2020

Hazelnut Games

--Provide casual players The best game-recommendation experience.

IRS PROJECT REPORT

ZHANG ZEKUN (A0215485B)

SHI ZHAOHENG(A0215413U)

TAO XIYAN (A0215472J)

NATIONAL UNIVERSITY OF SINGAPORE | INSTITUTE OF SYSTEM SCIENCE

HTTPS://GITHUB.COM/2020-IRS-G12/GameRecommender

Contents

Executive Summary

Project's Background

Over the last decade, digital platforms of media emerged as the internet became faster and more accessible to more consumers all over the world. The spread of IT use has spread due to ease of communication throughout the years because it helped connect buyers and sellers together on one system. There are platforms for ridesharing, video streaming/sharing, house rentals, and e-commerce. The video game industry has also been proven to be very profitable because of the consumer buying choices and engagement in the products. In the gaming industry, there are a few key leaders that dominate the mobile and console gaming platforms like Nintendo, Microsoft's Xbox and Apple. Some of the leaders in mobile and console gaming platforms have tried to get into the PC platform but they have not done it as successfully Valve's Steam (Staff, 2012). Microsoft and Apple have both created a store for their desktop devices, but they primarily help consumers with productivity.

Industry Analysis

Those games platforms are definitely with high business value. for example, before there were platforms like Steam for a desktop and laptop game marketplace. People used to have CD games or productivity applications that could be run on a computer. Some people still have those kinds of CDs but as computers have been developed, the CD drive has slowly phased out. For example, before the slow decline of CDs, Valve company created a platform for desktop and laptop computers. They provide users of different operating systems to get access to the games that developers have created for those operating systems. There are other competitors that created their own platforms for digital games too like Blizzard Entertainment, Ubisoft, and EA. A lot of popular game development companies put games into Steam to earn a profit. They do it through the Steam Direct, which is their partnership network. Steam also makes it possible for indie developers to earn a profit from the games they create. The independent developers also go through the same system, Steam Direct. Steam takes 30% of game profits from indie developers and game companies (Jessica Lam, 2018[1]). There isn't public financial data for Steam's or Valve's profits but it is estimated that they have $4.3 billion dollars in profits just from game sales alone in 2017 ("Steam Earned an Estimated $4.3B in 2017, Not Counting In-Apps and DLC — The Esports Observer," 2018). Many sites claim that Steam is extremely profitable, they control 70% of the PC download market ("Steam controls up to 70% of the PC download market and is," 2011).[2]

Industry Outlook

It's inevitable that there might be more digital game platforms that may pop up in the future that will have innovative ways to draw people in to buy games but there are some different ways to make Steam different from other companies.

As of December 6th, 2018, Epic Games, creator of Fortnite, launched a game marketplace platform that rivals Steam. It is supposed to incentivize indie game makers and publishers to sell their games on Epic Game’s Store because indie game makers and publishers could earn 88% of the profits instead of only 70%, which undercuts the other platforms by 18% (Gilbert, 2018). The outlook of Valve Corporation looks like it would continue to grow as technology is used more widely and if they continue to differentiate themselves from other competitors. As they continue to collect data on their users and use the data, they have collected over the twenty years they have been around; they could contribute a lot to artificial intelligence, machine learning, and virtual reality. Valve could create better games or productivity applications with that kind of information. If they decide to become a public company in the future, it would be a company to invest in.[2]

From this point of view, the future of the gaming platform is still full of unlimited space and business opportunities.

Problem Description

Problem Statement

The major problem we want to solve is to help light game players to find video games interesting for them.

Project Objective

To implement a web-based system with concise UI which can recommend games to non-core players with the help of AI and try to make profits through it.

Business Value

Industry strategy

The profit model of the game platform is to find a good game and promote it to players. After the player buys or downloads it, profit is divided with the game developer.

The role of the game platform is to connect developers and players. It has 2 tasks:

1. To help game developers find players;
2. To help players find good games.

Comparing the two points above, the most important point is the first point: to help developers find players. Without a large number of users, the above two points cannot be achieved. Moreover, the number of users of the game platform basically depends on whether we can help me find a game that suits me on the platform.

In short, there are a large number of users, and high-quality users is the most important thing for a gaming platform, as the so-called "user is king". Therefore, a review and recommendation system for games is one of the most important aspects of the game platform.

System user portrait & direction

Our system is a game recommendation system for light gamers. We have three functions: search, display, and smart recommendation. The difference with other existing game platforms on the market is that we are very friendly to casual gamers, do not require cumbersome registration steps, and there are no complicated and unintelligible game terms. Just based on your direct understanding of the game, we can help you find the game that suits you.

For example, Steam has a good model for its review and recommendation system for games that are already in the game marketplace. Prior to their current system, they used to have Steam Greenlight. This is a feature that allowed indie developers to pay a fee for feedback from the gaming community for potential games to be sold on Steam. The indie developers would get crowdsourced feedback on their proposed game (“Steam Blog :: Evolving Steam,” 2017). Many of the games that gained a lot of positive feedback were not very reliable because they were getting fake votes (Crider, 2017). This feature would have potentially done very well if there was a better vetting process and if they approved games to be sold in the Steam marketplace faster. According to Steam’s blog on Evolving Greenlight, they expressed that the feature was supposed to let independent developers distribute games in a more streamlined way. They discovered that they needed a new product to “improve the process of bringing more content to Steam and to connect customers to the products they wanted” (“Steam Blog:: Evolving Steam,” 2017). This led them to create Steam Direct, which is also known as the Steam Partner Network. This system has shown to be very profitable as mentioned earlier in the report.[1]

Another trend is the movement away from PC and console gaming toward mobile gaming – while mobile has exploded, PC and console growth has slowed to the low single digits.

Though winning in mobile would significantly accelerate growth, some traditional gaming platforms are still historically a console and PC business. As a result, in order to continue growing as it has in the past, they will have to succeed in a new business that requires a very different set of creative and commercial capabilities.

Whereas PC / console success depends on blockbuster games that consumers play for many hours on end, mobile success relies more on casual games that users repeatedly come back to for short amounts of time. In other words, the type of game that flourishes on PC and console, where traditional game platform has strong competencies, is very different from the type of game that works well on mobile, where traditional game platform has gaps.

In the future, our platform will not only be committed to providing lighter game users with easier and more customized game recommendation and evaluation systems, but will also be committed to expanding mobile game business in the future.

Overall, our system is a game recommendation system for casual gamers. We have three main functions: searching, display, and smart recommendation. The difference between other existing game platforms on the market is that we are very friendly to casual gamers. Our system does not require cumbersome registration steps, and there are no complicated and difficult to understand game terms. Just based on your direct understanding of the game, we can help you find a game which suits you. Our system is very suitable for casual gamers.

Solution Outline

Knowledge Modeling

Knowledge modeling is the footstone of our intelligent system. During this process, we need to figure out some valuable knowledge to recommend games which are interesting for certain user.

Firstly, we need to identify the knowledge, which includes source of information, insights from information source and knowledge acquisition technique. Knowledge identification sets the groundwork for the next stage of this project.

S/N	**Source of information**	**Insights from information source**	**Knowledge acquisition technique**
1	Game review websites (e.g. Metacritic)	These websites can provide comprehensive information about a video game (e.g. title, description, platform etc.)	Crawler
2	Video game experts	Experienced video game players or developers' advice can be great guidance for questionnaire design and recommender algorithm selection	Experts' advice
3	Users	Utilize cookies and questionnaire integrated in the system to find out users' preference on games	Questionnaire in system, Cookies

Table 1 Knowledge Identification

On the basis of the identification of knowledge sources, knowledge acquisition and necessary techniques allowing us to capture the problem-solving domain knowledge. A dependency (inference) diagram can be derived.

With this inference diagram, we commence our follow-up design and implement works.

