Tim Stewart (student ID number: 001476583)

C964 Capstone, Task 2

Tim’s Book Recommender: Assigning Similarity Between Books Using Links on Their Wikipedia Pages

Contents

Table of Figures

Table of Tables

A. Overview

Letter of Transmittal

Jeff Bezos, CEO, Amazon
410 Terry Avenue N
Seattle, WA 98109

July 29, 2022

RE: proposal to offer machine-learning algorithm to recommend books to customers at checkout

Dear Mr. Bezos,

As the top online retailer in the United States,[1] you already know that winning a new customer is up to 25 times as expensive as holding on to a customer you already have.[2] Amazon is the acknowledged master of the upsell. Features on your website such as "Customers who viewed this item also viewed…" and "Frequently bought together…" and "Recommended for you…" all tap into the vast business intelligence that Amazon has compiled by analyzing customers' past purchases.

The question for you today, Mr. Bezos, is how much longer your existing upsell methods will be good enough, and although they may be *good enough* for a while longer, could they be even better?

Based on my university-level research into deep-learning methods, I have developed an innovative way of making book recommendations that draws upon the "big data" available in tens of thousands of articles about books on the popular crowdsourced encyclopedia Wikipedia.[3] My neural-network model recommends similar books based on the quantity and type of hyperlinks in the Wikipedia articles about books. In my opinion, Wikipedia is an overlooked data source in the ecommerce industry. By using my machine-learning model trained on links found in Wikipedia articles, Amazon will make much more accurate book recommendations to its customers, resulting in increased sales and happier customers.

To showcase the power and capabilities of my book recommender system, I have created a prototype application that allows the user to choose a book and see a list of books that are similar to it. I urge you to carefully review the app prototype and the project proposal in the following pages and consider using my one-of-a-kind model to improve the accuracy of Amazon.com's book recommendations to its customers. For the low cost of $50,000 to cover costs of implementing my machine-learning model into your checkout pipeline, Amazon can secure its top spot in the online book market for years to come.

Sincerely,

Tim Stewart

[1] Amazon has a 41% share of the United States online retail market, as of October 2021 (Statista, 2021).
[2] "Depending on which study you believe, and what industry you're in, acquiring a new customer is anywhere from five to 25 times more expensive than retaining an existing one. It makes sense: you don't have to spend time and resources going out and finding a new client—you just have to keep the one you have happy" (Gallo, 2014).
[3] https://en.wikipedia.org/wiki/Main_Page

Problem Summary

The advent of the “big data” era has created immense opportunities and challenges for book shoppers and book retailers. On the one hand, the vast amount of information available about customers’ purchasing histories presents a source of intelligence unrivaled in past decades. On the other hand, the enormous quantity of information makes it nearly intractable using traditional techniques such as spreadsheets and database queries. This is where machine learning, a specialty that has been around only since the 1990s, comes to the fore. Machine learning crunches the gigabytes of data and reveals trends and similarities that are impossible to discern by any other method.

Machine learning can also address the problems customers have with finding new books to read, and these same machine-learning solutions enable retailers to offer better recommendations and make more sales.

My neural-network model was trained using 40,000 Wikipedia articles[4] about books and over a million individual hyperlinks, and the book similarities that the model reveals have likely never been identified before. If Amazon doesn’t take advantage of the extremely accurate book recommendations that my deep-learning model can produce, then I’m afraid Walmart or Target[5] will have to be the beneficiaries of this unique source of business intelligence.

The Data Product’s Benefits to Customers

The neural-network-based model, currently named Tim’s Book Recommender, is only 50 megabytes in size, but it represents the end result of fine-tuning more than 4 million trainable neurons using a state-of-the-art machine-learning-accelerating GPU and 12-core CPU. What this model can offer is an extremely accurate prediction of the similarity between the 40,000 books in its repertoire. The benefits that this highly complex model will bring to Amazon’s customers are easy to enumerate.

First, customers always want more and better recommendations, and Tim’s Book Recommender offers new and exciting recommendations based on a different set of data from what customers are used to seeing. Basing recommendations off what other customers bought or what items have the highest star ratings is so “yesterday.” My book recommender draws upon the combined knowledge of thousands of motivated people who collaborated to write these 40,000 Wikipedia articles about their beloved books. The hyperlinks in these articles are a unique data set that describes the general content of each book and relates the book to its place in the wider fabric of society and culture. With my book recommender, Amazon’s customers will get exciting new recommendations they haven’t seen before.

Second, in this day and age when customers have many options to choose from when shopping for books, from online retailers to brick-and-mortar bookstores, it is all too easy to lose a customer’s business by making inaccurate or irrelevant recommendations. Basing recommendations on what other people have bought fails to take into account the actual content of the books themselves. Just because someone else bought some of the same books that I did, that doesn’t mean that I am going to like all the other books this other person also likes. My book recommender system goes deeper than just comparing two people’s shopping carts or purchase histories: the neural-network model actually

[4] The exact number of Wikipedia articles that were used to train the neural-network model is 37,211.

[5] Walmart and Target are ranked #2 and #6, respectively, in market share of U.S. online retail as of October 2021 (Statista, 2021), and both sell books. Both are Amazon’s direct competitors in the book market.

compares the similarity of the books themselves, based on the hyperlinks in their Wikipedia articles, resulting in extremely accurate and relevant recommendations that will make Amazon's customers sit up and say "Wow, what a great recommendation!" If providing your customers with highly accurate book recommendations that entice them to add more books to their cart is one of Amazon's goals, then you need my book recommendation system.

Third, we live in an interconnected world, and online retailers should be looking for ways to draw together different sources of information from even unlikely places. Amazon might be used to filling up the columns of spreadsheets and tallying purchase histories in databases, but isn't it time to tap into information of a more artistic and creative style? The articles on Wikipedia are written in human language, not bits and bytes. The articles describe in detail not only the content of each book but how these books relate to other fields of human knowledge. By way of example, the Wikipedia page for *The Godfather*[6] by Mario Puzo not only has hyperlinks to obvious topics like the book's genre (crime novel), the author's name (Mario Puzo) and the publisher (G. P. Putnam's Sons), but also to such far-ranging topics as New York City and Long Island, World War II, the *New York Times* newspaper, former American president John F. Kennedy, the French writer Honoré de Balzac, and Frank Sinatra. All of these cultural elements are essential to understanding the place of this book in its bigger social context, and it is only with the help of a powerful machine-learning model like mine that the combined total of millions of details in all these Wikipedia articles about books can be synthesized and turned into a recommendation engine that tells customers what books are *really* similar to each other.

In short, when customers are buying books, they want to base their decision-making process on high-quality recommendations. Companies who are able to consistently provide reliable and interesting recommendations will not only win these customer's business today, but these customers will continue to come back time and again for fresh recommendations whenever they finish their current stack of bedside reading. If Amazon wants to be the store where these customers come for more great recommendations, then Amazon needs to integrate Tim's Book Recommender into its checkout process.

Data Product Overview

What makes my machine-learning model so special comes down to the data that was used to train it. The data set that was used to create this neural-network model can be described in two aspects. First, there are nearly 40,000 individual books in the data set. Second, there are 1,261,981 hyperlinks that are embedded in the 40,000 Wikipedia articles for these books.[7] This massive number of data points of books and links is exactly the right type of data set suitable for training a deep-learning model.

All these data points have been thoroughly processed by a deep-learning model to produce a highly accurate 50-dimensional vector embedding that can compute the degree of similarity between any two books among the 40,000 books in the model's repertoire. The hard work of training the neural-network model using multiple deep layers and dedicated hardware has been accomplished, and the finished

[6] https://en.wikipedia.org/wiki/The_Godfather_(novel)

[7] For example, links to "science fiction" appear in 5,468 different articles. Links to "New York City" appear in 958 different articles. Some links are not as popular: for example, only five book articles link to the topic "machine learning." Of the 1.2 million hyperlinks, there are only 314,381 distinct topics. Of the 314,381 distinct topics, 57% of them (=179,346) appear only one time anywhere in the articles. (That's a very long tail indeed!)

model can take the title of a book as input and return a list of up to 20 books whose similarity is based on the relative correspondence between the hyperlinks on these books' Wikipedia article pages.

Description of the Data Sources

The principal data source for Tim's Book Recommender are approximately 40,000 Wikipedia articles about books. These articles contain hyperlinks to various concepts, ideas, places, events, and people, and every book article has a unique combination of hyperlinks reflecting the various themes in the book and in the web of societal interconnections with the book. For example, the Wikipedia page for *Harry Potter and the Philosopher's Stone* by J. K. Rowling contains hyperlinks to Wikipedia articles about "London," "Boarding School," "Crumpet," "BBC," "C. S. Lewis," "Single Parent," "Accountability," "Spotify," and "Game Boy Advance." These hyperlinks contain far more detail and nuance about the book than, by comparison, what you might find in a library card catalog about a book with its stock of mainly bibliographical details.

The links from these 40,000 Wikipedia articles were extracted into a JSON file to facilitate preprocessing them and loading them into a machine-learning model. The uncompressed size of all the Wikipedia articles combined is 87 GB. The size of only the book-related Wikipedia pages is 2.2 GB. And the size of just the lists of hyperlinks that were used for the training set is 43 MB. The smaller the file, the faster it is to read it from disk and load it into a neural network. By progressively transforming the data from raw Wikipedia dump, to articles about books, to lists of hyperlinks, we increased the density and richness of the information to ensure that the deep-learning model had a clean and focused data source.

In the past, similarity between books has been estimated using shallow categories such as bibliographical details and subjective ratings from readers who "liked" or "didn't like" certain books. The use of the links from Wikipedia from this way is innovative, and it will produce refreshing book recommendations that are likely to surprise and delight Amazon's customers and keep them coming back and buying more books.

Objective and Hypotheses

The objective of Tim's Book Recommender is to provide Amazon's customers with compelling, interesting book recommendations that lead to customers buying the books that are being recommended. This objective dually benefits both Amazon, with regard to increased sales, and Amazon's customers, in terms of them having more enjoyment in finding good new books to read.

My hypothesis is that if we provide customers with book recommendations that are based on an unusual and rich data source, namely similar hyperlinks in the books' corresponding Wikipedia articles, then these customers will be pleased by the unexpected and tantalizing book recommendations being made to them and will feel compelled to buy those books.

Project Development Methodology

For past data-oriented projects I have used the SEMMA methodology[8] developed by SAS Institute. SEMMA has performed adequately for me before, but admittedly it is focused on the development of the data set and model. For Tim's Book Recommender I wanted to use a methodology that offered broader guidance to the project as a whole, not just the data portion. With this in mind, for my book-recommendation system I chose CRISP-DM[9] as the prevailing project methodology.

CRISP-DM contains guidelines for the selection, preparation, and modeling of a data set, but it goes a few steps further and relates these processes to a larger business strategy. The first step of CRISP-DM is "Business understanding." The idea here is that we are never working with data simply for its own sake. The handling of the data is always in service to a larger business objective.

I will have more to say about CRISP-DM and its application to the Tim's Book Recommender project in Section B of this document on page 21.

Funding Requirements

This project requires funding of $50,000 to hire three software contractors with deep expertise in app development to integrate my deep-learning model into Amazon's existing checkout pipeline. While it is certainly possible to hire experienced software developers for less money, it makes more sense for this project to pay top dollar, and I'd like to explain why.

First, we want senior developers who have experience writing software under a Non-Disclosure Agreement (NDA), since we don't want these developers sharing details of the machine-learning model with other ecommerce sites. Discreet developers tend to be more senior in experience and thus are more expensive.

Second, we want to decrease the remaining time-to-market (TTM)[10] as much as possible. In the field of machine learning, advances and breakthroughs are happening more rapidly than ever before. By paying more for software developers, we are more likely to be able to retain top talent who are accustomed to working quickly and efficiently. The sooner we make this book recommendation system available, the sooner we can begin upselling Amazon's customers and putting great new books into their hands.

Impact of the Product on Stakeholders

We cannot begin to predict the full impact this product may have without carefully considering who the stakeholders are. Certainly the Amazon corporation itself is a stakeholder, but the many people who own Amazon stock are also stakeholders. The shoppers who visit the Amazon website and who use the Amazon shopping app are also stakeholders who will be affected for better or for worse by the introduction of a new book recommendation feature.

[8] https://en.wikipedia.org/wiki/SEMMA

[9] https://en.wikipedia.org/wiki/Cross-industry_standard_process_for_data_mining

[10] "In commerce, time to market (TTM) is the length of time it takes from a product being conceived until its being available for sale. . . . A late product launch in any industry can negatively impact revenues—from reducing the window of opportunity to generate revenues to causing the product to become obsolete faster" (Wikipedia, 2022).

The initial impact of the recommender system will be felt first on the customers shopping for books. It may be a surprise for some of them to suddenly see book recommendations that are striking or unusual. However, this surprise at seeing unexpected book recommendations is one of the selling points of this data product. In years past, customers who bought, for example, a Harry Potter book might get recommendations for other Harry Potter books. How boring! But using this new Wikipedia-link-based recommendation system, someone who buys Harry Potter books might be recommended a tour guide of England, or an annotated history of broomsticks, or an autobiography from someone who graduated from boarding school. I predict that customers will be pleased by the interesting new recommendations, and as stakeholders in the outcome of this new data product, customer satisfaction should be one of our top priorities.

And then, quite naturally, we have our financial stakeholders, who are the Amazon corporation itself and its millions of stock shareholders. While it might contribute to an ideal world if Amazon's goals were simply to spread the joy of reading and help people find their next books, we have to accept the honest reality that in order for Amazon to continue to exist as a bookstore it has to sell books and make a profit doing so. And the book recommendation product is designed to do exactly that. By showing readers new kinds of book recommendations that make them want to click the "Buy" button immediately, we will be on track to see a measurable increase in Amazon's sales. Book readers win; Amazon wins; Amazon's shareholders win. It's a win-win-win situation!

Special Considerations on Sensitive Data

In many respects, the ecommerce industry has a much easier time when it comes to sensitive data. Hospitals and other healthcare providers are governed by the extremely strict Health Insurance Portability and Accountability Act of 1996 (HIPAA),[11] and all it takes is for a doctor or other medical professional to accidentally let slip someone's medical details to the wrong third party and they can be served with civil and criminal penalties. Amazon has it so easy compared to doctors and dentists!

The most obvious area that Amazon has to exercise caution and due diligence when it comes to sensitive data is regarding people's payment information. Amazon has many people's credit card and bank account numbers. This is remarkably convenient, but it means that a bad actor who can compromise Amazon's security systems has access to the financial credentials of millions of people. Fortunately, Amazon has already shown itself to exercise industry-standard security protocols such as HTTPS and SSL for connections to their website and using their app. The addition of a book recommendation feature to the smartphone app and website will be able to take immediate advantage of the pre-existing electronic security Amazon already has in place for its ecommerce operations.

But an area that is perhaps not so obvious yet is nevertheless an area of special sensitivity is people's choices in what they read. The books that people choose to read represent a wide variety of extremely personal and sensitive motivations. Someone might be shy about trying something new and they want to start by buying a book to learn more about it. They would be embarrassed if Amazon somehow let is slip that they were buying books on a new topic. Or consider a couple who might want to have a baby so they buy some books about the subject. Here again is a private family matter, and Amazon should not maintain records loosely in a way that this information could be leaked. We can even imagine that there are parts of the world where censorship laws prohibit people from learning about or even buying books

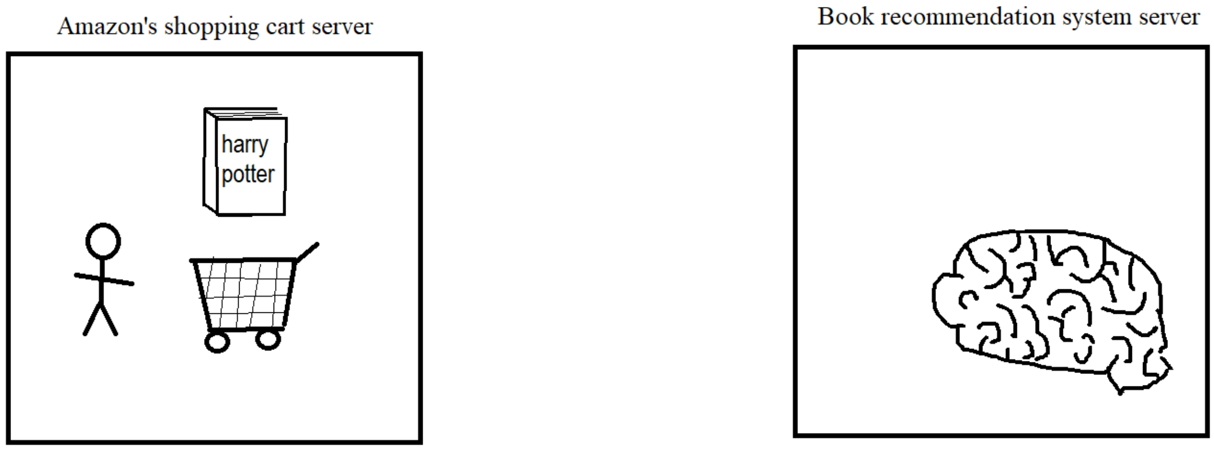
[11] https://www.hhs.gov/hipaa/index.html

about certain topics. If Amazon did not keep people's book purchases and recommendations private, then it would be a terrible outcome if a harsh government decided to press charges against a private citizen simply for buying a book on a censored topic.

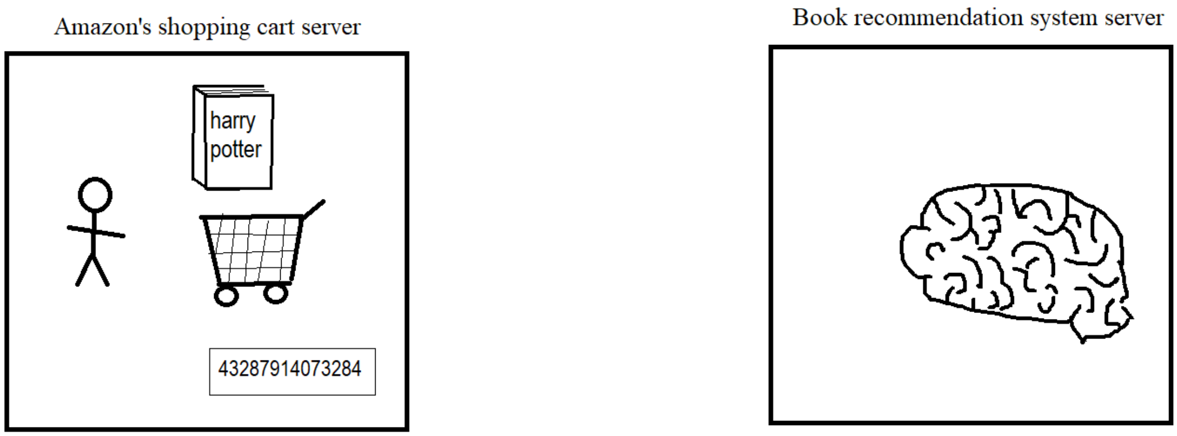
For this reason, the machine-learning model should be used in a way that does not keep long-term records about the books that are being fed into it or the book recommendations that result. Furthermore, it is not even necessary for the customer's name or account information to even be associated with the books that are being used as input to find similar books.

