of Algorithms that Make Sense of Data*. Cambridge University Press, Cambridge.