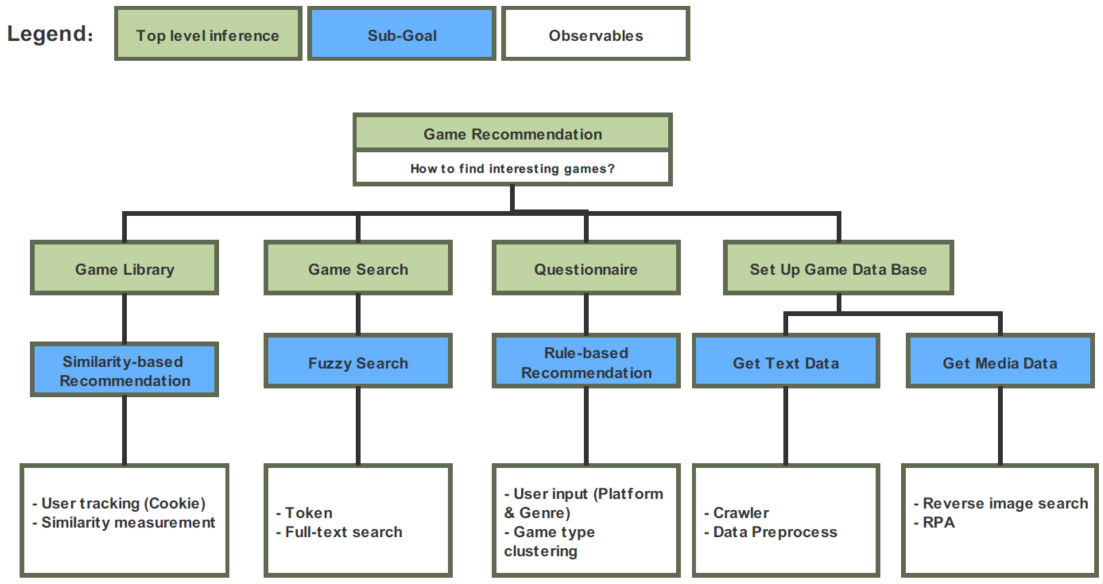


Figure 1 Dependency (inference) Diagram

Data Acquisition & Preprocess

To build a video game recommender system, we need to set up a video game database first. Because such databases or datasets which include the latest games' information always contain business value, it is hard to get ready to use datasets for our system. Therefore, we utilize web crawler to collect data from a video game review website – Metacritic, which aggregates many game information we need (Its robot.txt also allows us to scrape data from it).

After scraping necessary data and preprocess it properly, we set up a video game database. However, games' cover images on Metacritic are in very low resolution. To solve this, we use RPA to automatically find high resolution versions of these images via Bing's reverse image search.

Data Acquisition

Crawler

Scrapy is a fast-high-level web crawling and web scraping framework, used to crawl websites and extract structured data from their pages. It based on python, and we build our web crawler based on Scrapy. This image demonstrates Scrapy's working process.

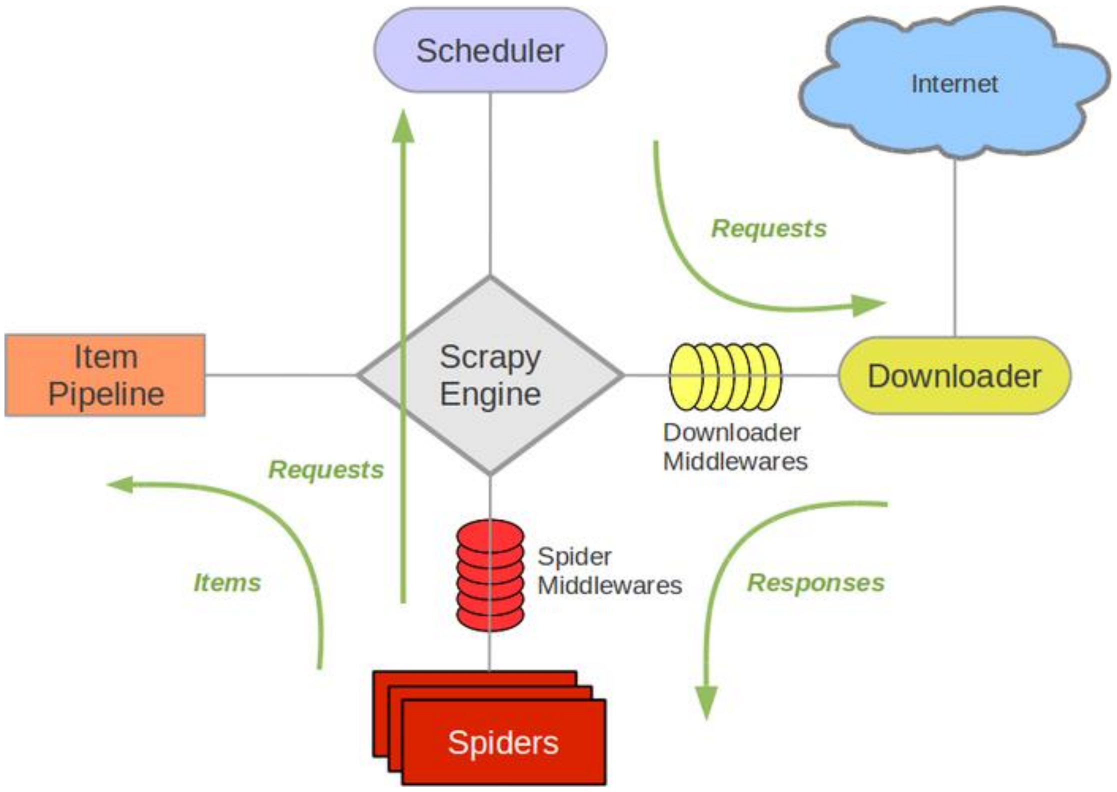


Figure 2 Scrapy's Working Process

Scrapy uses xpath to find elements on web pages, which gives us lots of flexibility. For different games' webpages that may have subtle differences in HTML structure, such flexibility provides us with the ability to handle differences via identical source code.

The fetched data stores in CSV tables. They will be preprocessed, and be saved in the system's DB.

Web Automation

On Metacritic, our crawler can collect enough text data for the system. But for media files, although Scrapy can download images from web pages during crawling process, the image files we download from Metacritic are in very low resolution.

To gain high resolution images and create a better user experience. We use an RPA tool – TagUI to handle Bing's reverse image search to get high quality images with these low-quality images we have. Even though sometimes it may find unsuitable results and need a human administrator to review its outcomes, is way more efficient than do this totally manually.

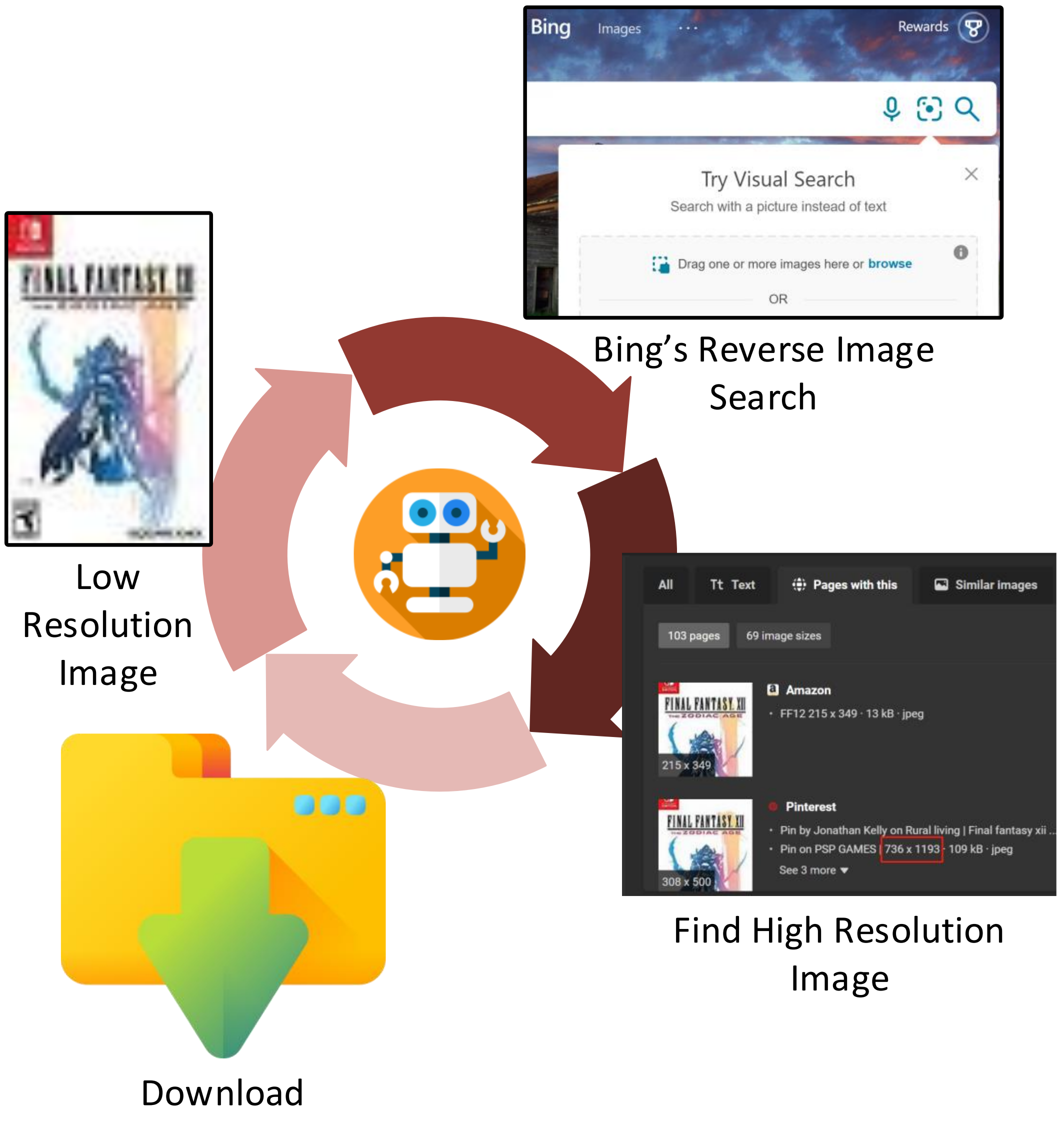

Bing
Images
Rewards
Try Visual Search
Search with a picture instead of text
Drag one or more images here or browse
OR
Bing's Reverse Image Search
Low Resolution Image
All
Text
Pages with this
Similar images
103 pages
69 image sizes
Amazon
FF12 215 x 349 · 13 kB · jpeg
215 x 349
Pinterest
Pin by Jonathan Kelly on Rural living | Final fantasy xii ...
Pin on PSP GAMES 736 x 1193 · 109 kB · jpeg
See 3 more
308 x 500
Find High Resolution Image
Download

Figure 3 RPA Agent's Working Process

Data Preprocess

After enough raw data is collected by crawler, we need to properly preprocess it, so that our recommender algorithms can have better performance. Concretely, we have these steps:

1) Genre Reduction: In raw data, there are over 100 different genres, but we only have 800 games in our database, so we need less and more generalized genres to help users to filter and find games they want to play. Then we get some expert advice and create a python problem to merge similar genres. We successfully reduce the amount of genre from over 100 to around 30.