For example, if a customer named John Smith has recently bought the book *Harry Potter and the Philosopher's Stone*, how could we show Mr. Smith some similar books but without exposing his personal information to the machine-learning model? Here I describe one method we could use to accomplish this, with some basic illustrations to help show the separation that is preserved between the customer's personal details and their book choices and recommendations.

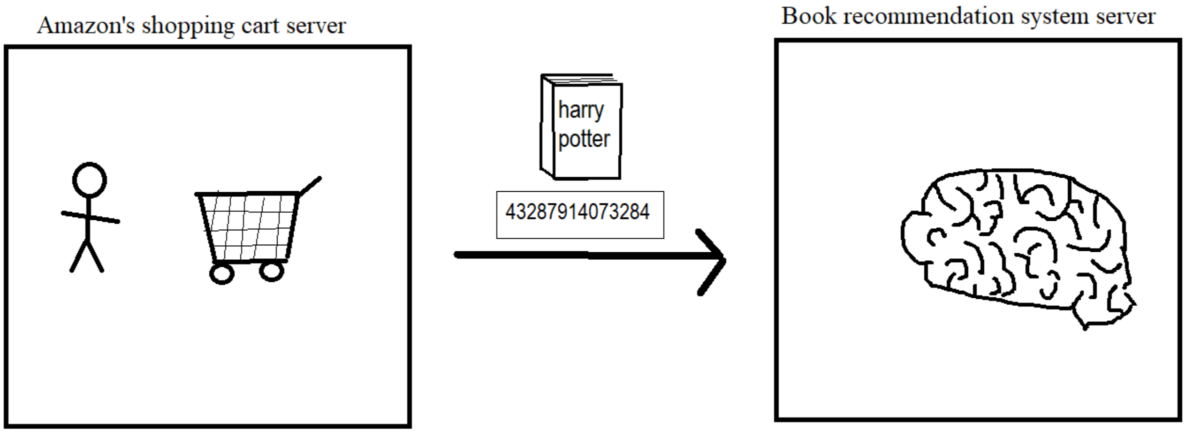
1. Let's start with a customer (the stick figure), and the book they put in their cart (*Harry Potter*). The book recommender is a brain since it's a neural network. We want to recommend more books to this customer without the book recommendation system having a record of who the person is.


Amazon's shopping cart server
harry potter
Book recommendation system server

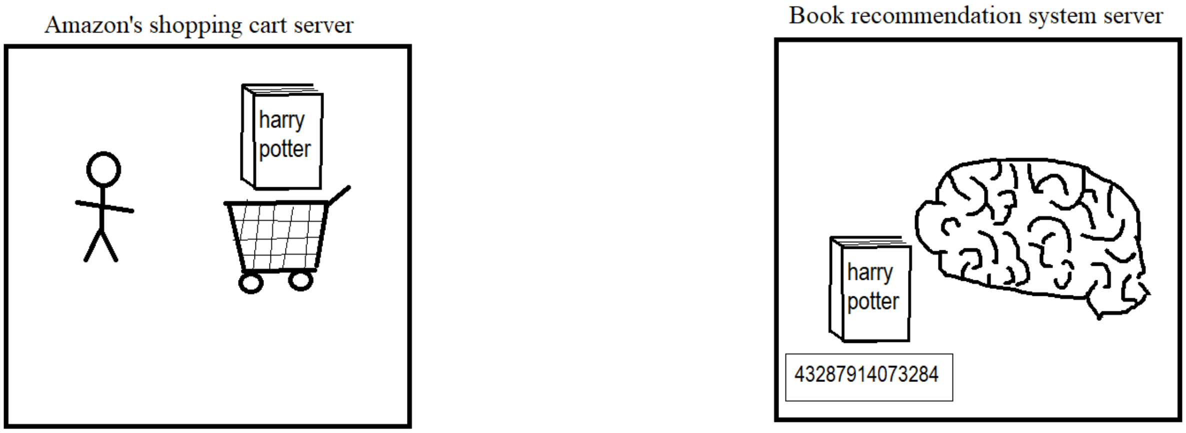
2. We protect the customer's privacy by creating a random number for them. This will be the customer's secret identity when we interact with the machine-learning model. (Here, the number starts 432879….)


Amazon's shopping cart server
harry potter
43287914073284
Book recommendation system server

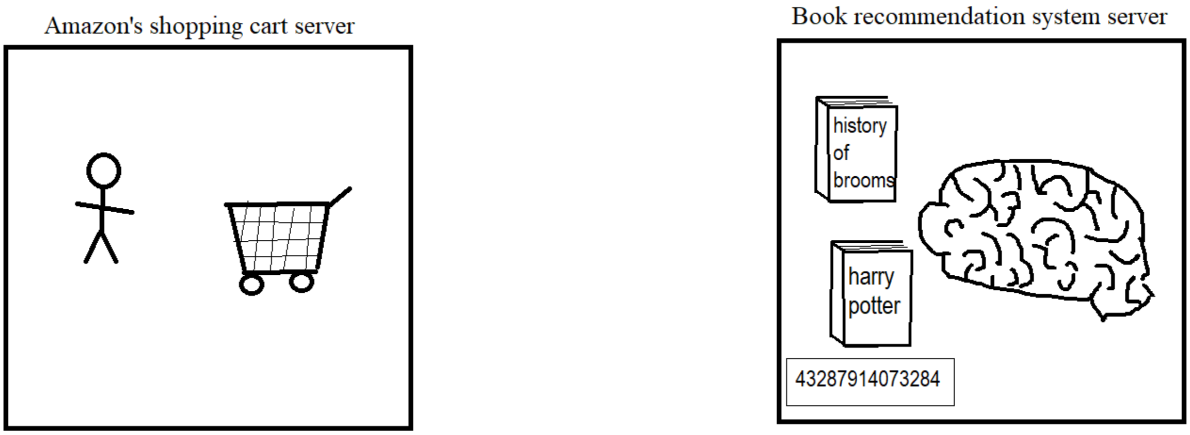
3. Now we send the input book (*Harry Potter*) and the secret identity number to the book recommendation server.


Amazon's shopping cart server
Book recommendation system server
harry
potter
43287914073284

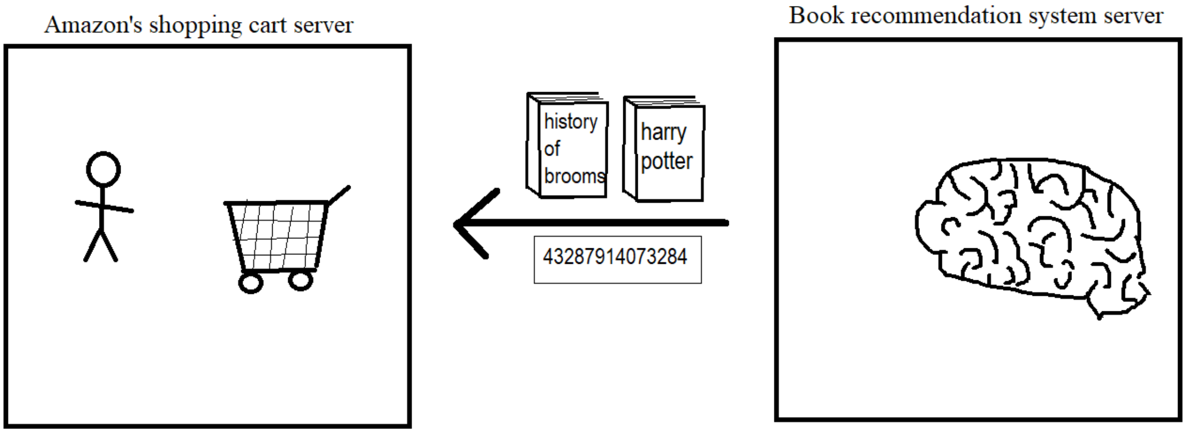
4. The book recommender thinks about a book similar to *Harry Potter* to recommend. (Note that the book recommender server does not know the customer's name or other details.)


Amazon's shopping cart server
harry
potter
Book recommendation system server
harry
potter
43287914073284

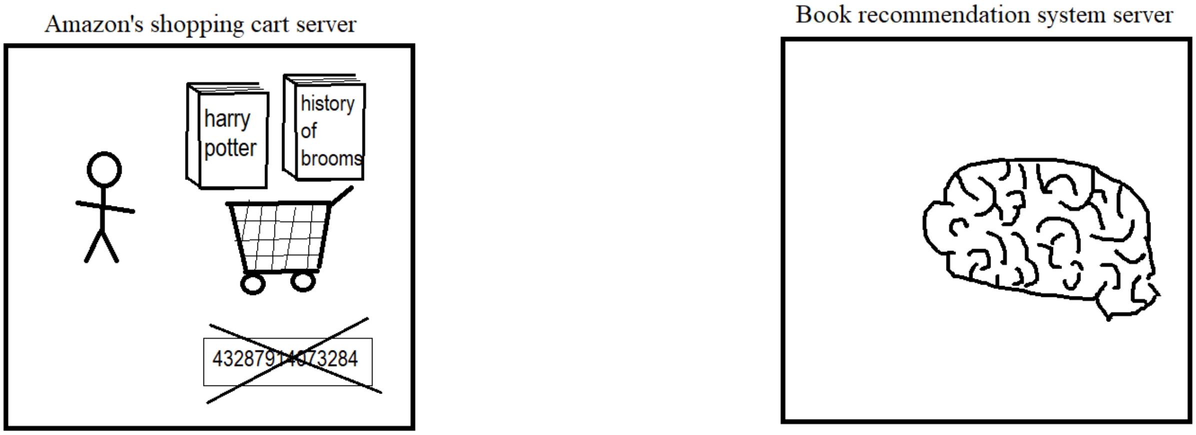
5. The book recommender picks *History of Brooms* as a similar book. Cool recommendation!


Amazon's shopping cart server
Book recommendation system server
history
of
brooms
harry
potter
43287914073284

6. The book recommender sends back the original *Harry Potter* book, plus the recommended book *History of Brooms*, plus the secret identity number. The book recommender never found out the customer's name or any of their personal details.


Amazon's shopping cart server
Book recommendation system server
history of brooms
harry potter
43287914073284

7. Now the customer sees the recommended book *History of Brooms*. They can decide to add this new book to their shopping cart or not. Meanwhile, the secret number is immediately deleted so there is no record of which customer was associated with it last. The customer's identify has been kept safe and secure from being copied to the book recommendation system.


Amazon's shopping cart server
Book recommendation system server
harry potter
history of brooms
43287914073284

These simple cartoons are in no way intended to trivialize how important it is to safeguard people's shopping choices, particularly when people are shopping for books, which tend to be personally meaningful and might even represent people's biggest dreams and ambitions. A person's inner life should always be treated with respect and not carelessly duplicated over and over again, making it increasingly likely that a data breach will spill those details to the public.

My Expertise

It’s only natural that Amazon would want to learn more about me and on what basis I am offering this data product and presenting it as a profitable financial investment. I am soon to be a graduate of Western Governors University,[12] a regionally accredited institution founded in 1997 to meet the needs of a variety of students, including those who want to earn a degree on a flexible schedule while continuing to hold down a full-time job. I am on track to earn a Bachelor of Science in Computer Science, having studied a plethora of subjects ranging from calculus and statistics, to ethics, to algorithms, to machine learning. It was this final subject, machine learning, that led me to apply the state-of-the-art skills I learned in my courses to the creation of a book recommendation system that tapped into a novel new source of data in order to provide readers with a wealth of interesting and new recommendations to fulfill their reading ambitions.

One of my strengths as an entrepreneur is coming up with big ideas, and most recently that idea was “how can we make better book recommendations to people?” While I could just set up a website and make this machine-learning algorithm available to some people, I want this data product to reach as many people as possible. This is a pretty big dream! Where could I find a huge number of readers who want help looking for their next book? The answer is Amazon.com! Using my machine-learning model to help Amazon customers buy additional books is a win-win-win in my book. The first win is for the readers who are going to receive exciting new recommendations that they’ve never heard of before. The second win is for Amazon, who is going to be selling lots more books. And the third win is for me, because it makes me happy to know that in a small way I am able to get more people reading more books that they will come to love.

[12] https://www.wgu.edu/

B. Technical Summary

Overview

This section deals with Tim's Book Recommender in technical detail. The book recommendation system is built on the foundation of a robust data set that was transformed using industry-standard methods and then used to train a well-architected neural network, which is now capable of presenting book recommendations in a customer-friendly format. See below for specific details about every part of this machine-learning model's development.

Problem Statement

Amazon's problem is that customers have become bored of the same old tired book recommendations based on shallow relationships between books. Customers are desperate for interesting book recommendations that break with the old ways of identifying similar books. Customer demand a more nuanced approach for recommending new books that are similar to books they've enjoyed in the past.

To put it bluntly, Amazon needs to tap into a new way of recommending books in order to hold on to their strategic market position. Whichever company is able to satisfy readers' curiosity and desire for interesting new books is going to have an enormous economic advantage. If Amazon doesn't make a switch to a more complex data model that is based on a brand new source of book details and relationships, their sales will stagnate and fall off and another company that is willing to take advantage of machine learning will be propelled into first place.

Customer Description

Amazon's customers have the financial means to buy books they are interested in reading, but they are also have refinement of taste, meaning that if they don't know what to buy or if they are given boring book recommendations, then they will patiently hold on to their dollars and put off purchases until a later time. In order to upsell Amazon's customers we have to actually offer the customers something that they value and that will compel them to make a purchasing decision. For readers of books, the value proposition is an endless supply of amazing book recommendations that aren't available anywhere else.

In addition to being able to give customers innovative book recommendations, the machine-learning model that I have developed will be able to definitively prove to customers that they are getting the very best recommendations because the model can show easy-to-understand data visualizations that will amaze the customers and convince them that they are truly benefiting from artificial intelligence at its most cutting edge. While some customers will be convinced of Amazon's superior book recommendation system simply by looking at the list of recommended books, other customers will want some proof that there is some amazing technology being used, and the data visualizations will be just the ticket to win these skeptical customers over. Soon, all of Amazon's customers will be true believers that Tim's Book Recommender is truly the best possible way to obtain high-quality recommendations about what books to buy next.

Gap Analysis of Amazon’s Existing Book Recommendation System

Amazon currently employs a variety of methods of recommending new products, including books, to customers. Their most well-known feature uses the conventional “People who bought X also bought Y” algorithm. For example, if a lot of other people who bought the *Harry Potter* book also bought a wizard costume, then anytime a person buys the *Harry Potter* book the algorithm would also recommend buying a wizard costume. One advantage of this system is that it is very easy to implement. A drawback of this system is that it is based on what other customers have bought, and since everybody is different, what was a good combination of shopping items for other people might not necessarily be a desirable combination of items for the next person in line. If I like reading stories about Harry Potter but I hate dressing up in costumes, then even though a lot of other people bought *Harry Potter* and a costume, I’m going to be annoyed if a costume is recommended to me, because that’s not what I’m interested in.

This is an area where my machine-learning model will have an advantage over Amazon’s existing solution. Because a machine-learning model based on Wikipedia links bases its recommendations on the books themselves and what ideas and concepts they are related to, the recommendations are based on a more objective foundation. No matter how many other people like *Harry Potter* or who like wizard costumes, the fact remains that we know *Harry Potter* is connected to London, brooms, and boarding schools because those are the links on its Wikipedia page, so those are very likely to be a reasonable basis for making book recommendations.

Description of Data

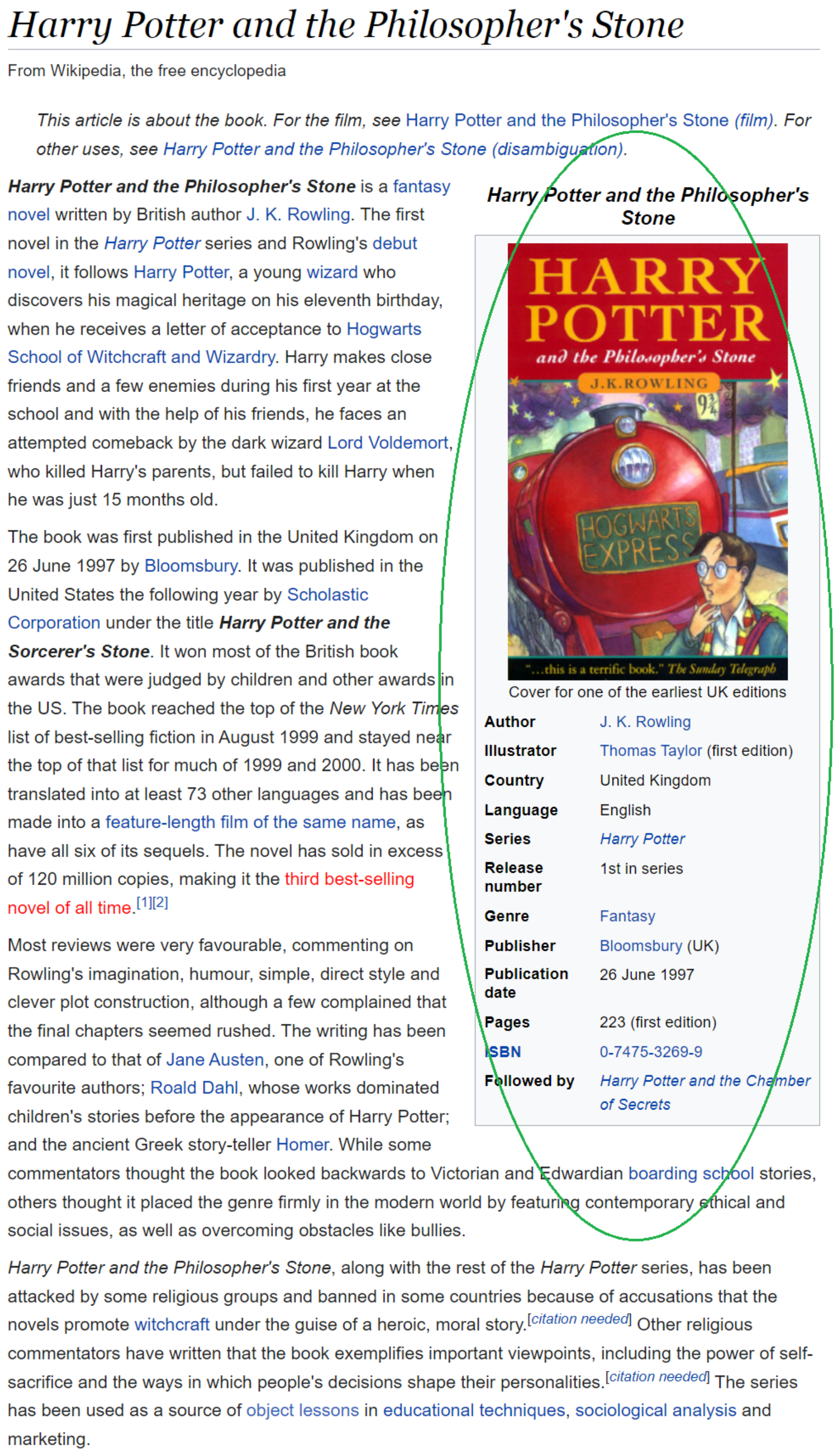
The contents of actual Wikipedia articles was used to formulate a data set suitable for training and validating a deep-learning model. In an unambiguously generous gesture, the Wikimedia Foundation that oversees Wikipedia regularly makes the entire contents of Wikipedia available for download in what they call a “dump,” as in a “dump of information.” The dump is a 20-gigabyte compressed zip file that expands to 87 GB when it is uncompressed. This archive contains the full text, including hyperlinks, of every article on Wikipedia. It’s an enormous trove of human knowledge, and the fact that you can download it all and store it on your computer is frankly astonishing.

I wrote a Python program to open one of these Wikipedia dump archives and check every article one at a time to see if it was about a book. This was a relatively straightforward process since most Wikipedia pages about books have a telltale “information box” that contains basic bibliographical information, and the presence of this “infobox” can be detected programmatically. Figure 1 shows a screenshot of the Wikipedia page for the first *Harry Potter* book, and I have annotated the image with the telltale “infobox” circled in green.

My analysis of the archive identified 40,000 Wikipedia articles about books, and from each article I extracted the book’s title and author and other bibliographical details as well as a list of the hyperlinks in the article that led to other articles on Wikipedia. For example, in Figure 1 the hyperlinks are in blue, and we see links to “fantasy novel,” “J. K. Rowling,” “debut novel,” “wizard,” “Hogwarts,” “Bloomsbury,” “Scholastic Corporation,” and others. These links function as a kind of DNA of the article, telling what important concepts or ideas are connected to this particular book.

Figure 1

A screenshot of the Wikipedia article for the first Harry Potter *book, with green oval added*



HARRY POTTER
and the Philosopher's Stone
J.K.ROWLING
9¾
HOGWARTS EXPRESS
"...this is a terrific book." The Sunday Telegraph

Harry Potter and the Philosopher's Stone

From Wikipedia, the free encyclopedia

This article is about the book. For the film, see Harry Potter and the Philosopher's Stone (film). For other uses, see Harry Potter and the Philosopher's Stone (disambiguation).

Harry Potter and the Philosopher's Stone is a fantasy novel written by British author J. K. Rowling. The first novel in the *Harry Potter* series and Rowling's debut novel, it follows Harry Potter, a young wizard who discovers his magical heritage on his eleventh birthday, when he receives a letter of acceptance to Hogwarts School of Witchcraft and Wizardry. Harry makes close friends and a few enemies during his first year at the school and with the help of his friends, he faces an attempted comeback by the dark wizard Lord Voldemort, who killed Harry's parents, but failed to kill Harry when he was just 15 months old.

The book was first published in the United Kingdom on 26 June 1997 by Bloomsbury. It was published in the United States the following year by Scholastic Corporation under the title ***Harry Potter and the Sorcerer's Stone***. It won most of the British book awards that were judged by children and other awards in the US. The book reached the top of the *New York Times* list of best-selling fiction in August 1999 and stayed near the top of that list for much of 1999 and 2000. It has been translated into at least 73 other languages and has been made into a feature-length film of the same name, as have all six of its sequels. The novel has sold in excess of 120 million copies, making it the third best-selling novel of all time.[1][2]

Most reviews were very favourable, commenting on Rowling's imagination, humour, simple, direct style and clever plot construction, although a few complained that the final chapters seemed rushed. The writing has been compared to that of Jane Austen, one of Rowling's favourite authors; Roald Dahl, whose works dominated children's stories before the appearance of Harry Potter; and the ancient Greek story-teller Homer. While some commentators thought the book looked backwards to Victorian and Edwardian boarding school stories, others thought it placed the genre firmly in the modern world by featuring contemporary ethical and social issues, as well as overcoming obstacles like bullies.

Harry Potter and the Philosopher's Stone, along with the rest of the *Harry Potter* series, has been attacked by some religious groups and banned in some countries because of accusations that the novels promote witchcraft under the guise of a heroic, moral story.[citation needed] Other religious commentators have written that the book exemplifies important viewpoints, including the power of self-sacrifice and the ways in which people's decisions shape their personalities.[citation needed] The series has been used as a source of object lessons in educational techniques, sociological analysis and marketing.

Harry Potter and the Philosopher's Stone

Cover for one of the earliest UK editions

Author	J. K. Rowling
Illustrator	Thomas Taylor (first edition)
Country	United Kingdom
Language	English
Series	*Harry Potter*
Release number	1st in series
Genre	Fantasy
Publisher	Bloomsbury (UK)
Publication date	26 June 1997
Pages	223 (first edition)
ISBN	0-7475-3269-9
Followed by	*Harry Potter and the Chamber of Secrets*

Figure 2 shows some of the extracted data in JSON[13] format. Note that each book was assigned a `book_id` since in many cases it was convenient to have an integer number that could be used to refer to a book rather than a long title. The `book_id` was especially helpful since there were many books in Wikipedia articles that did not have an ISBN number or other modern numeric code because they were published a century ago or more. For example, the beloved book *Little House on the Prairie* by Laura Ingalls Wilder was published in 1932, several decades before the ISBN was invented in 1967.

From the perspective of machine learning, the most important part of the JSON data is the list of links after the tag "wikilinks." As mentioned before, these links are the DNA of each article, and it's how we can compare how similar or dissimilar two articles are from each other. The more their links have in common, the more similar they are.

Extracting this book information along with the list of links meant that we no longer had to deal with the enormous multi-gigabyte Wikipedia archive files, which require a lot of time and computing power to access and search through. The JSON file containing the book's bibliographical details and the lists of wikilinks was only 43 megabytes in size for all 40,000 books (about 1 kilobyte of data per book, on average).

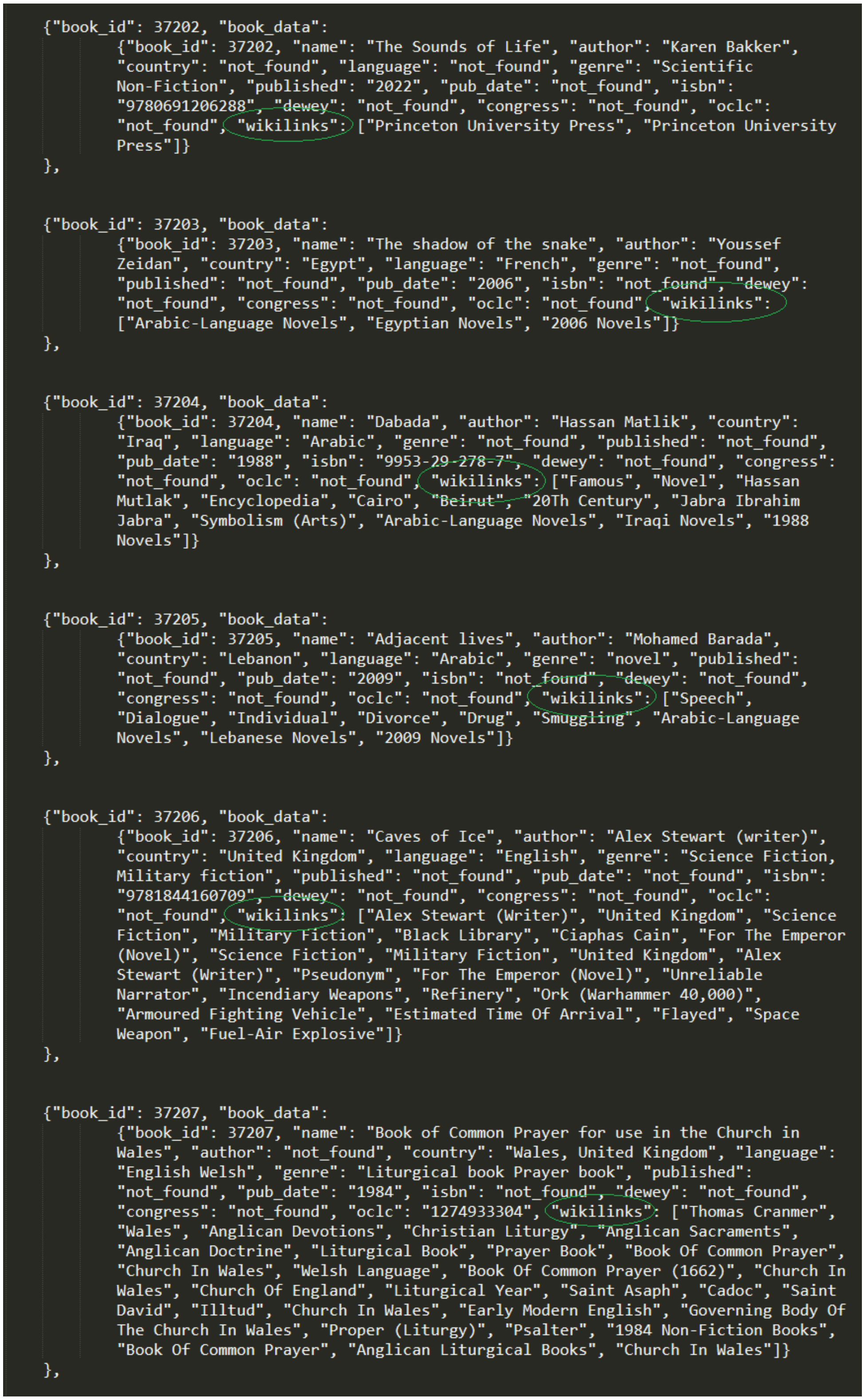
An additional advantage of the JSON format is that it is minimally "typed," meaning that almost all the data is considered a string. Another benefit of JSON is that it is well suited for lists of items, and the list of hyperlinks for each article is indeed in list format, as denoted by the square brackets that enclose the list of links immediately after the "wikilinks" tag in Figure 2.

We should also consider the possibility of regularly updating the machine-learning model, since new articles about books are added to Wikipedia every day. At the very least, from one year to the next we would expect to find new Wikipedia articles for the new books that have been published in the past 12 months. So it was highly strategic to find an efficient and reproducible way of extracting the book data from the Wikipedia dump, because 6 months or 12 months from now, I may be interested in re-running the data extraction procedures in order to train a fresh book recommendation model that includes information for the most recently published books.

[13] "JSON (JavaScript Object Notation) is an open standard file format and data interchange format that uses human-readable text to store and transmit data objects consisting of attribute–value pairs and arrays (or other serializable values). It is a common data format with diverse uses in electronic data interchange" (Wikipedia, 2022).

Figure 2

Excerpt of my JSON file showing bibliographical details and list of hyperlinks for several Wikipedia articles about books, with the "wikilinks" tag circled in green



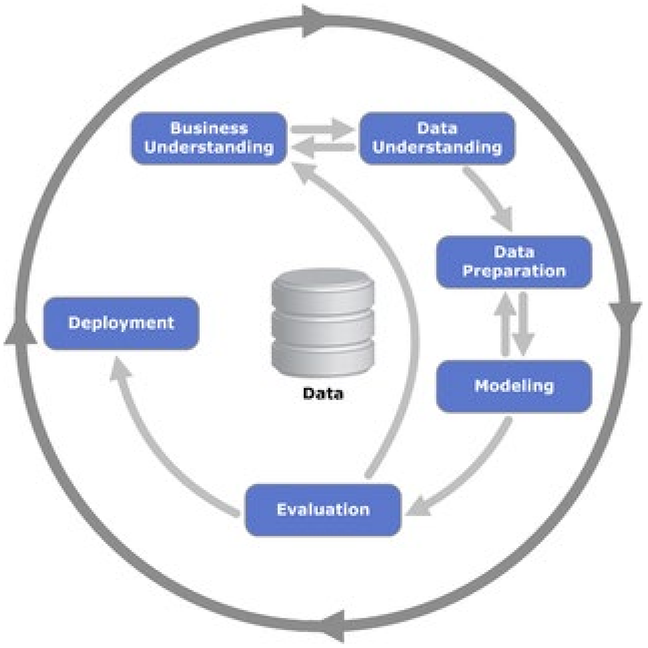
```
{"book_id": 37202, "book_data":
        {"book_id": 37202, "name": "The Sounds of Life", "author": "Karen Bakker",
        "country": "not_found", "language": "not_found", "genre": "Scientific
        Non-Fiction", "published": "2022", "pub_date": "not_found", "isbn":
        "9780691206288", "dewey": "not_found", "congress": "not_found", "oclc":
        "not_found", "wikilinks": ["Princeton University Press", "Princeton University
        Press"]}
},


{"book_id": 37203, "book_data":
        {"book_id": 37203, "name": "The shadow of the snake", "author": "Youssef
        Zeidan", "country": "Egypt", "language": "French", "genre": "not_found",
        "published": "not_found", "pub_date": "2006", "isbn": "not_found", "dewey":
        "not_found", "congress": "not_found", "oclc": "not_found", "wikilinks":
        ["Arabic-Language Novels", "Egyptian Novels", "2006 Novels"]}
},


{"book_id": 37204, "book_data":
        {"book_id": 37204, "name": "Dabada", "author": "Hassan Matlik", "country":
        "Iraq", "language": "Arabic", "genre": "not_found", "published": "not_found",
        "pub_date": "1988", "isbn": "9953-29-278-7", "dewey": "not_found", "congress":
        "not_found", "oclc": "not_found", "wikilinks": ["Famous", "Novel", "Hassan
        Mutlak", "Encyclopedia", "Cairo", "Beirut", "20Th Century", "Jabra Ibrahim
        Jabra", "Symbolism (Arts)", "Arabic-Language Novels", "Iraqi Novels", "1988
        Novels"]}
},


{"book_id": 37205, "book_data":
        {"book_id": 37205, "name": "Adjacent lives", "author": "Mohamed Barada",
        "country": "Lebanon", "language": "Arabic", "genre": "novel", "published":
        "not_found", "pub_date": "2009", "isbn": "not_found", "dewey": "not_found",
        "congress": "not_found", "oclc": "not_found", "wikilinks": ["Speech",
        "Dialogue", "Individual", "Divorce", "Drug", "Smuggling", "Arabic-Language
        Novels", "Lebanese Novels", "2009 Novels"]}
},


{"book_id": 37206, "book_data":
        {"book_id": 37206, "name": "Caves of Ice", "author": "Alex Stewart (writer)",
        "country": "United Kingdom", "language": "English", "genre": "Science Fiction,
        Military fiction", "published": "not_found", "pub_date": "not_found", "isbn":
        "9781844160709", "dewey": "not_found", "congress": "not_found", "oclc":
        "not_found", "wikilinks": ["Alex Stewart (Writer)", "United Kingdom", "Science
        Fiction", "Military Fiction", "Black Library", "Ciaphas Cain", "For The Emperor
        (Novel)", "Science Fiction", "Military Fiction", "United Kingdom", "Alex
        Stewart (Writer)", "Pseudonym", "For The Emperor (Novel)", "Unreliable
        Narrator", "Incendiary Weapons", "Refinery", "Ork (Warhammer 40,000)",
        "Armoured Fighting Vehicle", "Estimated Time Of Arrival", "Flayed", "Space
        Weapon", "Fuel-Air Explosive"]}
},


{"book_id": 37207, "book_data":
        {"book_id": 37207, "name": "Book of Common Prayer for use in the Church in
        Wales", "author": "not_found", "country": "Wales, United Kingdom", "language":
        "English Welsh", "genre": "Liturgical book Prayer book", "published":
        "not_found", "pub_date": "1984", "isbn": "not_found", "dewey": "not_found",
        "congress": "not_found", "oclc": "1274933304", "wikilinks": ["Thomas Cranmer",
        "Wales", "Anglican Devotions", "Christian Liturgy", "Anglican Sacraments",
        "Anglican Doctrine", "Liturgical Book", "Prayer Book", "Book Of Common Prayer",
        "Church In Wales", "Welsh Language", "Book Of Common Prayer (1662)", "Church In
        Wales", "Church Of England", "Liturgical Year", "Saint Asaph", "Cadoc", "Saint
        David", "Illtud", "Church In Wales", "Early Modern English", "Governing Body Of
        The Church In Wales", "Proper (Liturgy)", "Psalter", "1984 Non-Fiction Books",
        "Book Of Common Prayer", "Anglican Liturgical Books", "Church In Wales"]}
},
```

Methodology Used for Product Design and Development

When it comes to software development, there is no shortage of methodologies to choose from such as rapid application development (RAD),[14] the waterfall model,[15] and Agile.[16] All these models have their pros and cons, but for this project I decided to use the CRISP-DM[17] methodology to guide the development of this data product.

Figure 3

The relationships between the six phases of CRISP-DM are shown in this process diagram.



Note: Diagram from "Cross-industry standard process for data mining." In *Wikipedia*. Retrieved from https://en.wikipedia.org/wiki/Cross-industry_standard_process_for_data_mining. Diagram copyright 2012 by K. Jensen. Reprinted in accordance with CC BY-SA 3.0.

CRISP-DM is an acronym for "Cross-industry standard process for data mining." It is a process model based on an open standard that guides a team's efforts to develop, analyze, and deploy a data product.

There are six phases in the CRISP-DM methodology: business understanding, data understanding, data preparation, modeling, evaluation, and deployment. See Figure 3 for a visual representation of how these six phases relate to one another.

For a data-centered product such as my book recommender system, the phases of developing the data model largely parallel the phases of developing the product, so it is on that basis that I will describe the development of the data model and the product.

[14] "Rapid application development (RAD) . . . is both a general term for adaptive software development approaches, and the name for James Martin's method of rapid development. In general, RAD approaches to software development put less emphasis on planning and more emphasis on an adaptive process" (Wikipedia, 2021).

[15] "The waterfall model is a breakdown of project activities into linear sequential phases, where each phase depends on the deliverables of the previous one and corresponds to a specialization of tasks. The approach is typical for certain areas of engineering design. In software development, it tends to be among the less iterative and flexible approaches, as progress flows in largely one direction ('downwards' like a waterfall) through the phases of conception, initiation, analysis, design, construction, testing, deployment and maintenance" (Wikipedia, 2022).