2) On-hot genre table: In raw data, every game has a column to save its genres in a string that concatenate all its genres and delimiters. It is inconvenient for our algorithms to access these genres of each game. So, a preprocess python program is written to transform this into a one-hot table. Every game occupies one row and every genre

occupies one column, then we use 1 or 0 to represent if certain game belongs to certain genre.

We utilize a widely used python library – pandas to finish the data's preprocessing.

System Architecture

Overview

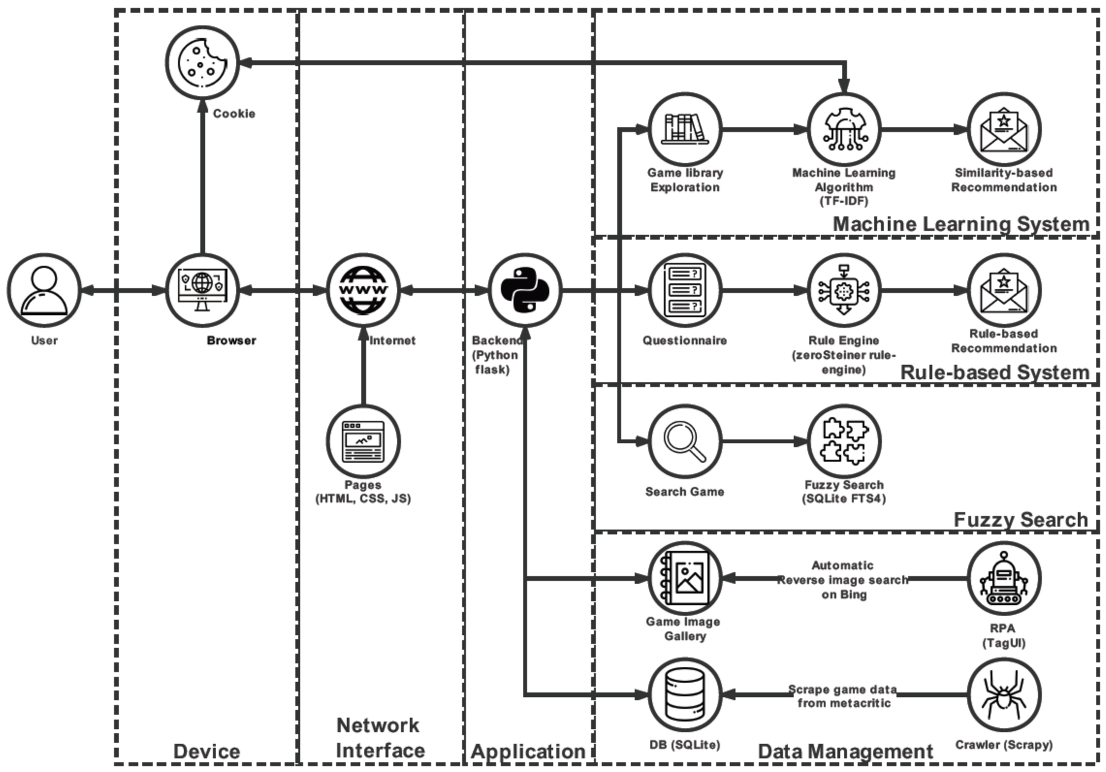


Figure 4 Whole System's Architecture

This system is web-based, which makes our potential users can access our system via both PCs and consoles (Xbox, PlayStation and Switch all have browsers). *Figure4* shows the system's architecture:

Backend

Since python is the most wide-used programming language to implement AI algorithms, which are the core of our recommender system, we decide to build the whole intelligent system's backend with python to reduce extra costs on system integration among different parts written in different languages. Because it's the first version, we try to keep it as light-weight as possible while keeping it useable. It is on this premise we choose SQLite as its DB and AWS EC2 as its deployment platform.

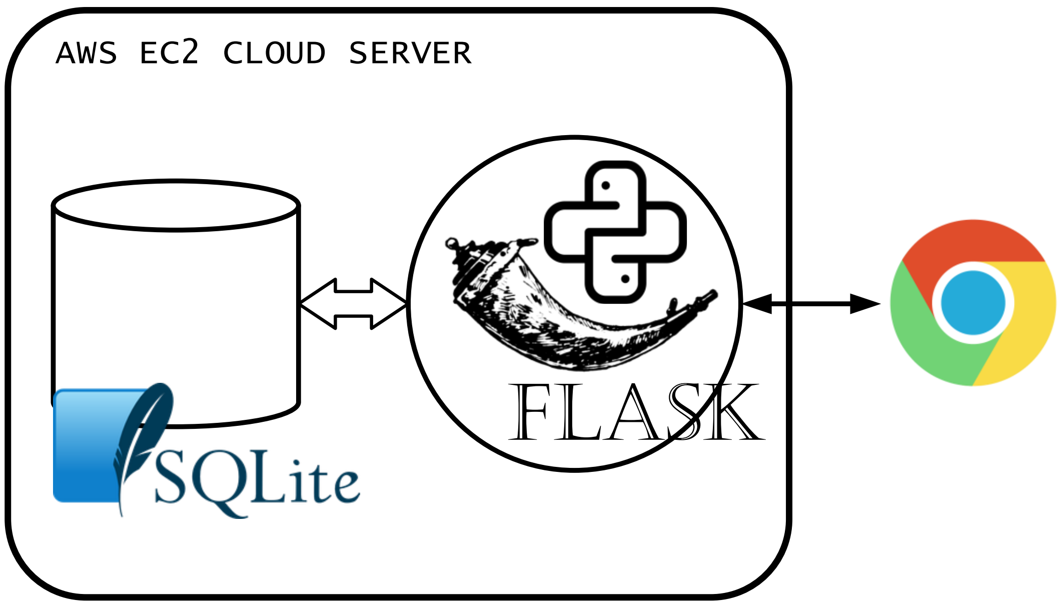


Figure 5 Backend Deployment

Backend Framework

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. However, Flask supports extensions that can add application features, which gives it the ability to scale up to complex applications.

For our game recommender system, such features of Flask let us be able to set up the system quickly as well as keep its potential to extend to a large application. Flask also contains a template engine named Jinja2, which is also written in python and integrated by default. Web template engine lets web designers and developers work with web templates to automatically generate custom web pages. In this case, we can generate a recommended game data list and use Jinja2 with our web page template to generate a page with these games displayed on it.

DB

SQLite is a C-language library that implements a small, fast, self-contained, high-reliability, full-featured, SQL database engine. SQLite file format is stable, cross-platform, and backward compatible.

We choose SQLite for it doesn't need complex configuration before using it, and it's also an easy task to replace SQLite with a more powerful DB software when it is necessary.

Cloud deployment

The system is deployed on could service.

Amazon Elastic Compute Cloud (Amazon EC2) is a cloud-based web service that lets us easily deploy our system online. With such cloud service, we can test, demonstrate and improve the system in a natural way, we no longer have to set up localhost for everyone involved in the project.

Frontend

Since this is a web-based application, the frontend consists of several HTML5 pages. We develop these pages with HTML5, JavaScript, jQuery and Ajax.

Pages

Page Name	Description
index.html	The front page. It will recommend some games according to users' browsing history saved in cookies.
gameLst.html	This page is used to show search results, which includes many asynchronous requests written in JavaScript & Ajax to implement search filters.
gameDetail.html	Every single game's detail is showed onthis page, it also recommends some games similar to the detailed one.
questionnaire.html	A questionnaire page, it will ask users some question and give a customized game recommendation.
recGameLst.html	After users fill out questionnaires, the customized game recommendation will be aggregated onthis page.

Table 2 Web Pages

Game Recommender

The core function of our system is to recommend games to our users, to provide a better user experience (Yan Guo, 2017 [3]), we integrate multiple recommendation methods in one system.