[16] "In software development . . . Agile practices include requirements discovery and solutions improvement through the collaborative effort of self-organizing and cross-functional teams with their customers/end users, adaptive planning, evolutionary development, early delivery, continual improvement, and flexible responses to changes in requirements, capacity, and understanding of the problems to be solved" (Wikipedia, 2022).

[17] "Cross-industry standard process for data mining, known as CRISP-DM, is an open standard process model that describes common approaches used by data mining experts. It is the most widely-used analytics model" (Wikipedia, 2022).

Business understanding. The first phase of CRISP-DM, and indeed the first phase of any business project, is to have a business understanding of the project's objective. In the case of the book recommender system, the business understanding is that we want to sell additional books to customers by giving them a recommendation so enticing that they can't refuse it.

Data understanding. The next step is to have an understanding of the data. From the perspective of the data model, this means selecting a data set that is large enough to be useful but not so large that it is unwieldy to work with or overly noisy due to unmeaningful details. The 40,000 books and the corresponding hyperlinks on their Wikipedia article pages lands in the sweet spot of being of a manageable size yet yielding a high information density. From a business perspective, using links from an open-source online encyclopedia to create a book similarity model represents an unusual and distinctive data source that sets this machine-learning product apart from the competition.

Data preparation. Here is where we take steps to transform the data so we can make the best use of it. Preparation in terms of this project meant extracting the essential data from the enormous Wikipedia dump and storing it in a structured file format such as JSON so that we can load it into a machine-learning model in a later step. From the business side of things, data preparation also meant deciding what data details customers are going to want. For the book recommendation product, I decided that customers would not be interested in bibliographical details like page count and publisher but that they would be extremely interested in details about which books are similar to other ones. This meant prioritizing the storage of the list of wikilinks for each book in a way that privileges them as the centerpiece of the upcoming data model.

Modeling. This is when the machine-learning component of this project really starts to shine. The lists of hyperlinks for each of the 40,000 books were fed into a neural network with multiple deep layers of neurons. The network was trained repeatedly using the TensorFlow[18] and Keras[19] software packages, fine-tuning over 4 million trainable parameters over multiple iterations (called "epochs") in continuous process of minimizing a specified loss function.[20] The end result was a data model that "understands" how similar two books are and can rank all 40,000 books in terms of their similarity with respect to a particular selected book. When we consider the significance of this in business terms, we have what is practically a revolutionary new way of guiding readers into new and interesting books that build on what books these readers have already enjoyed. With compelling book recommendations like these, we would expect (again, from a business standpoint) to see a dramatic increase in additional book sales owing to our new recommendations to customers.

Evaluation. Although modeling feels like the exciting climax of the machine-learning process, the necessary follow-up step is to see if the model is really any good. Data scientists and machine-learning experts are now cowboys who only live for the exciting parts. What separates the mavericks from the professionals is that the professionals go back and assess and evaluate in what ways the model performs well and in what ways it is performing unexpectedly or not fully up to task. In IBM's explanation of the "Evaluation" step in their tutorial on the CRISP-DM model, they describe it this way: "At this point, you've completed most of your data mining project. You've also determined, in the Modeling phase, that the models built are technically correct and effective according to the data mining success criteria that you defined earlier. Before continuing, however, you should evaluate the results of your efforts

[18] https://www.tensorflow.org/

[19] https://keras.io/

[20] For this use case, specifically the "binary crossentropy" loss function was used.

using the business success criteria established at the beginning of the project. This is the key to ensuring that your organization can make use of the results you've obtained" (IBM, 2021).

For the purposes of my evaluation phase with the book recommender model, I repeatedly entered the titles of books that I have read and was well familiar with, and I was surprised and delighted by the lists of similar books that the model returned. I typed in *The Godfather* (1969) by Mario Puzo, and one of the results was a book I had never heard of before, *The Winter of Frankie Machine* by Don Winslow published in 2006 just a few years ago. I entered the name of one of my favorite collections of short stories, *Dancing After Hours* (1996) by Andre Dubus, and the top result was *Citizenville: How to Take the Town Square Digital and Reinvent Government* (2013) by current California governor Gavin Newsom. I can safely say that I never would've predicted that a book of short stories would have a strong similarity to a book about reforming government. I'm sorely temped to buy Newsom's book just to find out what makes his book so similar to my beloved anthology of short stories.

Deployment. The last phase of the CRISP-DM model is deployment. This is where the data model goes out into the world and we get to test it on real, live people and obtain real, live data from people using it. In the case of Tim's Book Recommender, the deployment phase means letting real-world customers access the book recommender so we can see if the book suggestions are as compelling as we believe that they will be. In data research as it is in business, deployment is the finish line that everyone is trying to reach. Some might say that a product idea is just a dream until you actually start shipping it to your customers in the deployment phase.

An important aspect of the CRISP-DM methodology, or any modern methodology really, is that if the results you're getting at a particular phase aren't turning out as well as you'd hoped, you can backtrack to an earlier phase, change up your methods or fine-tune your parameters, and restart from that point forward and hope to get improved results. As the process diagram in Figure 3 shows, there are not only arrows moving the process forward toward deployment but also arrows circling backward, sometimes even many steps backward. A willingness to do-over and try again is as necessary a habit in data science as it is in business.

Deliverables

This project is chiefly about creating a new data product suitable for use in Amazon's existing checkout process. To that end, there are project deliverables in the form of a user guide, a quick-start guide, and a comprehensive installation guide for use in exactly reproducing the steps to develop this data product.

There are also product deliverables associated with the Tim's Book Recommender data product. In the software category, there are two deliverables: the AI model, and the app prototype.

- The AI model is a 50-megabyte binary file containing a trained neural network capable of identifying the degree of similarity between any two out of 40,000 books. The model is also capable of ranking all 40,000 books in terms of their degree of similarity to one particular book.
- The app prototype is a small, functional GUI application written using the Python programming language that allows users to search the model for books and display a list of books that are similar to it. In addition, the app prototype can generate three different data visualizations that use 2-dimensional plots to represent the similarity between books in an intuitive way.

Implementation

The actual implementation of Tim's Book Recommender involves several discrete steps. I expect that any disruption to Amazon's regular business will be negligible because the book recommendation feature is something that is being added to the existing checkout workflow rather than replacing an existing component.

First, the trained machine-learning model will need to be stored on Amazon's content delivery network (CDN) for use by Amazon's ecommerce servers. Next, a new multistep function will need to be added to the checkout routine. Figure 4 represents where this new functionality will be implemented in Amazon's checkout process. Because the book recommendation feature is being implemented as a new step, the impact to existing systems should be minimal, though as acknowledged in the Outcomes section immediately below, it may take slightly longer for customers to checkout since they may have to respond to whether to add the additional recommended books to their cart.

As shown in the flowchart in Figure 4 (next page), the privacy of customers is ensured by the implementation of a "secret number" to conceal their identity from the book recommendation server. This step was also described in the section "Special Considerations on Sensitive Data." This component is intended to minimize the amount of customer data that is spread out across various Amazon computer servers. After all, it's easier to secure a small amount of private customer data on a small number of servers versus trying to secure a vast amount of customer information duplicated across many servers.

Connecting the new model and its interface software to Amazon's existing checkout application is the reason for hiring three experienced software developers to execute the implementation phase. Because Amazon's ecommerce business operates at an extremely high scale with the highest possible levels of web traffic, time and effort applied toward integrating the model carefully and precisely is warranted.

Outcomes

We can expect to see several likely outcomes, some positive and some negative, as a result of the development of this data product and its implementation in the Amazon checkout process.

Probable positive outcomes:

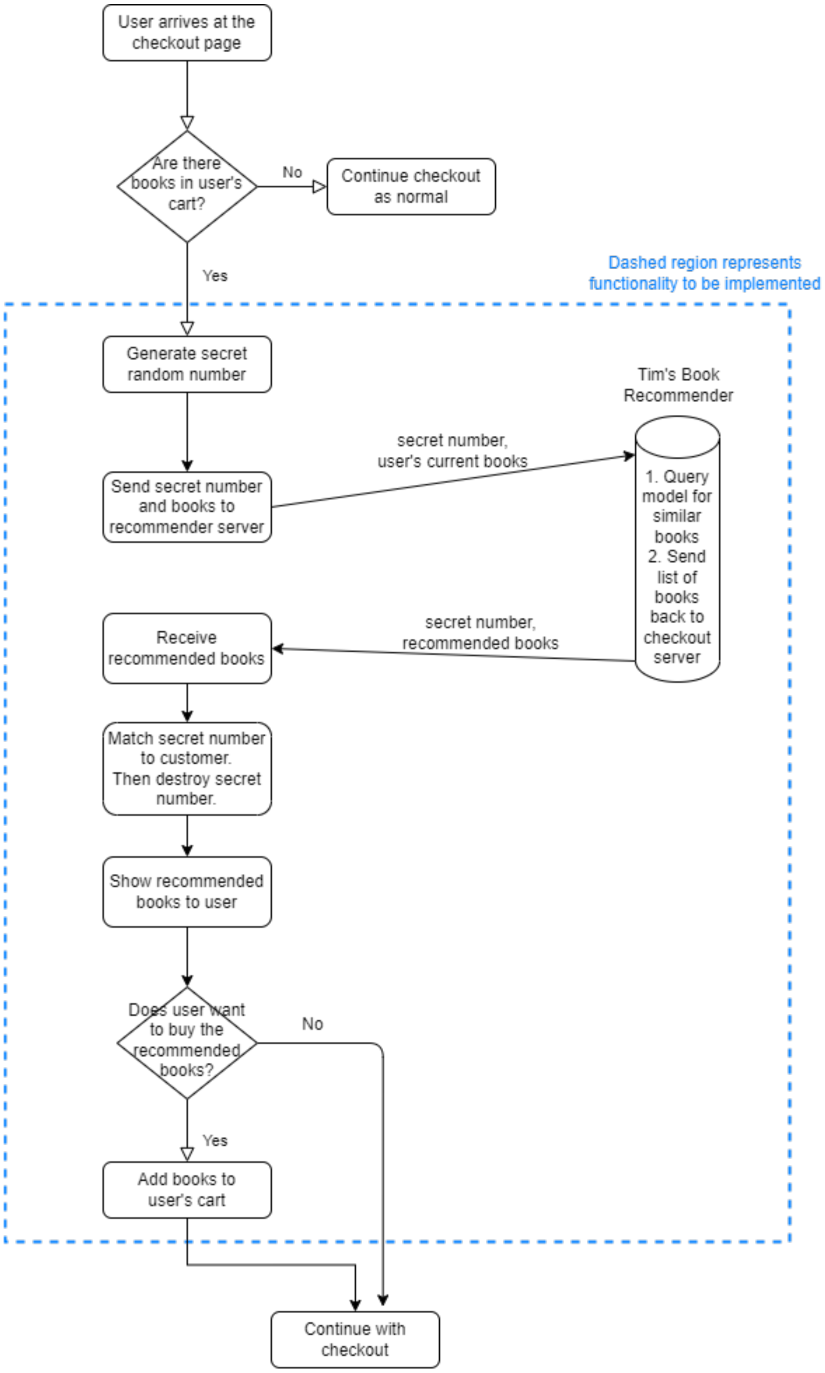
- Increased sales of books due to upselling customers on similar books
- Word-of-mouth about the new book recommendation feature

Possible negative outcomes:

- Increased time it takes customers to checkout, due to the extra step added
- Some customers will not agree with the recommendations of similar books
- Other companies may also start using unusual data sets to improve their own book recommendation systems, eroding Amazon's competitive edge

Figure 4

Flowchart showing new book recommender functionality to be implemented (in blue dashed region)


User arrives at the checkout page
Are there books in user's cart?
No
Continue checkout as normal
Yes
Dashed region represents functionality to be implemented
Generate secret random number
Tim's Book Recommender
secret number, user's current books
Send secret number and books to recommender server
1. Query model for similar books
2. Send list of books back to checkout server
secret number, recommended books
Receive recommended books
Match secret number to customer. Then destroy secret number.
Show recommended books to user
Does user want to buy the recommended books?
No
Yes
Add books to user's cart
Continue with checkout

Validation and Verification of the Data Product

A key part of any new technology rollout is to evaluate whether the solution actually meets the needs of the users. Even if the product developers and the sponsoring corporation and the project managers all agree that a product is wonderful, if the customers and users themselves don't benefit from the solution, then we can't call it a success. So ultimately it will the customers who tell us whether the new book recommendation system really and truly is a success.

In the case of implementing a book recommender in the Amazon checkout workflow, the ideal method of validating the product and verifying that it is meeting the customers needs would be a follow-up survey, such as an email sent to customers who purchased recommended books perhaps 1 week after purchase and again 1 month after purchase. The lead time of 1 week and 1 month gives customers a chance to start reading their new recommended books so that they are able to give sound judgment on whether they enjoyed the recommendation or whether they were disappointed by it. This valuable feedback would be an essential part of evaluating the validity of the book recommendation product.

As an additional step of customer satisfaction, customers who report that they were dissatisfied by the book recommendation could be offered a voucher good for a discount on their next book purchase at Amazon.com. As another possible remediation step, Amazon customers could adjust a setting in their Amazon account preferences to turn off the recommendation feature. After all, book recommendations should be something offered to customers as an additional service, not something forced on customers whether they like it or not.

Costs and Human Resources Requirements

The entirety of the software-development process for Tim's Book Recommender was conducted using the Python programming language.[21] According to the highly esteemed TIOBE[22] programming language survey, Python is the most popular programming language in the world as of July 2022. Furthermore, in the words of François Chollet, the creator of Keras, one of Python's artificial-neural-network and deep-learning libraries: "Python is by far the most widely used language for machine learning and data science" (Chollet, 2021, p. 20). In addition to Python, I used TensorFlow[23] and Keras,[24] two popular machine-learning libraries, to create and train the deep-learning model. All these software packages are open source and free of charge, so the cost of the programming environment is zero.

As for hardware, I used my existing desktop PC and Graphics Processing Unit (GPU), so these costs were also effectively zero.

Since the intended implementation location for this data product is Amazon's own servers, the cost of hosting is also effectively zero since Amazon owns Amazon Web Services (AWS),[25] a leading cloud-services provider with an approximately 50% share of the cloud infrastructure market in 2018.[26]

[21] https://www.python.org/

[22] https://www.tiobe.com/tiobe-index/

[23] https://www.tensorflow.org/

[24] https://keras.io/

[25] https://aws.amazon.com/

[26] "Owning almost half the world's public cloud infrastructure market, Amazon is the clear market leader" (Kinsta, 2022).

As regards human resources, the development of the data product itself was an individual, self-funded effort done by myself. Because it was a solo effort, I was able to take development only as far as an app prototype. With the expectation that Amazon will enter into a deal with me, it is my proposal that three experienced app developers be hired to perform the integration of the book recommender into Amazon's existing checkout pipeline. As noted above in the "Costs" section, I estimate that expenses of $50,000 will be adequate to cover the contracts for these three developers.

The following table details the personnel and funding requirements for each phase of the book recommendation model's development.

Table 1
Personnel and funding requirements, by phase

Phase	Personnel Required	Funding Required
Create and train a deep-learning book-recommendation model	1	$0
Develop prototype GUI app to demonstrate the model's capabilities	1	$0
Implement the deep-learning model into Amazon's existing customer checkout pipeline	3	$50,000

Milestones and Projected Timeline

There are, in a sense, two projects in play here. There is the project of creating a data product which I have already done, and there is the project of how to implement this data product into Amazon's current customer checkout process. With that in mind, I am here sharing two tables containing lists of milestones and timelines. The first table shows what has already happened. The second table shows how I would plan to implement this product for my hypothetical customer, Amazon.com.

This first table details the tasks involved in my project to create a book recommendation model using deep learning.

Table 2
Milestones and timeline for development project

Project Task	Dependencies	Resources Required	Start Date	End Date	Duration (days)
1. Download an archive containing all the articles from the Wikipedia website	none	self, computer	7/15	7/15	1
2. Extract the articles from the archive and prepare them for processing	1	self, computer	7/16	7/16	1
3. Write a program using Python to search all the articles and identify the articles that are about books	2	self, computer	7/17	7/18	2
4. Extract metadata details from the articles about books such as book title and book author and a list of the article's outgoing Wikipedia links	3	self, computer	7/19	7/20	2
5. Compose two data sets: - one with metadata about each book such as title, author, and outgoing links - one with all the possible outgoing links and their frequency of occurrence in the first data set	4	self, computer	7/21	7/22	2
6. Research suitable neural-network models for use in measuring the degree of similarity between elements in a large set	none	self, library	7/23	7/23	1
7. Program a machine-learning model using Python and Keras that creates a custom embedding using the outgoing links on the book pages as training data	6	self, computer	7/24	7/24	1
8. Select a loss function such that books whose articles have similar outgoing links are located near one another in high-dimensional space	7	self, computer	7/25	7/25	1
9. Train the model using data pairs consisting of a book and an outgoing link	5, 8	self, computer	7/26	7/27	2
10. Test the model using several examples, and use common sense to evaluate the model's usefulness to its hypothetical users	9	self, computer	7/28	7/28	1
11. Write user documentation and evaluate the success of the project and points of future improvement.	10	self, pencil and paper	7/29	8/1	4

This second table proposes a timeline and series of milestones for a project to implement the book recommender into Amazon's checkout process.

Table 3
Timeline and milestones for implementation project

Phase of Implementation	Dependencies	Resources Required	Start Date	End Date	Duration (days)
1. Hold a kickoff meeting between myself, my three contracted developers, and Amazon's representatives to review this timeline and milestone table	none	self, 3 devs, Amz reps	10/1	10/1	1
2. Evaluate Amazon's existing systems and select where to host the book recommendation model	1	self, 3 devs, Amz reps	10/2	10/4	3
3. Integrate the book recommendation function (see Figure XXX above) into Amazon's ecommerce platform, and turn it on	2	self, 3 devs	10/5	10/11	7
4. Perform regression testing to identify bugs and other errors that arise during integration	3	self, 3 devs	10/12	10/14	3
5. Additionally, review the ecommerce platform's systems logs and error reports to identify any unexpected side effects or bugs created by the introduction of the book recommendation functionality	3	self, 3 devs	10/15	10/21	7
6. Review sales data for the timeframe since the book recommendation feature was activated look for any immediate "bumps" or upward trends in book sales among customers who were given recommendations	3	self, 3 devs	10/22	10/28	7
7. Send survey questions via email to a small subset of customers who used the book recommendation service to ask their opinion of the quality of the book recommendations	6	self, 3 devs	10/29	10/30	2
8. Continue to monitor system logs and bug reports to ensure that the newly introduced feature is running smoothly	5	self, 3 devs, Amz reps	10/31	11/1	2

As shown in Table 3, the implementation and follow-up project is estimated to take 1 month, from October 1 to November 1.

C. Description of the Data Product

Descriptive and Prescriptive Methods

A well-designed application makes use of both descriptive and prescriptive data methods.