This section will introduce our recommend algorithms and implementations in detail. The workflow of the recommender can be described in following chart:

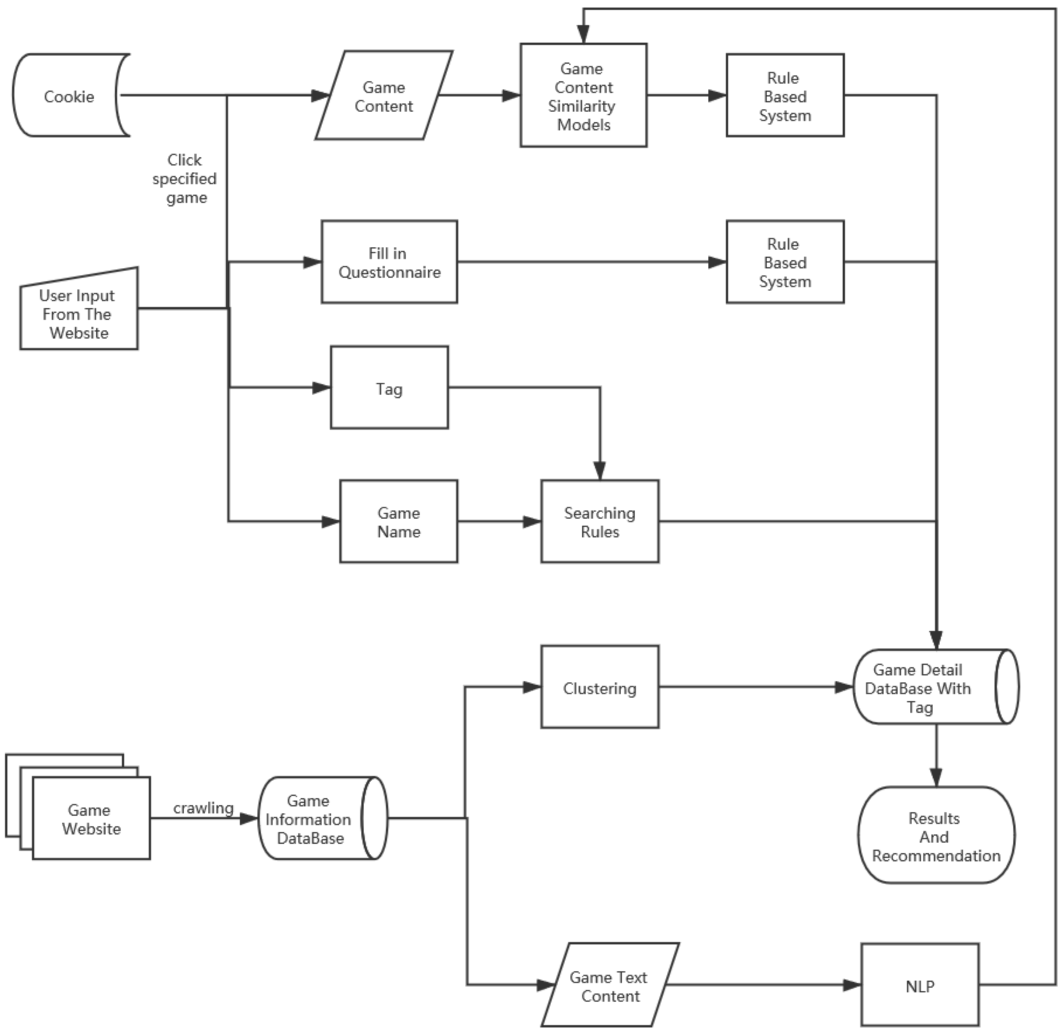


Figure 6 Workflow of Recommender

Algorithm Introduction

K-means

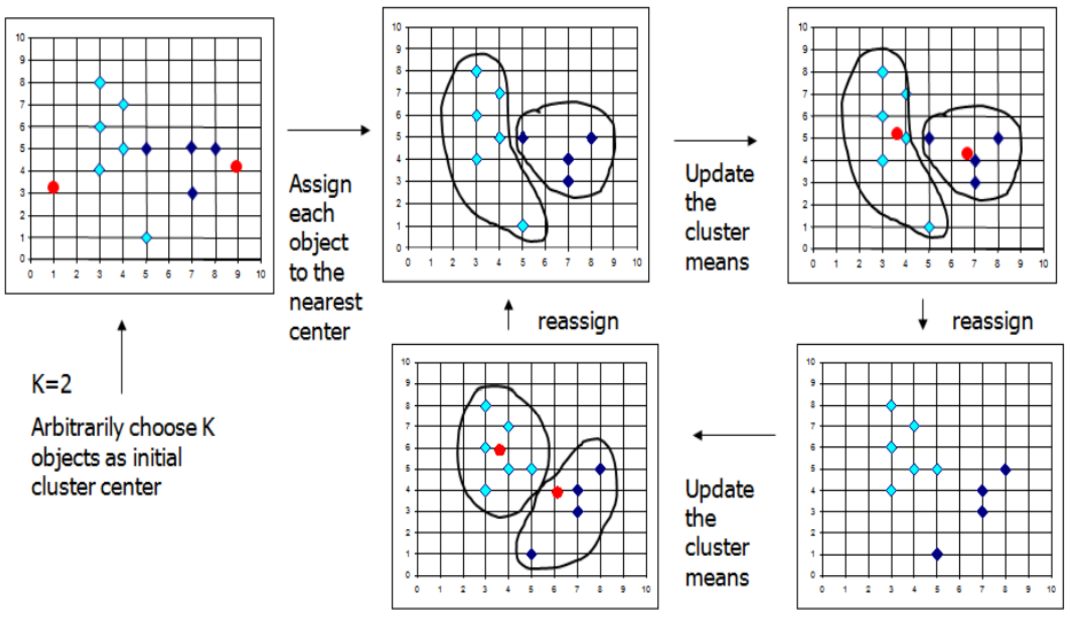
K-means clustering is one of the simplest and popular unsupervised machine learning algorithms. Typically, unsupervised algorithms make inferences from datasets using only input vectors without referring to known, or labeled, outcomes.

You will define a target number k, which refers to the number of centroids you need in the dataset. A centroid is the imaginary or real location representing the center of the cluster. Every data point is allocated to each of the clusters through reducing the in-cluster sum of squares. In other words, the K-means algorithm identifies k number of centroids, and then allocates every data point to the nearest cluster, while keeping the centroids as small as possible. The 'means' in the K-means refers to averaging of the data; that is, finding the centroid. (Khaled Alsabti, 1997[7])

To process the learning data, the K-means algorithm in data mining starts with the first group of randomly selected centroids, which are used as the beginning points for every cluster, and then performs iterative (repetitive) calculations to optimize the positions of the centroids. (P.S.Bradley, 2000 [6])

It halts creating and optimizing clusters when either:

- The centroids have stabilized — there is no change in their values because the clustering has been successful.
- The defined number of iterations has been achieved.



Assign each object to the nearest center
Update the cluster means
reassign
reassign
Update the cluster means
K=2
Arbitrarily choose K objects as initial cluster center

Figure 7 K-means

TfidfVectorizer

Typically, the TF-IDF weight is composed by two terms: the first computes the normalized Term Frequency (TF), aka. the number of times a word appears in a document, divided by the total number of words in that document; the second term is the Inverse Document Frequency (IDF), computed as the logarithm of the number of the documents in the corpus divided by the number of documents where the specific term appears. (Li-Ping Jing, 2020 [4]; Zhang Yun-tao, 2005 [5])

- TF: Term Frequency, which measures how frequently a term occurs in a document. Since every document is different in length, it is possible that a term would appear much more times in long documents than shorter ones. Thus, the term frequency is often divided by the document length (aka. the total number of terms in the document) as a way of normalization:
- TF(t) = (Number of times term t appears in a document) / (Total number of terms in the document).
- IDF: Inverse Document Frequency, which measures how important a term is. While

computing TF, all terms are considered equally important. However, it is known that certain terms, such as "is", "of", and "that", may appear a lot of times but have little importance. Thus we need to weigh down the frequent terms while scaling up the rare ones, by computing the following:

- IDF(t) = log_e(Total number of documents / Number of documents with term t in it).

CountVectorizer

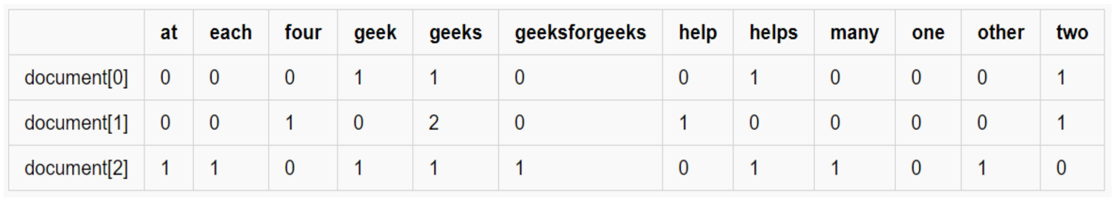
CountVectorizer is a great tool provided by the scikit-learn library in Python. It is used to transform a given text into a vector on the basis of the frequency (count) of each word that occurs in the entire text. This is helpful when we have multiple such texts, and we wish to convert each word in each text into vectors (for using in further text analysis).

Let us consider a few sample texts from a document (each as a list element):

document = [“One Geek helps Two Geeks”, “Two Geeks help Four Geeks”, “Each Geek helps many other Geeks at GeeksforGeeks.”]

CountVectorizer creates a matrix in which each unique word is represented by a column of the matrix, and each text sample from the document is a row in the matrix. The value of each cell is nothing but the count of the word in that particular text sample.

This can be visualized as follows:



	at	each	four	geek	geeks	geeksforgeeks	help	helps	many	one	other	two
document[0]	0	0	0	1	1	0	0	1	0	0	0	1
document[1]	0	0	1	0	2	0	1	0	0	0	0	1
document[2]	1	1	0	1	1	1	0	1	1	0	1	0

Table 3 CountVectorizer Example

Rule-based Recommender

Initial recommendation

As we know, we do not have any information about user when he/she first uses our system, which will cause a problem in how we can recommend some proper games for a new user. To solve this problem, we came up with an idea that we just recommend four hottest games in our dataset to the new users. But another problem appears, how can a game be thought of as a hot game, which needs us to build some rules to deal with it.

In our dataset, every game has its score that is given by players. It seems that the scores can be the standard to judge whether a game is hot or not, but, in fact, it is unreliable. For example, imagine you want to order Chinese food, you have a couple of options, one restaurant has a 5-star given by only 5 people while the other restaurant has 4.5-star given by 1000 people. Which restaurant would you prefer? Of course, the second one will be a better choice. So, the number of players of this game will be taken into consideration as an important factor for hot game evaluation. However, only the company that creates the game knows the accurate number of players of this game, so we use the comments about games as substitute.

Finally, we thought out a formula to solve the problem:

comments / mean comments * score / mean score

The mean comments and mean scores are calculated from the whole dataset.

Questionnaire design & cluster

Obviously, the initial recommendation is unsatisfying for users, so we must get more information to make the recommendation more satisfy the user. To complete this purpose, the system will let users customize their own game recommendation set---personalized recommendation.

For the personalized recommendation, we decide to design a simple questionnaire that only has two questions--- the genre and platform of games you like. Our system is designed for new players, most of them hardly know anything details about games, which is the reason for us to create a simple questionnaire.

A player will just focus on games that can play on the machine they have and most players only have one gaming platform (Kevin, 2019[8]). For example, a person who only has PC will not be interested in a PS4 game. So, platform information is critical for us to recommend games. The genre of game is also important information. Because, as far as we know, almost all players tend to play one or two kinds of games, it is weird to find someone loves playing many types of games (larger than 3), which makes game genre information significant.

However, there are too many types of games in our collected dataset even after proper preprocess (around 30 genres), which makes it impossible to display all the genres for users to choose. So, the solution we thought out is to apply clustering method k-means to assign those genres into 8 classes.

First, we use “CountVectorizer”, which is used to convert a collection of text documents to a vector of term/token counts and also enables the pre-processing of text data prior to generating the vector representation, to extract features from the genres of game in dataset. And then, we apply k-means on these features and get the results. Next, we analyze 8 groups to find some similarity of genres that are in the same group by using our domain knowledge. Finally, we have another 8 new genres for games in dataset, including Puzzle Adventure, Platformer, Sports (Competitive Games), Role-playing, Open-world Adventure, Shooter, Strategy and Linear Story Adventure.

According to these groups, we find 8 typical games, which are played by many people and have a good reputation, for these 8 classes. And the system shows videos of these games in the questionnaire for users to choose.

Rule engine

The questionnaire we design is dramatically flexible. Users can make multiple choices for each question and can choose nothing. Because different decisions made by users will have different searching strategies, we decide to use rule engine to help system to search games that satisfy the requirements of user.

We apply rule-engine, a lightweight, optionally typed expression language with a custom grammar for matching arbitrary Python objects, to build the rule-based system. Rule Engine expressions are written in their own language, defined as strings in Python. Some features of this language include optional type hinting, matching strings with regular expressions, datetime datatypes, data attributes.

With the help of rule-engine, we build twelve rules to search the proper games for users, which is the final part of the personalized recommendation.

Similarity-based Recommender

Similarity measurement

It is obvious that people always tend to play games that have some similarities with each other. For example, a person who is addicted to KOF (a fighting game) will have a great possibility to want to try Street Fighter (another fighting game that is similar with KOF). And, of course, we want our system to have this function, which we will use TfidfVectorizer and CountVectorizer to achieve that.

The most important step is to extract some features from some text information of games and build TfidfVectorizer and CountVectorizer matrices.

- **TfidfVectorizer matrix** The features we want to extract from description of the game is some unique words and if there are two games that have a lot of same unique words, we can simply consider this two games are similar with each other. And the TF-IDF score is the frequency of a word occurring in a document, down-weighted by the number of documents in which it occurs. This is done to reduce the importance of words that frequently occur in plot overviews and, therefore, their significance in computing the final similarity score. So, we use TfidfVectorizer to deal with the descriptions of games in the dataset to generate a huge TfidfVectorizer matrix.

- **CountVectorizer matrix** We will combine platform and genre of game together as a text value to apply CountVectorizer to get feature matrix. Why do we apply TfidfVectorizer again? Because what we want to do here is not to extract unique words, we just want to know the similarity of those text values directly. This algorithm will calculate frequency of every word that you provided and present those frequencies as a matrix.

When users click a link of game, the information of this game will be sent to the backend. Then, we will find the TfidfVectorizer of this game in the matrix and calculate cosine similarity between the selected game and the rest of games. Next, we will select the top

60 games that get high similarity scores. Among those 60 games, we will choose another 20 games that have higher scores (we introduce in **Initial recommendation**) than the rest of them. The same procedure will be applied by using CountVectorizer matrix. As a result, we get two sets of 20 games by using those two matrices. A game thatappears in both sets will be the first choice for the final recommendation and the second choice will be the one that has greater scores. Finally, we will get 8 games and display them on the website.

Using cookie to track users

An HTTP cookie (web cookie, browser cookie) is a small piece of data that a server sends to the user's web browser. The browser may store it and send it back with later requests to the same server. Typically, cookies are mainly used for three purposes: Session management, personalization and tracking.

In our case, we use cookies to track users' behaviors in browsers and then transmit the data to the backend. The data containing what games one user has browsed can help the system to reveal his game preference and give customized recommendations, which contain games measured (use methods mentioned above) to be similar with browsed ones.

Game Search

There are hundreds of games in the system's game library, that's why users need a smart search function to help them find games accurately. Apparently, we implement some filters to help users to do some further selections. However, only on the basis of a good search implementation, these filters could become handy.

Full-text search

Full-text search is a more advanced way to search a database. Full-text search quickly finds all instances of a term (word) in a table without having to scan rows and without having to know which column a term is stored in. Full-text search works by using text indexes. A text index stores positional information for all terms found in the columns you create the text index on. Using a text index can be faster than using a regular index to find rows containing a given value. The most common (and effective) way to describe full-text searches is "what Google, Yahoo, and Bing do with documents placed on the World Wide Web".

FTS4 are SQLite virtual table modules that allow users to perform full-text searches on a set of documents. In our system, we set up a virtual table including games' titles, genres (without one-hot preprocessing, stored in string) and description. With full-text searches, users can find out games they love conveniently.

System Flow

This system gives users some different ways to find games interesting for them: Search, Questionnaire, front page recommendation. This flowchart describes the whole recommendation process.