For this data product, a prescriptive (i.e., non-descriptive) method was used for the loss function employed while training the machine-learning model. The specific prescriptive method employed was a function called binary cross entropy. This is a standard loss function used primarily in binary classification tasks. For the purposes of our data product, the loss function wanted to answer the question “does this book and hyperlink go together, or not go together?” This was how the model was trained to recognize how similar two books were based on their list of hyperlinks.

For a descriptive method, I chose to employ three data visualizations in the main GUI’s dashboard that help the user to see how similar their original book is to other books that the model knows about, as well as how similar the 10 most popular genres are to each other. The *x*-axis and *y*-axis of the visualizations are labeled numerically in order to facilitate comparing data values across two or even three of the provided data visualizations. This is to assist orienting the user amidst the cloud of dot plots. (Recall that these graphs are showing 40,000 data points!)

The data visualizations serve two purposes. First, they give direct information to the user about the relationship between their original book and similar books. Second, the visualizations, especially the complex genre visualization, showcases the vast amount of information that the model is using for its similarity predictions. This showcasing of the large data set underlying the model’s reports is intended to boost the user’s confidence in the accuracy and complexity of the model’s recommendations.

Data Set Availability

The data set used to train the deep-learning model was a transformation of the Wikipedia data set that is publicly available[27] from the Wikimedia Foundation which oversees Wikipedia. The wealth of information in these Wikipedia “dumps” makes them an extremely attractive source of data either as a sole data set or as a data-laden complement to other information sources being used.

The Wikipedia dumps are generated on a monthly basis, so an additional advantage of using these dumps as a primary data set is that it’s possible to download a more recent dump and use it to re-train a model to take advantage of new information that has been added to Wikipedia. In the case of Tim’s Book Recommender, periodically re-training the neural-network model with a more recent Wikipedia dump means that newly published books as well as additional details that have been added to pre-existing books will make the model even more accurate and up-to-date, which ultimately benefits the end users by giving them superior recommendations.

[27] https://dumps.wikimedia.org/backup-index.html

Decision-Support Functionality

Without the assistance of artificial intelligence, people are able to cope quite well and make important decisions using the intelligence they were born with in their head plus the information they have learned in their years of experience of being alive. What artificial intelligence offers to the human race is a way of seeing patterns and identifying solutions that is orders of magnitude greater in terms of power and speed than what the ordinary human mind is capable of. In the case of buying and reading books, readers are certainly able to judge a book "by its cover" and make a snap decision on whether this book is similar to other books they've liked in the past or whether it will be an interesting and satisfying read. But this hypothetical book shopper doesn't have the contents of an encyclopedia at their disposal, so the basis on which they decide to buy or not buy the book is extremely limited.

This is where the deep-learning model can support the decision-making that book shoppers have to make when buying books. In a sense, human beings can "delegate" the bulk of the decision-making to a computer model that seeks out the kind of solutions that the person wants, which in this case is new books that are similar to an old book that the person likes.

An analogy would be a wheelchair. A wheelchair supports the mobility of people, but the person in the wheelchair still has the final say on where to go. Similarly, a book recommendation system can present a list of similar books to the reader, but this goes no farther than being a support role. Ultimately the shopper makes the decision whether to buy the book or not. This is why artificial intelligence merely supports decision-making by people, never supplanting it.

Featurization of the Data Set: Word Embeddings

As much as we talk about how "smart" and "intelligent" computers are, in reality they are little more than dumb rocks (i.e., the elemental material called silicon) with electricity running through them. Computers, not even deep-learning models, can "think" or "know" anything like we people can think about things and know things. I say this as something who thinks that artificial intelligence has enormous promise for the ways it can help people! But when it comes down to it, computers (and A.I. models included) only understand zeros and ones. Binary. We can use this binary language to represent things like integers and decimals and even letters and words and so on into even more complicated data like photographs and musical recordings. But to the CPU and GPU of a computer, it eventually has to be turned into numbers for it to be able to process it.

That's why one of the key parts of developing an A.I. model is to determine what kind of representation the data needs to have for it to be been into the model for training. It's only partly true that we feed a list of hyperlinks into the data model. The data model can't directly process a list of words or links, since they aren't numbers. So the first step is to convert our list of hyperlinks for each Wikipedia article into a numeric format the machine-learning software will be able to work with.

In the case of textual data, this usually means converting the data set into numbers by using an "embedding." Artificial-intelligence experts and university professors Russell and Norvig explain word embeddings in the context of artificial intelligence in this way: "How should we encode a word into an input vector x for use in a neural network? . . . We would get better generalization if we reduced this [word] to a smaller-size vector, perhaps with just a few hundred dimensions. We call this smaller, dense vector a word embedding: a low-dimensional vector representing a word. Word embeddings are learned automatically from the data. . . . Each one is just a vector of numbers" (2021, pp. 856–857). This industry-standard technique of converting textual data into numeric embeddings was used for Tim's Book Recommender.

The way a neural network "learns" is by creating an enormous network of individual nodes that have various numeric values on them. The neural network accepts input from the user on one side of the network and filters the tiny bits and details of data through this network until a value comes out the other side. During training, if the value that comes out is accurate, then we preserve the numeric values on all the little nodes in the network. But if the value that comes out is inaccurate, then we slightly modify some or all of the numeric values in an attempt to obtain better output on the next round of training. Since a neural network deals with numbers, we have to turn our data into numbers so that it can be sent through the network.

Figure 5 gives an example of a word embedding such as was used for the book and hyperlink data set in Tim's Book Recommender. As shown, it is a sequence of 50 different floating point (i.e., decimal) numbers. Although a list of high-precision numbers like this doesn't make much sense to us, to a computer it could be the representation of the book title *Harry Potter and the Philosopher's Stone* or the representation of a hyperlink to the Wikipedia article on broomsticks. What's important is that this is the computer's native language and it can use information in this format to feed and train a neural network.

Figure 5
Sample 50-dimensional vector embedding

```
[ 0.223573651, 0.5484899154, 0.2275413954, 0.5618424183,
0.5205047654, 0.3037905001, 0.4316202017, 0.2621903757,
0.1690598478, 0.5871165303, 0.4590439666, 0.8071389936,
0.5043781721, 0.5675473683, 0.0700235788, 0.2869563115,
0.4053817546, 0.8551566225, 0.0715572819, 0.9466886786,
0.0134743249, 0.0496052688, 0.7450363617, 0.0051908428,
0.2048556416, 0.3585506643, 0.5171062078, 0.0923798967,
0.4196956103, 0.5357801278, 0.1396698006, 0.7685441921,
0.2041120598, 0.0268994785, 0.7387828193, 0.6594846080,
0.0667189141, 0.4325159303, 0.0509194237, 0.9765230176,
0.5553772244, 0.5235080957, 0.5208397049, 0.5526474069,
0.4608745424, 0.4253502305, 0.2553688455, 0.5943323936,
0.5157193024, 0.6949273149 ]
```

Methods and Algorithms for Data Exploration and Preparation

Computers are such powerful tools because they are capable of following their coding instructions to the letter, never getting tired and never deviating from their assigned protocol. Unfortunately that strength strikes two ways: it also means that if we give a computer noisy or bad data to work with, it will continue to implement its coding on that bad data even if the end result is going to be inaccurate or just plain wrong. It's up to the programmer or data scientist to give the computer the best data possible so that the computer can focus on its job of executing the machine-learning code.

In the case of my data set, that meant depending on my own machine-learning expertise and experience to know which details from the data set ought to be excluded so that the deep-learning process could focus on just those details that will contribute to the model's usefulness. Since the purpose of the model is to compare the similarity of books based on the links found on their Wikipedia pages, that meant that I should exclude all the parts of the Wikipedia articles except for the list of links. That also meant that to ensure that the names on the hyperlinks were as consistent as possible, that I should remove low-information words (also termed "stop words") such as "a," "an," "the," and "of" to avoid distracting the machine-learning process with irrelevant function words. It also meant lowercasing the data since the distinction between uppercase and lowercase was not relevant for the purposes of detecting similarity. All these operations fall under the heading of preprocessing the data in such a way that the machine-learning model is set up for success.

The primary method that was used during this phase of data preparation was the use of "search and replace" with what are called "regular expressions."[28] Traditional search-and-replace methods require the user to input the exact word to be searched for and exactly what to replace it with. For example, if we have the text "The quick brown fox jumped over the lazy dog" and we searched for "fox" and replaced it with "goat" then the resulting text would be "The quick brown goat jumped over the lazy dog." The method surgically replaced one word with another.

Regular expressions take searching and replacing to the next level. Rather than searching for a specific word like "fox," we can search for patterns of words. For example, the pattern to describe "all words that start with the letter 'f'" looks like this in regular expression format: `\bf[a-z]+\b` . Each backslash, letter, plus sign, and square bracket has a special meaning that encodes the textual pattern we're looking for. This expression would find "fox," but it would also find "fawn" and "feline."

To successfully prepare a data set of 40,000 records that each contain a list of hyperlinks (for a total count of 1.2 million hyperlinks!), there was no way I could come up with a list of all the exact words to search for and replace. But by leveraging the power of regular expressions, I described word patterns that I wanted to delete and Python's built-in capability for accepting regular expressions with its search-and-replace method would do the hard work for me. For example, here are a sampling of some of the regular expressions that were used to delete the definite and indefinite articles at the beginning of the names of hyperlinks as well as the preposition "of" anywhere it appeared in the title.

Table 4
Sample regular expressions used to clean and preprocess data set

Description	Regular Expression	Sample Original Text	Sample Processed Text
Delete indefinite article "a" from beginning of string	^[Aa]\b	A Tale of Two Cities	Tale of Two Cities
Delete indefinite article "an" from beginning of string	^[Aa]n\b	An Unexpected Guest	Unexpected Guest
Delete definite article "the" from beginning of string	^[Tt]he\b	The New York Times	New York Times
Delete preposition "of" anywhere in string	\bof\b	Tale of Two Cities	Tale Two Cities

[28] "A 'regular expression' . . . is a sequence of characters that specifies a search pattern in text. Usually such patterns are used by string-searching algorithms for "find" or "find and replace" operations on strings, or for input validation. Regular expression techniques are developed in theoretical computer science and formal language theory" (Wikipedia, 2022).

This method of using regular expressions combined with search-and-replace made the names of the hyperlinks more consistent and prevented the machine-learning model from thinking, for example, that "The New York Times" and "New York Times" were two different newspapers when they are actually the same newspaper.

Data Visualization Functionality

They say a picture is worth a thousand words, and even avid readers who are shopping for their next book can appreciate the high information density that comes with a well-designed diagram or data plot. In the case of Tim's Book Recommender, I wanted to include data visualizations to help explain how the deep-learning model was making its recommendations. While many people are inclined to agree that artificial intelligence can sift through data and give us answers far more quickly than the human mind can, it's still important in my opinion to teach people how computers are doing it and in some sense provide evidence or proof of the sort of work the computer is doing behind the scenes. Although the average person doesn't understand the science behind multidimensional word embeddings or regular expressions, anyone can look at a graph and observe that some points are closer than other points and see that there is some mathematics going on in the background that proves that some books are more similar than other ones.

Although the neural-network model uses vectors containing 50-dimensions of numerical data to represent the similarity of books, it's possible to convert these 50-dimensional data structures into a more familiar 2-dimensional "(*x*, *y*)" format that can be shown on a graph or plot. This process of changing 50-dimensional data to 2-dimensional data while preserving its important characteristics is called "dimensionality reduction," and what makes it so special is that the book similarities that are present in 50 dimensions are still there in 2 dimensions.

The figures on the next page demonstrate the three kinds of data visualizations that are available on the GUI interface's dashboard. The selected book is *A Tale of Two Cities* by Charles Dickens. The first visualization, Figure 6, is a plot of the selected book in blue, with the books ranked as most similar to that book in green. Finally the remaining 40,000 or so books that the model knows about are plotted with yellow dots in the periphery. This visualization shows us that some of the similar books are more similar and some are less similar.

Figure 6
Data visualization of a book and its similar books

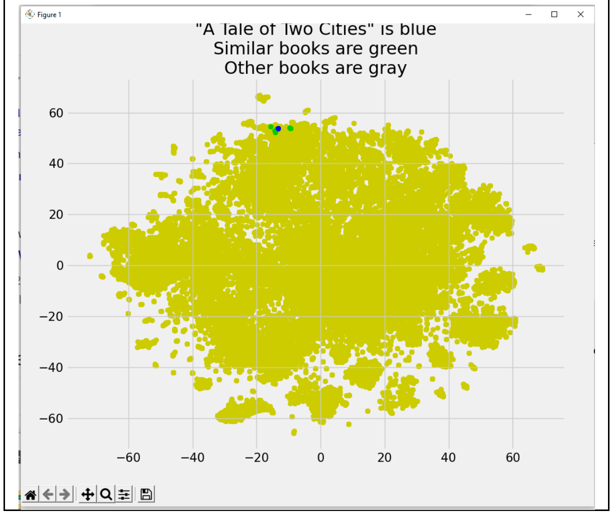
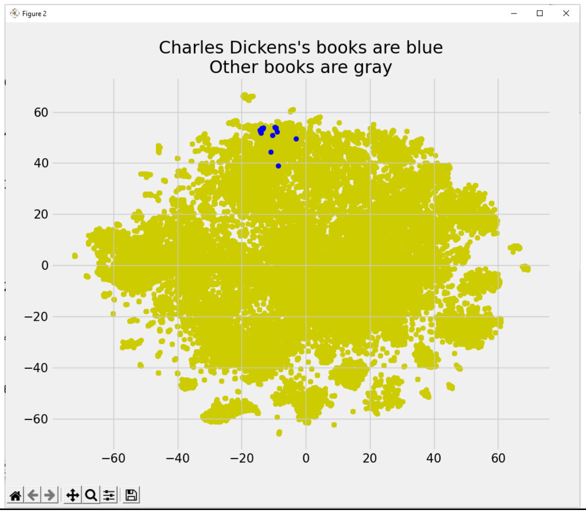
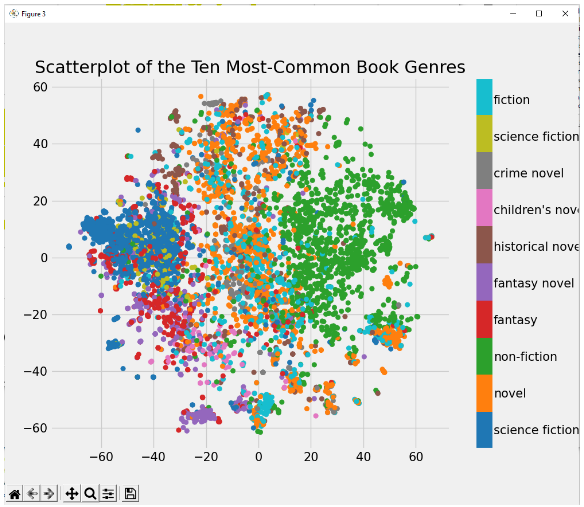


Figure 7
Data visualization of books by the same author



The second visualization, Figure 7, plots the selected book with other books by the same author. This visualization reveals that books are not necessarily exactly similar even when they are written by the same author. We see from this visualization that some of Charles Dickens's books are somewhat far away. We might naively assume that all of Charles Dickens's books would be in a tight cluster, but the reality is that Dickens wrote books on a variety of themes and topics. If we were to visit the Wikipedia pages for all of Dickens's books, we would see some overlap in the kinds of hyperlinks but also many differences. It is the differences that account for the dispersal of data points in this second visualization.

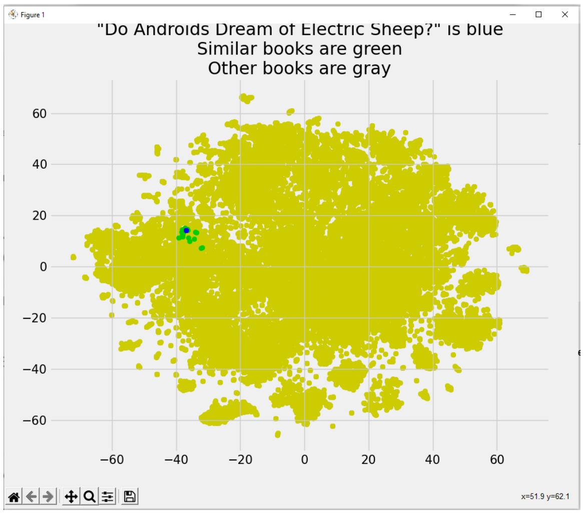
Figure 8
Genre map of 10 most common literary genres



Finally there is a third data visualization available in the app prototype, shown in Figure 8, that assigns different colors to the top ten most common book genres and plots all the books in these genres on the same graph. The intention with this visualization is to provide a map of sorts so that users of the GUI can see what genres are prevalent in the regions where their selected book and its similar books are located.

The dots for *A Tale of Two Cities* and its similar books are all located at the top of the dot graph, and we can see the prevalence of orange dots in that top area, representing the "novel" genre.

Figure 9
Data visualization of sample book



The genre map showed us that the region on the left side of the dot graph has a lot of dark blue, which stands for the “science fiction” genre. So we can test the accuracy of the model by choosing a science fiction novel and seeing if the resulting dot is in that dark blue area or not. A popular science fiction novel is *Do Androids Dream of Electric Sheep?* by Philip K. Dick. If we select this book and show the data visualization for this book and similar books we get the graph shown in Figure 9. We see that Dick’s science fiction book is near the intersection of coordinates (–40, 20). Referencing our genre map, we see that this location is part of the cloud of dark blue dots representing science fiction novels. With this simple test, we see that the machine-learning model’s understanding of books and genres seems fairly accurate.

Interactive Query Functionality

Graphics designers and programmers face the ongoing challenge of helping users understand how to use the interface to an application. Designers and marketers praise applications that are “intuitive” and “easy to use.” The objective is that a person should be able to start using an app and have the app “do what’s right” and “do what’s expected.” To put things another way, a user shouldn’t have to think about how to use an application—the user should just be able to start using it.

With these assumptions in mind, I wanted the interactive query functionality of the GUI app to be as easy to use as possible. I didn’t want the user to have to read a laundry list of instructions before making a query. I wanted the user to be able to just start typing and for the app to make perfect sense as the user went along. Early on, I thought about having the user type in the entire title of a book, but I discarded this idea because not everybody remembers the exact title of a book. (Plus, if the user made a spelling mistake, then it would be very difficult to try to “fuzzy match” what the user typed with the name of an actual book.)

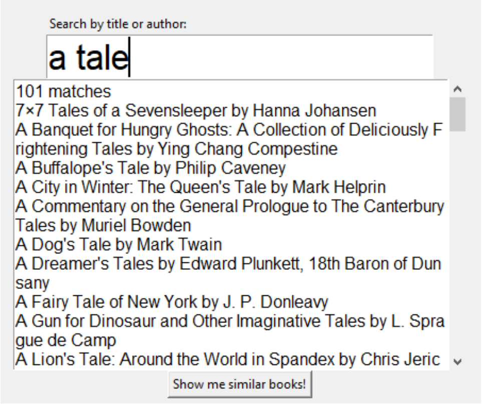
Another possibility was to have the user interact using mouse clicks to choose a book graphically. This strategy makes sense in a lot of ways, but when there are 40,000 book titles to choose from, the user would have to do a lot of scrolling around or clicking and zooming in and out to locate the book of their choice. This way didn’t seem feasible either.

In the end I designed the interactive query function to have a hybrid interface. The user types in some keywords, and a results box shows titles that match the keywords that the user typed in. The user can type more words from the title or parts of the authors name, and the query system behind the scenes can use those additional search terms to narrow down the list of books the user is looking for.

For example, let’s suppose the user is looking for the novel *A Tale of Two Cities*. Figures 10–14 on the next page show the interactive search results changing as the user types each word.

Figure 10

Search results in progress, part 1

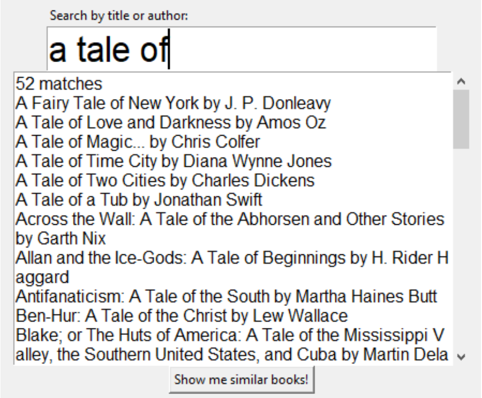


Search by title or author: a tale

101 matches
7×7 Tales of a Sevensleeper by Hanna Johansen
A Banquet for Hungry Ghosts: A Collection of Deliciously Frightening Tales by Ying Chang Compestine
A Buffalope's Tale by Philip Caveney
A City in Winter: The Queen's Tale by Mark Helprin
A Commentary on the General Prologue to The Canterbury Tales by Muriel Bowden
A Dog's Tale by Mark Twain
A Dreamer's Tales by Edward Plunkett, 18th Baron of Dunsany
A Fairy Tale of New York by J. P. Donleavy
A Gun for Dinosaur and Other Imaginative Tales by L. Sprague de Camp
A Lion's Tale: Around the World in Spandex by Chris Jeric

Figure 11

Search results in progress, part 2

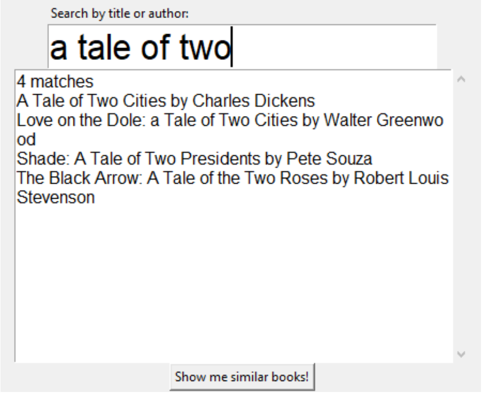


Search by title or author: a tale of

52 matches
A Fairy Tale of New York by J. P. Donleavy
A Tale of Love and Darkness by Amos Oz
A Tale of Magic... by Chris Colfer
A Tale of Time City by Diana Wynne Jones
A Tale of Two Cities by Charles Dickens
A Tale of a Tub by Jonathan Swift
Across the Wall: A Tale of the Abhorsen and Other Stories by Garth Nix
Allan and the Ice-Gods: A Tale of Beginnings by H. Rider Haggard
Antifanaticism: A Tale of the South by Martha Haines Butt
Ben-Hur: A Tale of the Christ by Lew Wallace
Blake; or The Huts of America: A Tale of the Mississippi Valley, the Southern United States, and Cuba by Martin Dela

Figure 12

Search results in progress, part 3

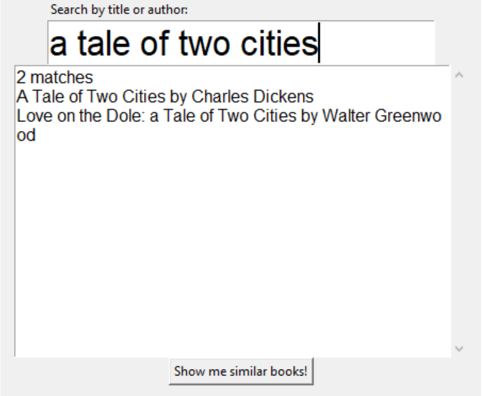


Search by title or author: a tale of two

4 matches
A Tale of Two Cities by Charles Dickens
Love on the Dole: a Tale of Two Cities by Walter Greenwood
Shade: A Tale of Two Presidents by Pete Souza
The Black Arrow: A Tale of the Two Roses by Robert Louis Stevenson

Figure 13

Search results in progress, part 4



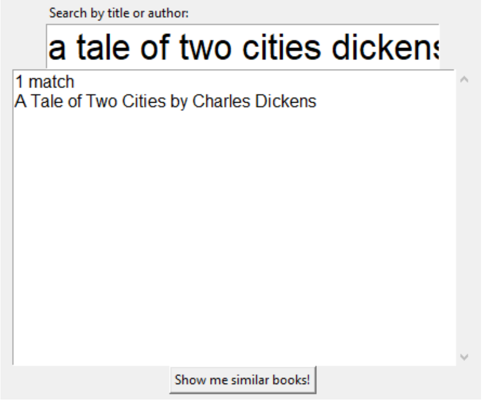
Search by title or author: a tale of two cities

2 matches
A Tale of Two Cities by Charles Dickens
Love on the Dole: a Tale of Two Cities by Walter Greenwood

With each new keyword the user types, the list of search results shrinks as the list of possible titles becomes shorter and shorter. If the user wanted to be absolutely sure of finding the exact book, they could even add the author's name, and in this case the search result now becomes a single result:

Figure 14

Search results in progress, part 5



Search by title or author: a tale of two cities dicken

1 match
A Tale of Two Cities by Charles Dickens

I chose to implement this hybrid interactive search functionality for two reasons. First, by having the search results be interactive the user can stop adding keywords as soon as they see the book they want to choose. This helps the user avoid a lot of unnecessary typing. Second, the dynamic nature of the changing search results gives the user confidence that they are using a sophisticated piece of software that is finely-tuned to be able to understand and process the input that the user is giving it. When we ask a user to trust the recommendations provided by Tim's Book Recommender, we don't want to underestimate the power of "showing off" the technological prowess that the app is capable of. When a user sees how quickly and intelligently the app is able to look up and find the book they're looking for, they are going to be that much more likely to believe that the actual recommendation functionality is likewise well-designed and reliable.

Methods and Algorithms for Machine Learning

The heart and soul of this book recommendation system is the use of machine learning to train a model to recognize the similarity between any two books by measuring the degree of similarity between the hyperlinks on the books' Wikipedia pages. With 40,000 books in total and 1.2 million individual hyperlinks, it would be frankly impossible for any human being to be able to make useful sense of this flood of data in order to produce lists of similar books on demand. But on a modern personal computer outfitted with a state-of-the-art GPU, the training calculations can be performed in 15 minutes, and subsequent use of the trained model can return results for any query in about 5 minutes.

The machine-learning method that I chose for this project is called deep learning. Deep learning is distinguished from other kinds of machine learning by the use of a network of individual nodes called neurons that are organized into layers of separate networks. Information flows into the first layer of neurons where it is transformed and passed into the next layer. After a certain number of layers, the resulting information comes out of the model. The "deep" in deep learning refers to the interior layers of the model, which can contain upwards of millions of neurons. Believe it or not, there are aspects that computer scientists do not fully understand of how these highly complex networks are able to modify themselves efficiently in order to produce increasingly accurate results. So another meaning of the "deep" in "deep learning" is that some aspects of these A.I. models is beyond our comprehension.

I used a relatively straightforward model design for the purposes of this data product. The data set about books and hyperlinks was transformed into "embeddings," which are a numeric representation of textual data. The numbers were fed into a model containing three layers. The first two layers used a total of about 4 million data values that could be tweaked and adjusted while training the model. The large number of data variables that must be tuned and changed during training is the reason why powerful Graphics Processing Units (or GPUs) are so commonly used in machine learning. GPUs are designed to process millions of pixels per second, and with very little retooling they can efficiently process millions of other kinds of data such as neurons.

The last layer of the model is an output layer that represents the model's capability to compare two books and assign a similarity value to the relationship. When this comparison is performed 40,000 times, it's possible for the model to rank all 40,000 books in its repertoire in terms of how similar each book is to all the other books.

Evaluation of the Data Product's Accuracy

The accuracy of the data product was evaluated in three ways. And in the hypothetical circumstance of Amazon deciding to incorporate Tim's Book Recommender into its checkout pipeline, there would be two additional ways we could validate the product's accuracy.

When it comes to machine learning, testing the accuracy of the results that are coming out is an integral part of every step of the process. As the first and primary method of evaluating the data product's accuracy, the deep-learning model was trained by partitioning the data set into training data and validation data. During the 15 epochs of training, validation data was regularly used to guide the adjustment being made to the trainable parameters in the model as part of the overall objective of minimizing the loss function. At the conclusion of training, additional data (called the testing set) that the model hadn't seen yet was used to finally test the model for accuracy.

Secondly, I entered the names of books that I was well familiar with and I was able to evaluate (as an avid reader) whether the results shown by the model were or were not in fact representative of similarity. For example, besides the examples already given of *Harry Potter*, *A Tale of Two Cities*, *Do Androids Dream of Electric Sheep?*, I also entered titles like *The Name of the Rose* by Umberto Eco and *Dancing After Hours* by Andre Dubus. In all cases, the recommendations given by the model were either immediately known to me as good, similar recommendations or, upon researching the recommended books, I could see why the model found them similar.

A third way of vetting the accuracy of the model was to generate data visualizations of several books and their similar books and compare the locations of those books on the dot graphs to the genre map that can be produced by the app prototype's GUI. Here again, books that were of a certain genre were located in the same area that the genre map would predict that these books would be located in.

To address on the hypothetical example of this data product being implemented into Amazon.com's checkout procedure, there are two main ways that we would use that outcome to continue to assess the accuracy of the book recommendation system. First, we would measure how often customers are adding the recommended books to their cart. In the parlance of ecommerce, this is called "conversion."[29] If we see a pattern where customers are repeatedly adding the recommendations to their shopping cart, we can interpret that as evidence that customers are satisfied or are at least genuinely curious about the recommendations. And on the contrary, if we see that even when customers are shown recommendations that they consistently turn the recommendations down, we could take that as a signal that our recommendations are not being welcomed. In either case, the checkout data showing what percentage of recommendations are being acted upon would give us valuable business intelligence as to whether the model is working as we would wish or whether some adjustment or redesign is warranted.

Second, I propose for there to be at least one email survey sent to Amazon customers who purchase additional books based on the recommendations they were given. This "human touch" of getting feedback from the actual customers would complement the more data-driven validation steps described such as checking sales figures and conversion rates.

[29] "*conversion*, noun. An online advertising performance metric representing a visitor performing whatever the intended result of an ad is defined to be" (Wiktionary, 2022).

Security Features

Security is always a factor when it comes to ecommerce. The advantage of my data product Tim's Book Recommender is that it is intended to be implemented in the checkout process for an established ecommerce website, meaning that typical security features are already in place, such as HTTPS and SSL, and bank-grade encryption is already employed when the customer's forms of payment are being used.

But a security feature that might not be in place is that of protecting the user's right to privacy about what books they either are reading or might want to read. It is for this reason that I have designed Tim's Book Recommender from the ground up to be hosted on a separate computer server from the rest of the ecommerce hardware, so that by use of a secret number we can completely anonymize the book recommendation process. This way any information about what books the customer has already purchased is not duplicated to the book recommendation server, and the recommendations that the book recommender makes are linked briefly to an anonymous secret number instead of being permanently associated with the customer's account.

We live in an age where on social media people's reputations can be affected simply by sharing what books or movies they have seen or might like to see. The court of public opinion is at its all-time strength, and the speed at which information flies on the Internet only accelerates the risk that details about what we think or believe or read will be misused or manipulated. We are still learning about the ways that artificial intelligence systems use the information that we give it. Who is to say that if we attach a customer's name to our query for book recommendations that the deep-learning system might find a way to start using that customer's name in its recommendations or, heaven forbid, share that customer's name in the results of other customer's recommendations. The risk is too great to take, and so erecting a strict firewall between the customer's details and the book recommendation system seems appropriate not only from an ethical stance but also from the perspective of ensuring that the neural network is not contaminated by irrelevant details such as other customers' names.

Having good security is not just good technology policy, it is also a good selling point for current and future customers. Given a choice between two ecommerce platforms, customers are likely to choose one that they feel respects their privacy and doesn't sell their details to the highest bidder. In the hypothetical case of Amazon incorporating Tim's Book Recommender into their systems, I would recommend advertising the strong privacy guarantees of our machine-learning model as a specific benefit and feature that we offer on top of making excellent book recommendations.

Product Monitoring and Maintenance

Ongoing monitoring and maintenance of the system are critical parts of the implementation. Just because the system is installed and switched on doesn't mean that the work is done. On the contrary! Changes to other parts of the checkout pipeline might have side effects that require bug fixes to the book recommendation system. Or we might find out that there are subtle bugs that only occur infrequently. For this reason we can't just monitor the system for a week and stop there. Monitoring has to be continuous for the lifetime of the system.

As examples of the kind of adequate monitoring that would need to be put in place, we would want to periodically search the website logs for warning signals such as the recommendation system taking too much time to return results. When a customer is trying to check out, the last thing they want is for an extra step to delay them when they are ready to complete the sale. Computer logs that show how long each step is taking would alert IT personnel that there might be an issue with the network connection between the checkout server and the book recommendation server.

Another important way of monitoring the system would be to regularly have employees walk through the process as if they were a customer so that they can “see what the customer sees.” We can’t always count on customers to call or email when they have a problem. Oftentimes people will have a problem on a website and just give up instead of reporting it, and it’s difficult to identify when people are just giving up and going away. But by pretending to be a user and conducting the typical actions of adding a book to our cart, reading the recommendations, and sometimes accepting the recommendations and sometimes declining them, quality testers can see for themselves if there are any unwarranted bugs or interactions that might be affecting the customer experience.

Maintenance is a broad topic, and while checking and testing of the system (as described above) is part of it, another aspect is bringing the recommender system up to date as needed. That might mean updating relevant software packages such as Python, TensorFlow, and Keras that are used to compute the book similarities. But maintenance also includes ensuring that the model itself is up to date. The model is based on the hyperlinks that fill the pages of book-related articles on Wikipedia. Every month there are additional books being published and new Wikipedia articles being written to describe them. There are also changes to existing Wikipedia articles, such as new insights or recent news or revivals of interest in older books. These changes will probably result in some hyperlinks being deleted and some hyperlinks being added. This changes the kinds of connections between each book and all the other books around it, and if we want the book recommender to stay relevant we need to capture these changes on an ongoing basis by retraining the model with a fresh dump of Wikipedia, perhaps on a regular schedule every 6 months or so.

Finally, word of mouth can make or break a new product, and even if the initial buzz about the book recommender is good, if the recommender starts to break down or give poor results because the monitoring and maintenance work isn’t being done, then all that goodwill and positive reception can be eroded very quickly. What customers prize as much as quality is reliability. After all, what good is an A.I. book recommendation system if it times out regularly or returns errors? It would be better to have no recommendations at all than flaky and inconsistent ones.

Dashboard with Three Visualizations

I created three data visualizations that can be accessed from the prototype GUI’s dashboard and displayed to the user in floating windows, as shown in Figure 15. The purposes of these visualizations are threefold.

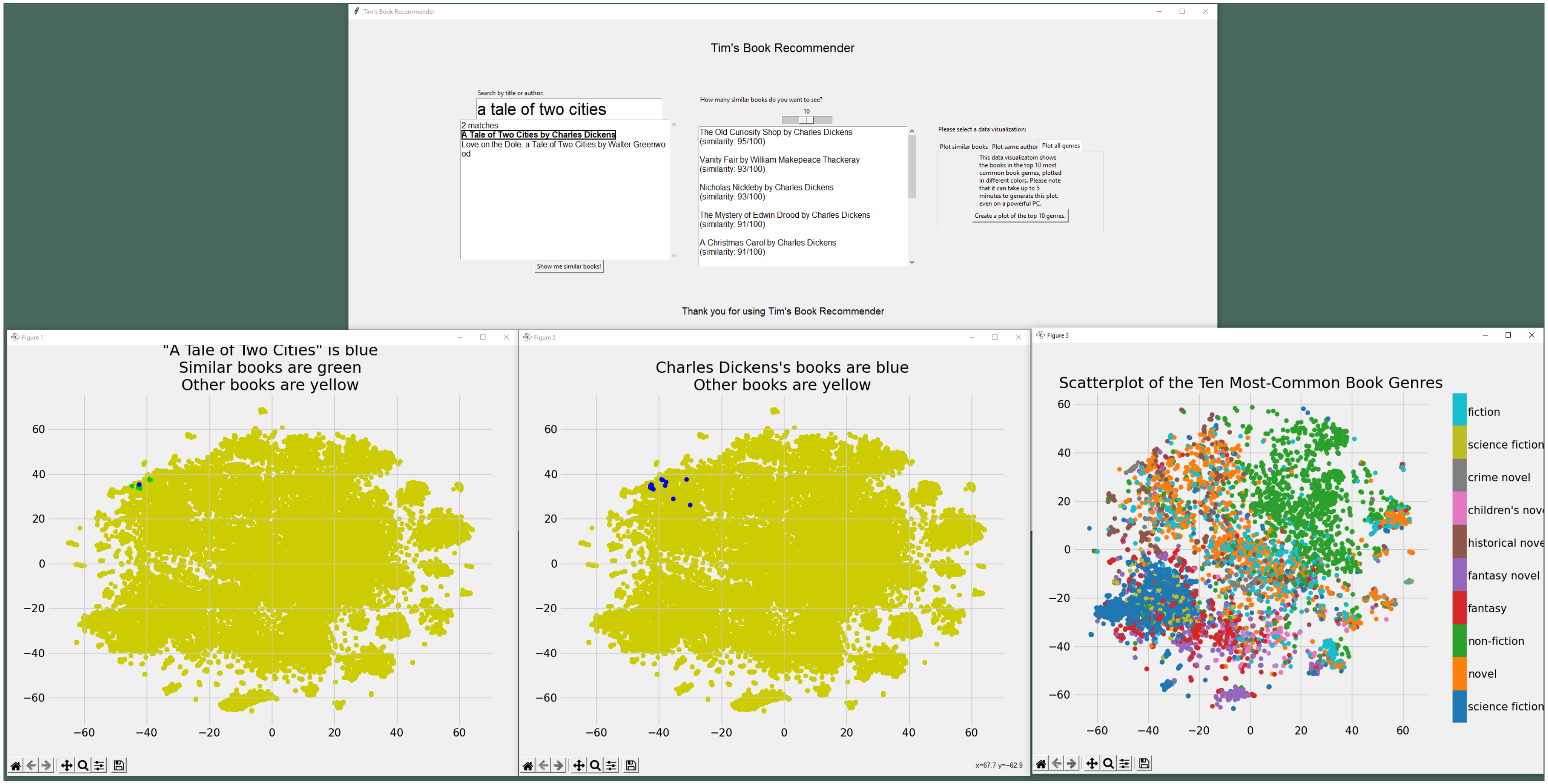
First, since a picture is worth a thousand words, having a dot graph showing the relative closeness of books that are similar versus books that are not similar offers the user a vivid and memorable impression of the similarity score of the books.