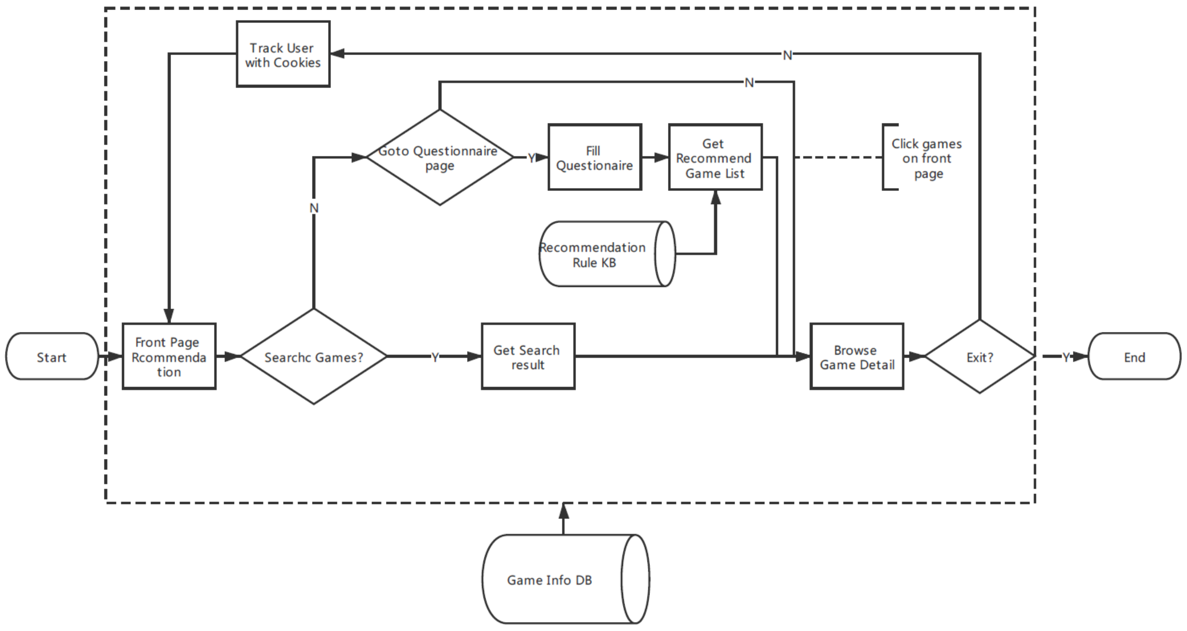


Figure 8 System Flow

Conclusion

Our team successfully constructs a game recommendation platform and deploys it online in this project. Along the way, all of us pick up useful skills.

In the knowledge gathering phase. We got filed knowledge in various ways:

- Crawler is used to collect game information
- Professional people's advice is used to guide the questionnaire design
- Machine learning algorithms can help us find out knowledge about how to measure two games similarity

To implement the system, we use python, flask, SQLite, JavaScript as well as TagUI to develop a basic web application. Furthermore, rule-based and similarity-based recommendation are joined to make the system intelligent.

All these techniques we learn from developing this project are practical and helpful in our further careers.

Thanks to all team members' hard work, we finally finished a nice game recommender. Some of our friends tried this recommender and found games really interesting for them.

Improvement

After we finish this system, we also find there are some points we need to improve in the future:

- User accounts: In the current system, all users are anonymous, user tracking is implemented by using cookies. We plan to develop a user login system so that we can let our users to have their wish lists and track their behaviors more accurately.
- More platforms: Today, more and more players begin to play video games on their mobile phones, we plan to add mobile game recommendation function in next version.

Bibliography

[1] The Business Model and Strategy of Valve Corporation/Steam. Jessica Lam. Michigan State University. MI 452 : Media Entrepreneurship. December 9, 2018. https://medium.com/@j.lam/the-business-model-and-strategy-of-valve-corporation-steam-f7dd799988bb

[2] Crider, M. (2017, June 16). What Is Steam Direct, and How Is It Different from Greenlight? Retrieved from. https://www.howtogeek.com/311311/what-is-steam-direct-and-how-is-it-different-from-greenlight/

[3] Yan Guo, Minxi Wang, Xin Li. Application of an improved Apriori algorithm in a mobile e-commerce recommendation system[J]. Industrial Management & Data Systems, 2017.

[4] Li-Ping Jing; Hou-Kuan Huang; Hong-Bo Shi. Improved feature selection approach TFIDF in text mining[J]. IEEE, 2002.

[5] Zhang Yun-tao, Gong Ling & Wang Yong-cheng. An improved TF-IDF approach for text classification[J]. Journal of Zhejiang University-SCIENCE A, 2005, (6):49-55.

[6] P.S.Bradley, K.P.Bennett, A.Demiriz. ConstrainedK-MeansClustering[J]. Mathematical Sciences, 2000.

[7] Khaled Alsabti, Sanjay Ranka, Vineet Singh. An efficient k-means clustering algorithm[J]. Electrical Engineering and Computer Science, 1997, 43.

[8] Kevin Westcott, Jeff Loucks, David Ciampa, Shashank Srivastava. Digital media trends survey. The Deloitte Center for Technology, Media & Telecommunications, 2019.

Appendix

Project Proposal

Problem

Over the past few years, buying digital games on platforms like Steam gradually become mainstream among players. However, nearly all mature digital game platforms' target users are core gamers who know and played dozens of video games. That's why these platforms are fill of professional game terms like “Metroidvania”, a player with minor gaming experience will always find it’s difficult to pick games from such game platforms. The major problem we want to solve is to help light game players to find video games interesting for them.

Solution

In this project, we are going to build an intelligent system that consists of a concise user interface, powerful algorithms, and the latest game information. We utilize easy-to-understand adjectives and vivid videos to illustrate different game genres to casual players. Then let AI recommend games for them. We hope this system can help these players find games interesting for them and make profits through video game promotions.

Objective

To build a minimum value product (MVP) -- A web-based system with concise UI which can recommend games to non-core players with the help of AI.

Plan & Approach

We plan to implement our web-based system with python and a lightweight python web server framework -- flask. On the basis of it, we will develop rule-based recommendation and similarity recommendation functions powered by rule-engine and NLP algorithms.

Project Schedule

Day1 – Day2	Crawl data, preprocess, and build game information data set.
Day3 – Day5	Develop minimum backend and frontend functions to support the following development.
Day6 – Day8	Collection field knowledge and design questionnaire, then implement rule-based recommendation function.
Day8 – Day10	Implement similarity-based recommendation, tune the algorithm.
Day11-Day12	Integrate the system, then test, and debug.
Day12-Day14	Add search function to the system.
Day15	Usability testing, get feedback.
Day16-Day17	Improve the whole system according to the feedback

Table 4 Project Schedule

System Functionalities Mapping

Machine Reasoning	Rule-based system that we have learnt from this course is used to build the personalized recommendation system and provide some proper recommendation strategies for other recommendation algorithms.
Reasoning Systems	k-means, the algorithm that we are taught in this course, helps us to assign tens of genres of games into 8 classes, which make the implementation of personalized recommendation possible.
Cognitive Systems	TF-IDF is an NLP algorithm that is related to this course, which is also the core of the similarity-based recommendation. The technique that applies TF-IDF on texts to generate the feature vector and use cosine similarity to calculate the similarity score of two vectors in order to make decisions whether these two texts are similar or not is the key thing that I have learnt from this course.

Table 5 System Functionalities Mapping

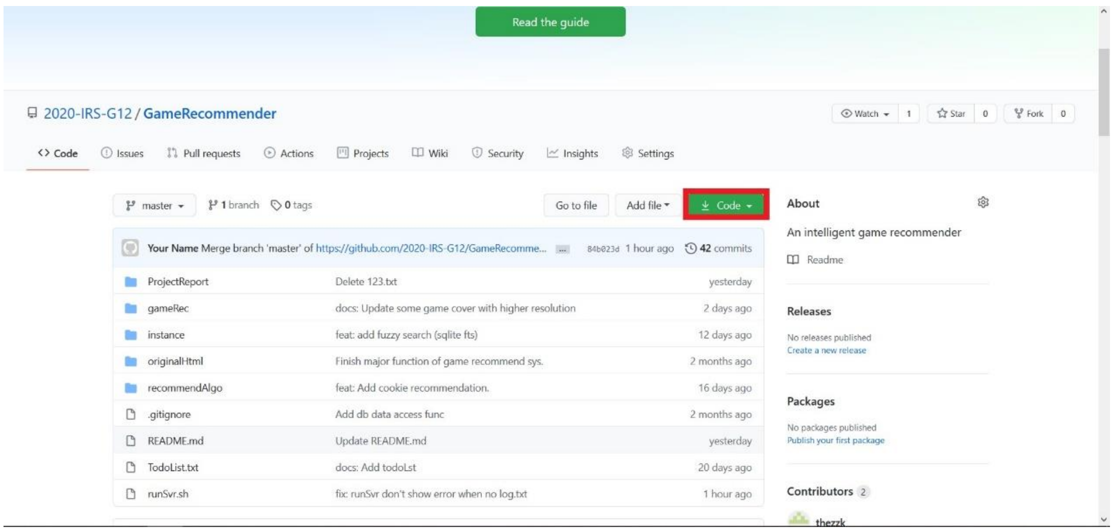
Local Installation Guide

Running Environment

ISS-VM

Clone the project

Clone our project from https://github.com/2020-IRS-G12/GameRecommender


Read the guide

2020-IRS-G12 / GameRecommender

Watch 1 · Star 0 · Fork 0

Code · Issues · Pull requests · Actions · Projects · Wiki · Security · Insights · Settings

master · 1 branch · 0 tags · Go to file · Add file · Code

Your Name Merge branch 'master' of https://github.com/2020-IRS-G12/GameRecomme... 84b023d 1 hour ago 42 commits

ProjectReport	Delete 123.txt	yesterday
gameRec	docs: Update some game cover with higher resolution	2 days ago
instance	feat: add fuzzy search (sqlite fts)	12 days ago
originalHtml	Finish major function of game recommend sys.	2 months ago
recommendAlgo	feat: Add cookie recommendation.	16 days ago
.gitignore	Add db data access func	2 months ago
README.md	Update README.md	yesterday
TodoList.txt	docs: Add todoLst	20 days ago
runSvr.sh	fix: runSvr don't show error when no log.txt	1 hour ago

About

An intelligent game recommender

Readme

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Contributors 2

thezzk

Install Python

https://www.ics.uci.edu/~pattis/common/handouts/pythoneclipsejava/python.html

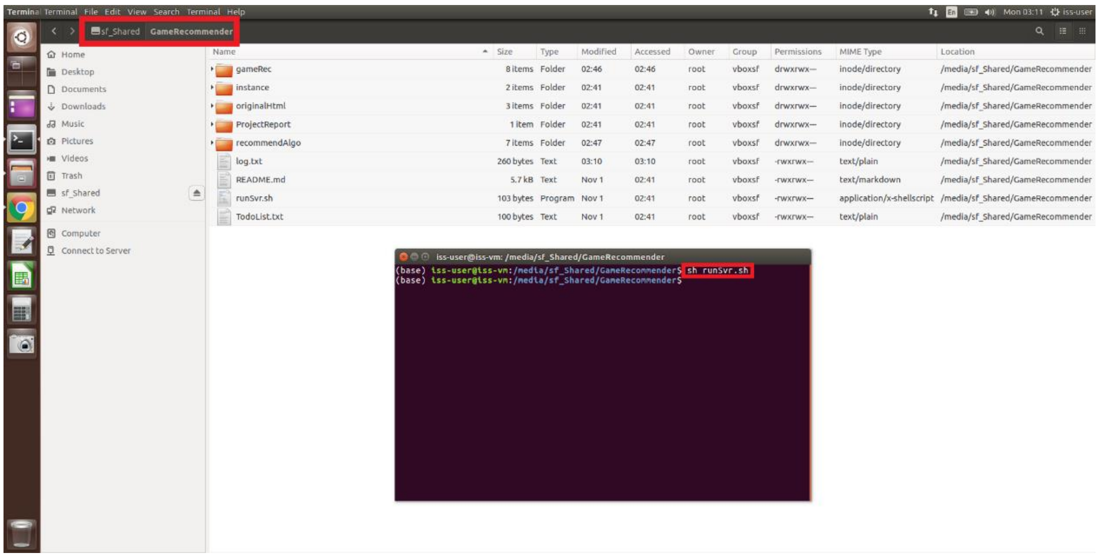
Install Packages

Execute commands:

```
pip install flask

pip install flask --upgrade

pip install rule-engine

pip install pysqlite3
```

Run Project

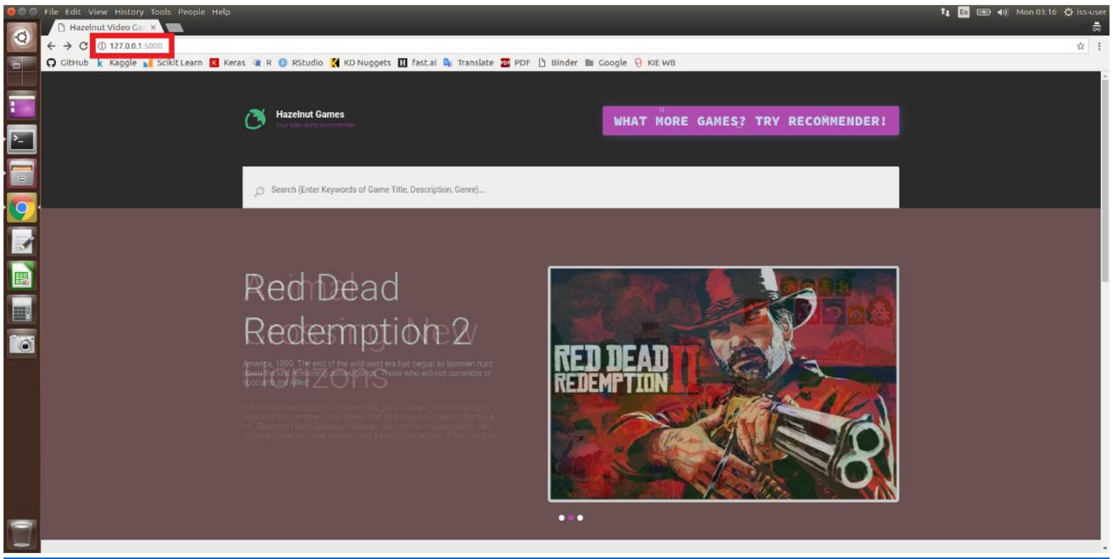
Lunch Terminal and enter the **GameRecommender** directory, then execute the command: sh runSvr.sh



Name	Size	Type	Modified	Accessed	Owner	Group	Permissions	MIME Type	Location
gameRec	8 items	Folder	02:46	02:46	root	vboxsf	drwxrwx—	inode/directory	/media/sf_Shared/GameRecommender
instance	2 items	Folder	02:41	02:41	root	vboxsf	drwxrwx—	inode/directory	/media/sf_Shared/GameRecommender
originalHtml	3 items	Folder	02:41	02:41	root	vboxsf	drwxrwx—	inode/directory	/media/sf_Shared/GameRecommender
ProjectReport	1 item	Folder	02:41	02:41	root	vboxsf	drwxrwx—	inode/directory	/media/sf_Shared/GameRecommender
recommendAlgo	7 items	Folder	02:47	02:47	root	vboxsf	drwxrwx—	inode/directory	/media/sf_Shared/GameRecommender
log.txt	260 bytes	Text	03:10	03:10	root	vboxsf	-rwxrwx—	text/plain	/media/sf_Shared/GameRecommender
README.md	5.7 kB	Text	Nov 1	02:41	root	vboxsf	-rwxrwx—	text/markdown	/media/sf_Shared/GameRecommender
runSvr.sh	103 bytes	Program	Nov 1	02:41	root	vboxsf	-rwxrwx—	application/x-shellscript	/media/sf_Shared/GameRecommender
TodoList.txt	100 bytes	Text	Nov 1	02:41	root	vboxsf	-rwxrwx—	text/plain	/media/sf_Shared/GameRecommender

```
iss-user@iss-vm: /media/sf_Shared/GameRecommender
(base) iss-user@iss-vm:/media/sf_Shared/GameRecommender$ sh runSvr.sh
(base) iss-user@iss-vm:/media/sf_Shared/GameRecommender$
```

Open the browser and key in the address**:** http://127.0.0.1:5000/



Then, you can use our system following the User Guide.