Second, although the list of recommended books is the primary benefit of Tim’s Book Recommender, another goal might be to teach the user about the variety and range of the content of books written by a single author. For example, the books from one author might be in a tight cluster, indicating that they mostly wrote about the same themes. Other authors, by contrast, might have their books scattered all over the genre map, illustrating that they explored a wide variety of topics and concepts.

Third, these data visualizations are a jumping-off point for possible future additional features of Tim's Book Recommender. For example, I can imagine allowing users to zoom in extremely close and see dotted lines connecting similar books, allowing the user to "travel" as it were from book to book in the same way that people drive on highways from city to city. Exploring the landscape of books graphically in this way might open all new ways of thinking about reading as well as new opportunities to discover new books to read.

Figure 15

Screenshot of GUI dashboard with three data visualizations for A Tale of Two Cities



Tim's Book Recommender

Search by title or author: a tale of two cities

2 matches
A Tale of Two Cities by Charles Dickens
Love on the Dole: a Tale of Two Cities by Walter Greenwood

How many similar books do you want to see? 10

The Old Curiosity Shop by Charles Dickens
(similarity: 95/100)

Vanity Fair by William Makepeace Thackeray
(similarity: 93/100)

Nicholas Nickleby by Charles Dickens
(similarity: 93/100)

The Mystery of Edwin Drood by Charles Dickens
(similarity: 91/100)

A Christmas Carol by Charles Dickens
(similarity: 91/100)

Please select a data visualization:

This data visualizatoin shows the books in the top 10 most common book genres, plotted in different colors. Please note that it can take up to 5 minutes to generate this plot, even on a powerful PC.

Thank you for using Tim's Book Recommender

D. Additional Documentation

Business Vision and Requirements

Book-reading consumers have been frustrated by the flood of books available for sale on ecommerce websites, and they demand better recommendations of what books to buy and read next. Traditional methods of book recommendations such as recommending other books by the same author or other books in the same genre have ultimately proved unsatisfying because similar bibliographical details or similar positive reader reviews are not reliable indicators whether two books are going to be similar or spark a similar level of interest or excitement in readers.

The time has come for a new way to evaluate the similarity between books, and my proposed method is based on a deep-learning model that was trained using the hyperlinks present on Wikipedia pages for 40,000 books. The similarity between any two books is based on the similarity between their Wikipedia page hyperlinks connecting to all kinds of different topics, concepts, places, people, and events that have some connection to the book. These links might be about things that happened inside the book, or the links might have to do with the real world, such as how readers responded to the book or how events the book bear a striking resemblance to current world events. These hyperlinks are capable of such variety and diversity that they are a superior method of comparing the similarity of books compared with the old way of primarily looking at just author and genre.

A book-recommendation system that looks below the surface for the hidden connections between two books will present readers with astonishing and interesting new kinds of recommendations that will delight readers who are tired of simplistic kinds of recommendations. The deep-learning model used by Tim’s Book Recommender is just such a recommender system able to finally meet readers’ wishes for complex recommendations based on a highly multifaceted comparison of the similarities between books.

Data Sets

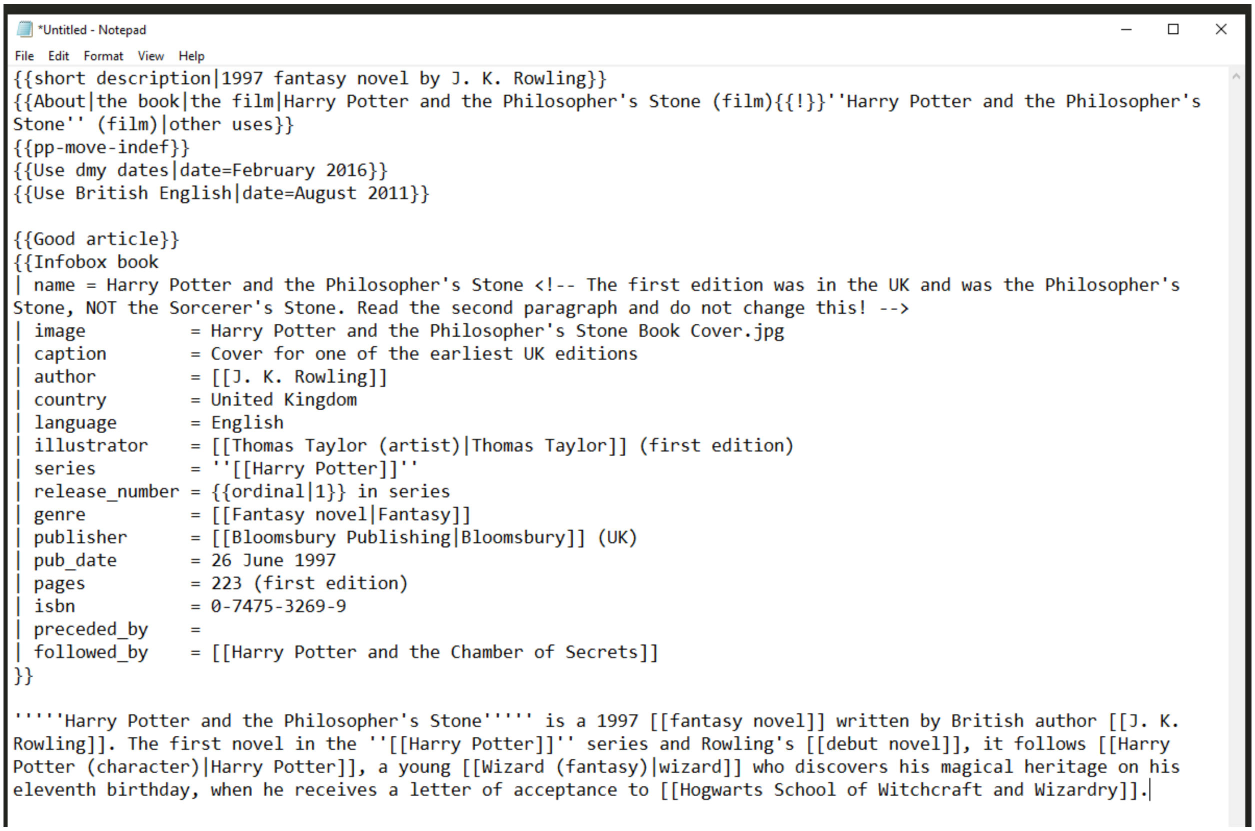
The data set that was used to train the deep-learning model started out as a 20 GB compressed archive of the entire Wikipedia website that the Wikimedia Foundation calls a “dump” (as in the entire contents of Wikipedia just “dumped” into one file). I downloaded this file as a BZ2 compressed file, and when I uncompressed it on my home computer it expanded into an 87 GB XML file that contained the text and internal article structure for all 6.5 million Wikipedia articles.

This is, of course, information overload for what I planned to do. These 6.5 million articles cover almost every topic known in human existence. I only want the articles about books.

So my next step was to extract from the XML only the articles about books. This resulted in 40,000 files that were just the articles about books. Figure 16 shows a sample of the source code of one of these articles, in this case the beginning of the data record about the first *Harry Potter* book. All 40,000 Wikipedia articles were written in this compact yet readable markup language that Wikipedia uses behind-the-scenes for all its articles.

Figure 16

Wikitext source code for Wikipedia article about Harry Potter and the Philosopher's Stone



```
{{short description|1997 fantasy novel by J. K. Rowling}}
{{About|the book|the film|Harry Potter and the Philosopher's Stone (film){{!}}''Harry Potter and the Philosopher's
Stone'' (film)|other uses}}
{{pp-move-indef}}
{{Use dmy dates|date=February 2016}}
{{Use British English|date=August 2011}}

{{Good article}}
{{Infobox book
| name = Harry Potter and the Philosopher's Stone <!-- The first edition was in the UK and was the Philosopher's
Stone, NOT the Sorcerer's Stone. Read the second paragraph and do not change this! -->
| image          = Harry Potter and the Philosopher's Stone Book Cover.jpg
| caption        = Cover for one of the earliest UK editions
| author         = [[J. K. Rowling]]
| country        = United Kingdom
| language       = English
| illustrator    = [[Thomas Taylor (artist)|Thomas Taylor]] (first edition)
| series         = ''[[Harry Potter]]''
| release_number = {{ordinal|1}} in series
| genre          = [[Fantasy novel|Fantasy]]
| publisher      = [[Bloomsbury Publishing|Bloomsbury]] (UK)
| pub_date       = 26 June 1997
| pages          = 223 (first edition)
| isbn           = 0-7475-3269-9
| preceded_by    =
| followed_by    = [[Harry Potter and the Chamber of Secrets]]
}}

'''''Harry Potter and the Philosopher's Stone''''' is a 1997 [[fantasy novel]] written by British author [[J. K.
Rowling]]. The first novel in the ''[[Harry Potter]]'' series and Rowling's [[debut novel]], it follows [[Harry
Potter (character)|Harry Potter]], a young [[Wizard (fantasy)|wizard]] who discovers his magical heritage on his
eleventh birthday, when he receives a letter of acceptance to [[Hogwarts School of Witchcraft and Wizardry]].
```

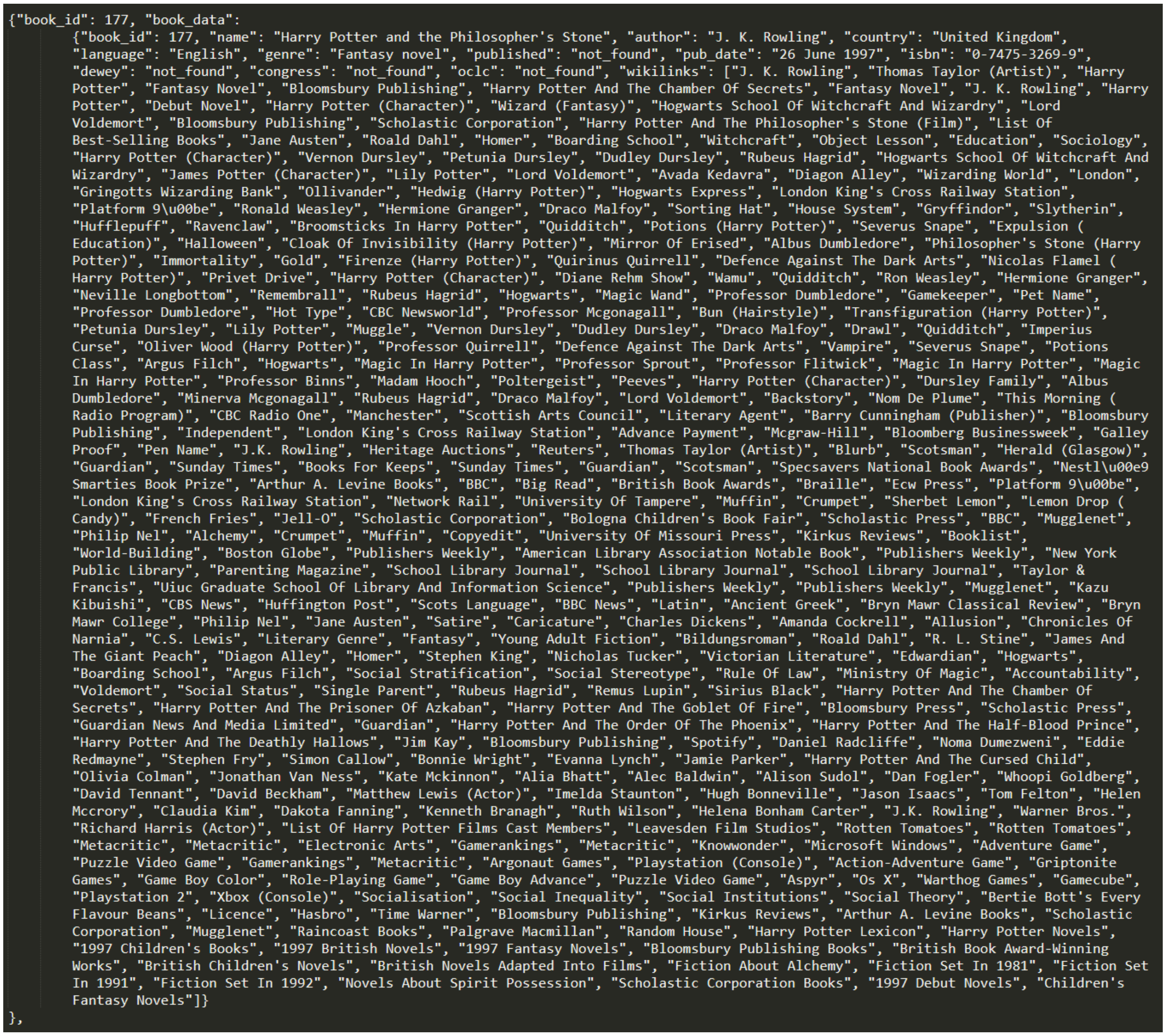
This article markup format was far too much information for what I intended to do. It's possible that I could have fed the entire article's content into a deep-learning model, but most likely that would have resulted in a model with too much noise and not enough accuracy to be of any real use. So the 40,000 articles on books were transformed to produce a much smaller data set in JSON format that contained just a few essential bibliographical details about each book plus a list of the hyperlinks that were present in each book's article page.

Figure 17 on the next page is an excerpt from the books.json file showing the record for the first *Harry Potter* book. Note the extremely densely packed information. I wanted the data files to be as lean and dense as possible with no filler so that they would load quickly from disk and be processed as quickly as possible in the CPU and the GPU.

The first three rows of the listing shown in Figure 17 comprise the book's bibliographical details. The enormous remainder of the JSON record is the list of hyperlinks that were in this particular Wikipedia article. These hyperlinks are what was fed into the deep-learning model to train it on how to measure the similarity between two books based on the similarity of their lists of hyperlinks. If you take the time to browse through the links shown, you might be surprised to see such links as "Braille," "Crumpet," "French Fries," "Social Stratification," and "Spotify." The remarkable ingenuity and variety of the hyperlinks is what makes the deep-learning model in Tim's Book Recommender such a unique and compelling book recommender compared with more-traditional methods of book recommendation such as books by the same author or books in the same genre.

Figure 17

JSON data for Harry Potter book, showing long list of links in the Wikipedia article



```
{"book_id": 177, "book_data":
        {"book_id": 177, "name": "Harry Potter and the Philosopher's Stone", "author": "J. K. Rowling", "country": "United Kingdom",
        "language": "English", "genre": "Fantasy novel", "published": "not_found", "pub_date": "26 June 1997", "isbn": "0-7475-3269-9",
        "dewey": "not_found", "congress": "not_found", "oclc": "not_found", "wikilinks": ["J. K. Rowling", "Thomas Taylor (Artist)", "Harry
        Potter", "Fantasy Novel", "Bloomsbury Publishing", "Harry Potter And The Chamber Of Secrets", "Fantasy Novel", "J. K. Rowling", "Harry
        Potter", "Debut Novel", "Harry Potter (Character)", "Wizard (Fantasy)", "Hogwarts School Of Witchcraft And Wizardry", "Lord
        Voldemort", "Bloomsbury Publishing", "Scholastic Corporation", "Harry Potter And The Philosopher's Stone (Film)", "List Of
        Best-Selling Books", "Jane Austen", "Roald Dahl", "Homer", "Boarding School", "Witchcraft", "Object Lesson", "Education", "Sociology",
        "Harry Potter (Character)", "Vernon Dursley", "Petunia Dursley", "Dudley Dursley", "Rubeus Hagrid", "Hogwarts School Of Witchcraft And
        Wizardry", "James Potter (Character)", "Lily Potter", "Lord Voldemort", "Avada Kedavra", "Diagon Alley", "Wizarding World", "London",
        "Gringotts Wizarding Bank", "Ollivander", "Hedwig (Harry Potter)", "Hogwarts Express", "London King's Cross Railway Station",
        "Platform 9\u00be", "Ronald Weasley", "Hermione Granger", "Draco Malfoy", "Sorting Hat", "House System", "Gryffindor", "Slytherin",
        "Hufflepuff", "Ravenclaw", "Broomsticks In Harry Potter", "Quidditch", "Potions (Harry Potter)", "Severus Snape", "Expulsion (
        Education)", "Halloween", "Cloak Of Invisibility (Harry Potter)", "Mirror Of Erised", "Albus Dumbledore", "Philosopher's Stone (Harry
        Potter)", "Immortality", "Gold", "Firenze (Harry Potter)", "Quirinus Quirrell", "Defence Against The Dark Arts", "Nicolas Flamel (
        Harry Potter)", "Privet Drive", "Harry Potter (Character)", "Diane Rehm Show", "Wamu", "Quidditch", "Ron Weasley", "Hermione Granger",
        "Neville Longbottom", "Remembrall", "Rubeus Hagrid", "Hogwarts", "Magic Wand", "Professor Dumbledore", "Gamekeeper", "Pet Name",
        "Professor Dumbledore", "Hot Type", "CBC Newsworld", "Professor Mcgonagall", "Bun (Hairstyle)", "Transfiguration (Harry Potter)",
        "Petunia Dursley", "Lily Potter", "Muggle", "Vernon Dursley", "Dudley Dursley", "Draco Malfoy", "Drawl", "Quidditch", "Imperius
        Curse", "Oliver Wood (Harry Potter)", "Professor Quirrell", "Defence Against The Dark Arts", "Vampire", "Severus Snape", "Potions
        Class", "Argus Filch", "Hogwarts", "Magic In Harry Potter", "Professor Sprout", "Professor Flitwick", "Magic In Harry Potter", "Magic
        In Harry Potter", "Professor Binns", "Madam Hooch", "Poltergeist", "Peeves", "Harry Potter (Character)", "Dursley Family", "Albus
        Dumbledore", "Minerva Mcgonagall", "Rubeus Hagrid", "Draco Malfoy", "Lord Voldemort", "Backstory", "Nom De Plume", "This Morning (
        Radio Program)", "CBC Radio One", "Manchester", "Scottish Arts Council", "Literary Agent", "Barry Cunningham (Publisher)", "Bloomsbury
        Publishing", "Independent", "London King's Cross Railway Station", "Advance Payment", "Mcgraw-Hill", "Bloomberg Businessweek", "Galley
        Proof", "Pen Name", "J.K. Rowling", "Heritage Auctions", "Reuters", "Thomas Taylor (Artist)", "Blurb", "Scotsman", "Herald (Glasgow)",
        "Guardian", "Sunday Times", "Books For Keeps", "Sunday Times", "Guardian", "Scotsman", "Specsavers National Book Awards", "Nestl\u00e9
        Smarties Book Prize", "Arthur A. Levine Books", "BBC", "Big Read", "British Book Awards", "Braille", "Ecw Press", "Platform 9\u00be",
        "London King's Cross Railway Station", "Network Rail", "University Of Tampere", "Muffin", "Crumpet", "Sherbet Lemon", "Lemon Drop (
        Candy)", "French Fries", "Jell-O", "Scholastic Corporation", "Bologna Children's Book Fair", "Scholastic Press", "BBC", "Mugglenet",
        "Philip Nel", "Alchemy", "Crumpet", "Muffin", "Copyedit", "University Of Missouri Press", "Kirkus Reviews", "Booklist",
        "World-Building", "Boston Globe", "Publishers Weekly", "American Library Association Notable Book", "Publishers Weekly", "New York
        Public Library", "Parenting Magazine", "School Library Journal", "School Library Journal", "School Library Journal", "Taylor &
        Francis", "Uiuc Graduate School Of Library And Information Science", "Publishers Weekly", "Publishers Weekly", "Mugglenet", "Kazu
        Kibuishi", "CBS News", "Huffington Post", "Scots Language", "BBC News", "Latin", "Ancient Greek", "Bryn Mawr Classical Review", "Bryn
        Mawr College", "Philip Nel", "Jane Austen", "Satire", "Caricature", "Charles Dickens", "Amanda Cockrell", "Allusion", "Chronicles Of
        Narnia", "C.S. Lewis", "Literary Genre", "Fantasy", "Young Adult Fiction", "Bildungsroman", "Roald Dahl", "R. L. Stine", "James And
        The Giant Peach", "Diagon Alley", "Homer", "Stephen King", "Nicholas Tucker", "Victorian Literature", "Edwardian", "Hogwarts",
        "Boarding School", "Argus Filch", "Social Stratification", "Social Stereotype", "Rule Of Law", "Ministry Of Magic", "Accountability",
        "Voldemort", "Social Status", "Single Parent", "Rubeus Hagrid", "Remus Lupin", "Sirius Black", "Harry Potter And The Chamber Of
        Secrets", "Harry Potter And The Prisoner Of Azkaban", "Harry Potter And The Goblet Of Fire", "Bloomsbury Press", "Scholastic Press",
        "Guardian News And Media Limited", "Guardian", "Harry Potter And The Order Of The Phoenix", "Harry Potter And The Half-Blood Prince",
        "Harry Potter And The Deathly Hallows", "Jim Kay", "Bloomsbury Publishing", "Spotify", "Daniel Radcliffe", "Noma Dumezweni", "Eddie
        Redmayne", "Stephen Fry", "Simon Callow", "Bonnie Wright", "Evanna Lynch", "Jamie Parker", "Harry Potter And The Cursed Child",
        "Olivia Colman", "Jonathan Van Ness", "Kate Mckinnon", "Alia Bhatt", "Alec Baldwin", "Alison Sudol", "Dan Fogler", "Whoopi Goldberg",
        "David Tennant", "David Beckham", "Matthew Lewis (Actor)", "Imelda Staunton", "Hugh Bonneville", "Jason Isaacs", "Tom Felton", "Helen
        Mccrory", "Claudia Kim", "Dakota Fanning", "Kenneth Branagh", "Ruth Wilson", "Helena Bonham Carter", "J.K. Rowling", "Warner Bros.",
        "Richard Harris (Actor)", "List Of Harry Potter Films Cast Members", "Leavesden Film Studios", "Rotten Tomatoes", "Rotten Tomatoes",
        "Metacritic", "Metacritic", "Electronic Arts", "Gamerankings", "Metacritic", "Knowwonder", "Microsoft Windows", "Adventure Game",
        "Puzzle Video Game", "Gamerankings", "Metacritic", "Argonaut Games", "Playstation (Console)", "Action-Adventure Game", "Griptonite
        Games", "Game Boy Color", "Role-Playing Game", "Game Boy Advance", "Puzzle Video Game", "Aspyr", "Os X", "Warthog Games", "Gamecube",
        "Playstation 2", "Xbox (Console)", "Socialisation", "Social Inequality", "Social Institutions", "Social Theory", "Bertie Bott's Every
        Flavour Beans", "Licence", "Hasbro", "Time Warner", "Bloomsbury Publishing", "Kirkus Reviews", "Arthur A. Levine Books", "Scholastic
        Corporation", "Mugglenet", "Raincoast Books", "Palgrave Macmillan", "Random House", "Harry Potter Lexicon", "Harry Potter Novels",
        "1997 Children's Books", "1997 British Novels", "1997 Fantasy Novels", "Bloomsbury Publishing Books", "British Book Award-Winning
        Works", "British Children's Novels", "British Novels Adapted Into Films", "Fiction About Alchemy", "Fiction Set In 1981", "Fiction Set
        In 1991", "Fiction Set In 1992", "Novels About Spirit Possession", "Scholastic Corporation Books", "1997 Debut Novels", "Children's
        Fantasy Novels"]}
},
```

Table 5 on the next page shows details about the various stages of data transformation that were used to bring the original Wikipedia dump of 86 GB down to a more manageable and useful size of 43 MB.

Table 5

Description and key details at each stage of data transformation

Data Set Description	Filename	File Format	Size (GB/MB)	Transformation Tool	Notes
Wikipedia dump file (compressed)	enwiki-20220701-pages-articles-multistream.xml.bz2	BZ2 (i.e., .tar)	20 GB	n/a	Compression ratio was 23%
Wikipedia dump file (uncompressed)	enwiki-20220701-pages-articles-multistream.xml	XML	87 GB	7-Zip	XML namespace of the dump file: http://www.mediawiki.org/xml/export-0.10/
entire Wikipedia articles about 40,000 books	[individual filenames] 0000001.txt, 0000002.txt, etc.	wikitext	2.2 GB	Python script	description of the wikitext markup language: https://en.wikipedia.org/wiki/Help:Wikitext
summary bibliographical details plus lists of hyperlinks for 40,000 books	books.json	JSON	43 MB	Python script	

Python Source Code for Data Transformation and Analysis

All source code for this project has been included in three zip archive files included with the task submission. These files are:

- capstone-wp-reader.zip
- capstone-embedder.zip
- capstone-quickstart.zip

Detailed instructions for extracting and running the Python source code in the capstone-quickstart.zip file are given in the section "Quick Start Guide (Windows)" on page 51.

Hypothesis Verification

My hypothesis is that by offering book recommendations drawn from a refreshing and innovative source, we can persuade book shoppers to take a chance on buying a newly recommended book. In the context of my ambitions to sell Tim's Book Recommender to Amazon for implementation into their checkout process, the most significant way of proving my hypothesis correct would be to compare book sales before and after the integration of my new data product into Amazon's website. If we see that sales figures increase as a direct result of customers buying the books recommended by my deep-learning model, this would validate my hypothesis. If we see no change in sales or a decrease in sales, this would prove my hypothesis to have been overly optimistic and, in short, incorrect.

Parameters have thus been defined within which we can prove or disprove the hypothesis, and although the idea of selling this model to Amazon is purely a hypothetical idea, I feel confident that the books recommended by my neural-network model would be striking and attractive enough to prove the hypothesis true should it ever occur that this model was included in Amazon's checkout pipeline.

Visualizations

The three data visualizations that were selected for inclusion in the dashboard GUI offered additional insights to the user about the relationship between their originally selected book and either (a) other similar books, or (b) other books by the same author. The visualizations provided proof of the extensive data processing capabilities of the deep-learning model in that it was able to compute the relative similarities of 40,000 books and convert this web of relationships into a 2-dimensional representation.

The genre map in particular not only provided a context for interpreting the dot graphs of similar books and same-author books, but also showed users the relative similarity of the top 10 most common literary genres represented in the model's book repertoire.

Assessment of Product's Accuracy

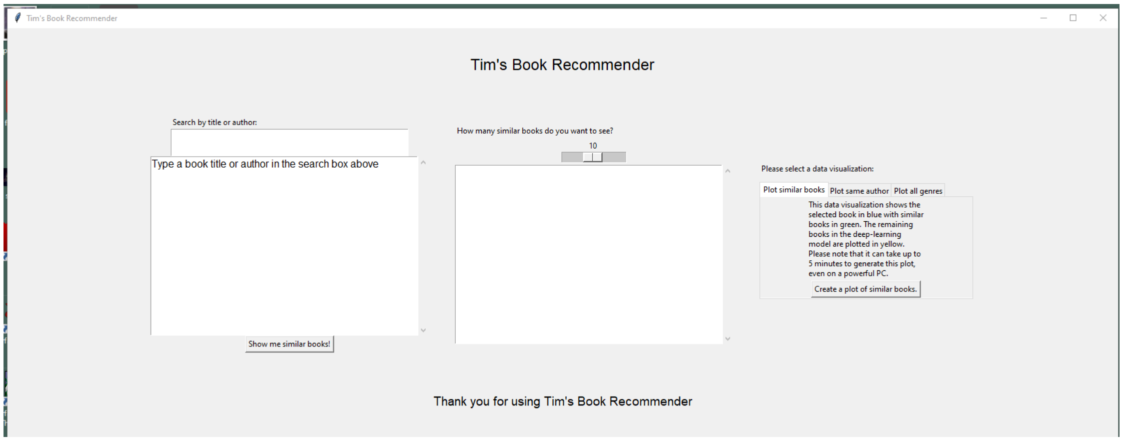
The accuracy of my data product was assessed in several ways. First, I entered the titles of books I was intimately familiar with, and I was able to vouch that the model's output consisted of books that were not only similar but also striking.

Second, I examined the genre map showing where books with the same literary genre were plotted in the same color, and I identified an area where science fictions were predominant. Then I searched for a science fiction book I was familiar with and plotted this new book on the map. I saw that the science fiction book was plotted in the same area that the genre map said was populated by science fiction books. This informal test served to validate the accuracy of the dimensional reduction of the neural-network model from 50 dimensions down to an easily graphed 2 dimensions (i.e., *x* and *y* coordinates).

User Guide

Thank you for choosing Tim’s Book Recommender! Here are a few suggestions on how to make the most of your using this innovative new tool for finding new books to read. First, be sure that Tim’s Book Recommender is installed using the instructions in the “Quick Start Guide” section below.

When Tim’s Book Recommender GUI program is started, you will see this friendly dashboard:


Tim's Book Recommender
Tim's Book Recommender
Search by title or author:
How many similar books do you want to see?
10
Please select a data visualization:
Plot similar books
Plot same author
Plot all genres
Show me similar books!
Create a plot of similar books.

Type a book title or author in the search box above

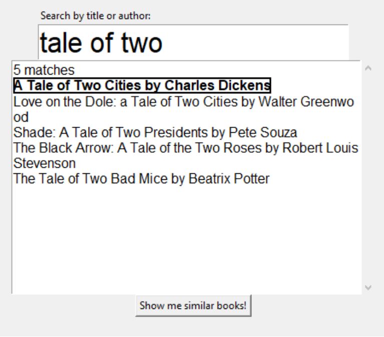
This data visualization shows the selected book in blue with similar books in green. The remaining books in the deep-learning model are plotted in yellow. Please note that it can take up to 5 minutes to generate this plot, even on a powerful PC.

Thank you for using Tim's Book Recommender

The left section lets you choose a book. The middle section shows you books similar to your book. The right section

lets you create data visualizations of information about the book you chose.

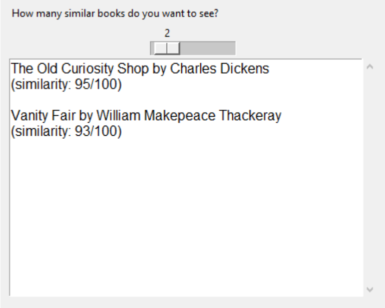
Start typing a title or author’s name in the “Search by title or author” field to see matches. Click on a match with your mouse. Then click the “Show me similar books!” button.



tale of two

5 matches
A Tale of Two Cities by Charles Dickens
Love on the Dole: a Tale of Two Cities by Walter Greenwood
Shade: A Tale of Two Presidents by Pete Souza
The Black Arrow: A Tale of the Two Roses by Robert Louis Stevenson
The Tale of Two Bad Mice by Beatrix Potter

A list of books similar to yours will appear. The default number is 10 similar books, but using the slider interface you can choose from 1 to 20 books. The example shows 2 books.



The Old Curiosity Shop by Charles Dickens
(similarity: 95/100)

Vanity Fair by William Makepeace Thackeray
(similarity: 93/100)

Finally, in the right section, choose from three data visualizations. Click the tabs for “Plot similar books,” “Plot same author,” and “Plot all genres” and click the button on each tabbed page to create a plot. Please be aware that it can take up to 10 minutes or more to generate a plot, depending on your computer’s hardware capabilities. Happy book exploring!

Quick Start Guide (Windows)

System requirements:

- Microsoft Windows 10 or higher
- Python 3.7 or higher already installed (see https://www.python.org/downloads/ if necessary)
- Windows PowerShell already installed

Instructions:

1. Open a PowerShell shell
    a. From Windows, press Control-Escape on the keyboard to open the Start menu.
    b. Start typing “powershell” on the keyboard.
    c. When the Windows PowerShell app appears in the menu, click on the icon or “Open.”
    d. In the PowerShell shell, create a new directory where you wish and enter it.
    ```
    mkdir capstone-quickstart
    cd capstone-quickstart
    ```
    e. In Windows, unzip the contents of capstone-quickstart.zip into this new directory.
2. In Powershell, create a new virtual environment.
    ```
    python -m venv .venv
    ```
3. And activate the virtual environment.
    ```
    ./.venv/Scripts/Activate.ps1
    ```
4. Install the requirements using pip.
    ```
    pip install -r requirements.txt
    ```
5. Then start the GUI program from the command line.
    ```
    python main.py
    ```
6. Now follow the User Guide above for helpful hints about using the program.
7. To exit the GUI program, just click the X in the upper right corner, as with any program.
8. Finally, exit PowerShell either by typing “exit” or clicking the X in the upper right corner.
    ```
    exit
    ```

In-Depth Installation Guide (Windows)

It is **strongly** recommended to use the "Quick Start Guide" to start the GUI app and explore the capabilities of the machine-learning model that way. However, the instructions below walk you through the steps to completely reproduce this project, starting from the very beginning with the Wikipedia dump and progressing through all stages of machine learning up through the finished GUI app. (Note that training a deep-learning network without a GPU can result in extremely long training times.)

(All zip archive files referenced below were included in my task submission.)

1. Download a recent Wikipedia dump of the English language Wikipedia along with its associated index file. For example:
    a. Dump: https://dumps.wikimedia.org/enwiki/20220720/enwiki-20220720-pages-articles-multistream.xml.bz2 (19.6 GB)
    b. Index file: https://dumps.wikimedia.org/enwiki/20220720/enwiki-20220720-pages-articles-multistream-index.txt.bz2 (232.2 MB)
2. First, install my Python program to process the Wikipedia dump and extract the articles.
    a. Create a new directory
    b. Unzip the contents of the `capstone-wp-reader.zip` archive into the new directory
    c. Using PowerShell, go to that directory, create and activate a new virtual environment, and run `pip -r requirements.txt`
    d. Update these Python source code files so the directory paths and filenames reflect your system setup, e.g., paths for the Wikipedia dump and index, and the target directory for books.json, wikilinks.json, and firehose_fields.txt files:
        i. config.py
        ii. main.py
        iii. bz2_to_parquet.py
        iv. parquet_to_json.py
    e. From PowerShell, with the virtual environment active, run `python main.py`
3. Next install my Python program to train a deep-learning model to create embeddings for the wikilinks that will be used to measure book similarity.
    a. Create a new directory
    b. Unzip the contents of the `capstone-embedder.zip` archive into the new directory
    c. Using PowerShell, go to that directory, create and activate a new virtual environment, and run `pip -r requirements.txt`
    d. Update the appropriate directory paths and filenames in these files, as before:
        i. config.py
        ii. my_data.py
    e. From PowerShell, with the virtual environment active, run `python main.py`
4. Next install the GUI using the "Quick Start Guide" above, but before you start the GUI first copy these files you created in steps 2 and 3 above into that directory:
    a. books.json (from step 2)
    b. model3.h5 (from step 3)
5. Finally, start the GUI per the "Quick Start Guide." Congratulations! You reproduced the project.

References

Chollet, F. (2021). *Deep learning with Python* (2nd ed.). Manning.

Gallo, A. (2014, October 29). *The value of keeping the right customers.* Harvard Business Review. https://hbr.org/2014/10/the-value-of-keeping-the-right-customers

IBM. (2021, March 8). *Evaluation overview.* https://www.ibm.com/docs/en/spss-modeler/18.2.0?topic=evaluation-overview

Jensen, K. (2012). *Process diagram for CRISP-DM*. Reprinted in accordance with CC BY-SA 3.0.

Kinsta. (2022, June 8). *Cloud market share: A look at the cloud ecosystem in 2022.* https://kinsta.com/blog/cloud-market-share/

Russell, S., & Norvig, P. (2021). *Artificial intelligence: A modern approach.* (4th ed.). Hoboken, NJ: Pearson.

Statista. (2021, October 29). *Market share of leading retail e-commerce companies in the United States as of October 2021.* https://www.statista.com/statistics/274255/market-share-of-the-leading-retailers-in-us-e-commerce/

Wikipedia. (2022). *"Agile software development."* https://en.wikipedia.org/wiki/Agile_software_development

Wikipedia. (2022). *"Garbage in, garbage out."* https://en.wikipedia.org/wiki/Garbage_in,_garbage_out

Wikipedia. (2022). *"JSON."* https://en.wikipedia.org/wiki/JSON

Wikipedia. (2022). *"Rapid application development."* https://en.wikipedia.org/wiki/Rapid_application_development

Wikipedia. (2022). *"Regular expression."* https://en.wikipedia.org/wiki/Regular_expression

Wikipedia. (2022). *"Time to market."* https://en.wikipedia.org/wiki/Time_to_market

Wikipedia. (2022). *"Waterfall model."* https://en.wikipedia.org/wiki/Waterfall_model

Wiktionary. (2022). *"Conversion."* https://en.wiktionary.org/wiki/conversion