Supported Browsers

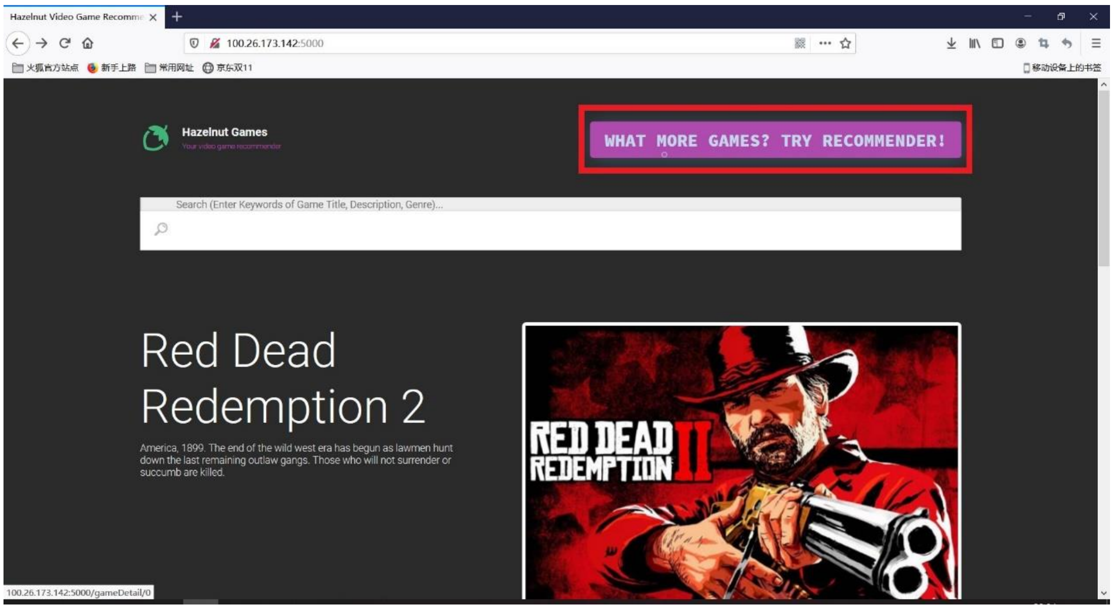
Firefox, Google Chrome

User Guide

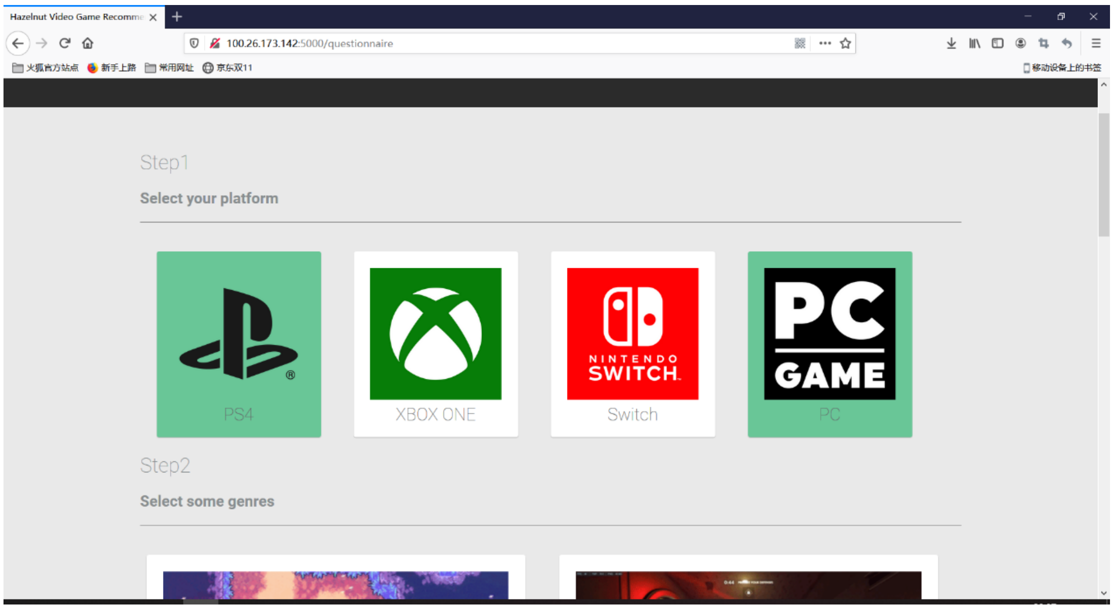
Please click this link http://100.26.173.142:5000/

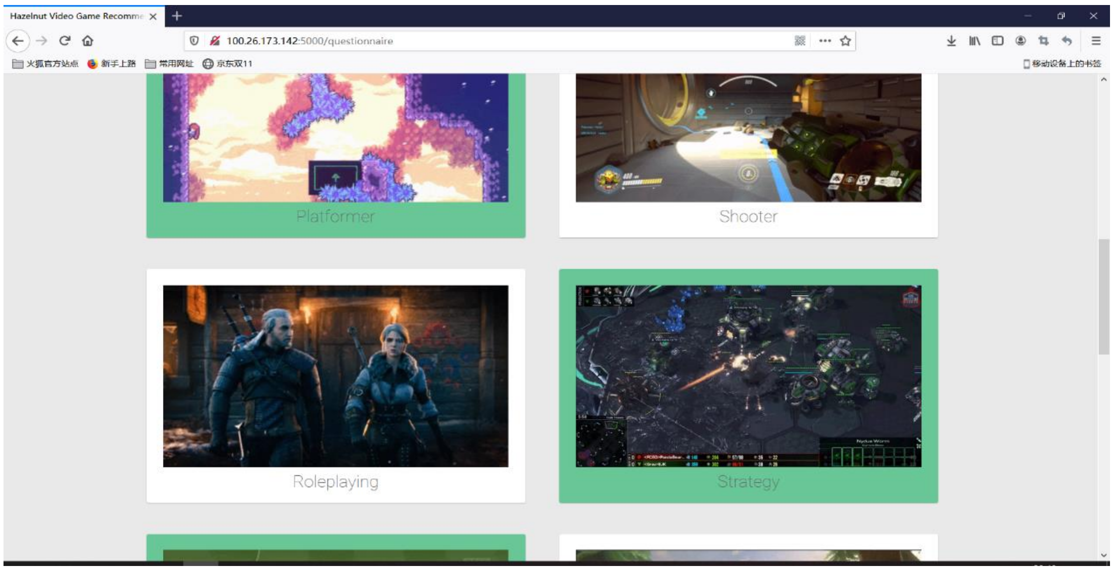
We recommend you to use Firefox or Google Chrome

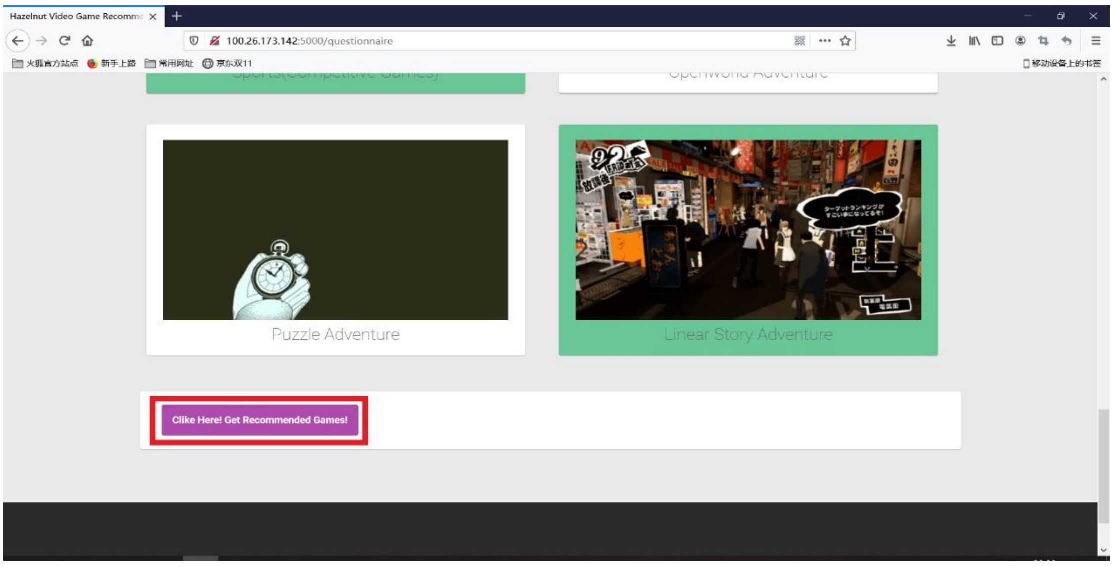
You can click the button in the red block to customize your recommendation game list


Hazelnut Video Game Recomm
100.26.173.142:5000
Hazelnut Games
Your video game recommender
WHAT MORE GAMES? TRY RECOMMENDER!
Search (Enter Keywords of Game Title, Description, Genre)...
Red Dead Redemption 2
America, 1899. The end of the wild west era has begun as lawmen hunt down the last remaining outlaw gangs. Those who will not surrender or succumb are killed.
100.26.173.142:5000/gameDetail/0

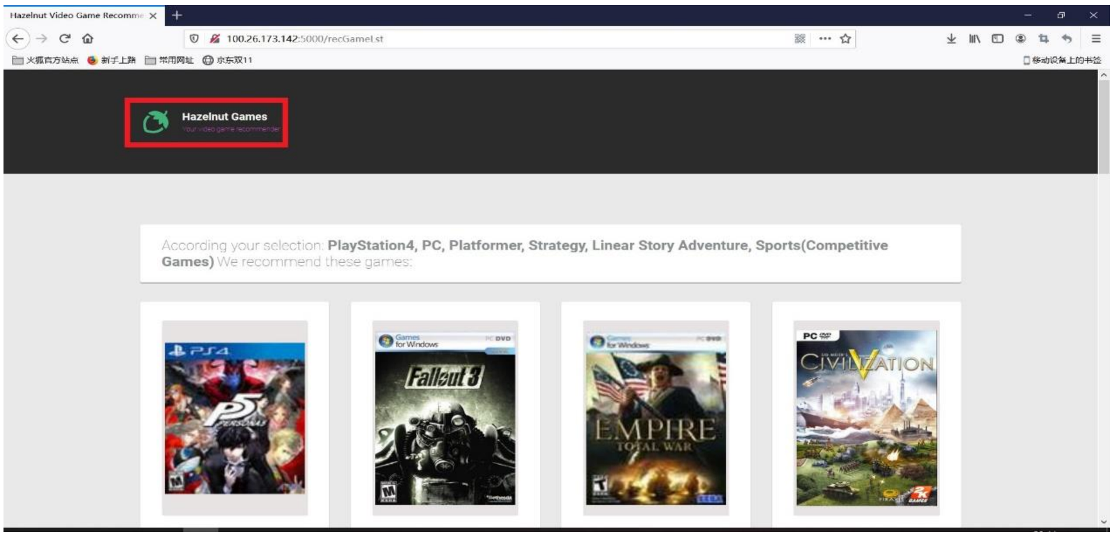
There is a questionnaire which have two question to ask you to complete. For each question, you can make multiple choice and you can select nothing. Finally, you click the button in the bottom of the website. You will get 16 games that system recommends for you according to the information you provide.




Hazelnut Video Game Recomme
100.26.173.142:5000/questionnaire
Platformer
Shooter
Roleplaying
Strategy

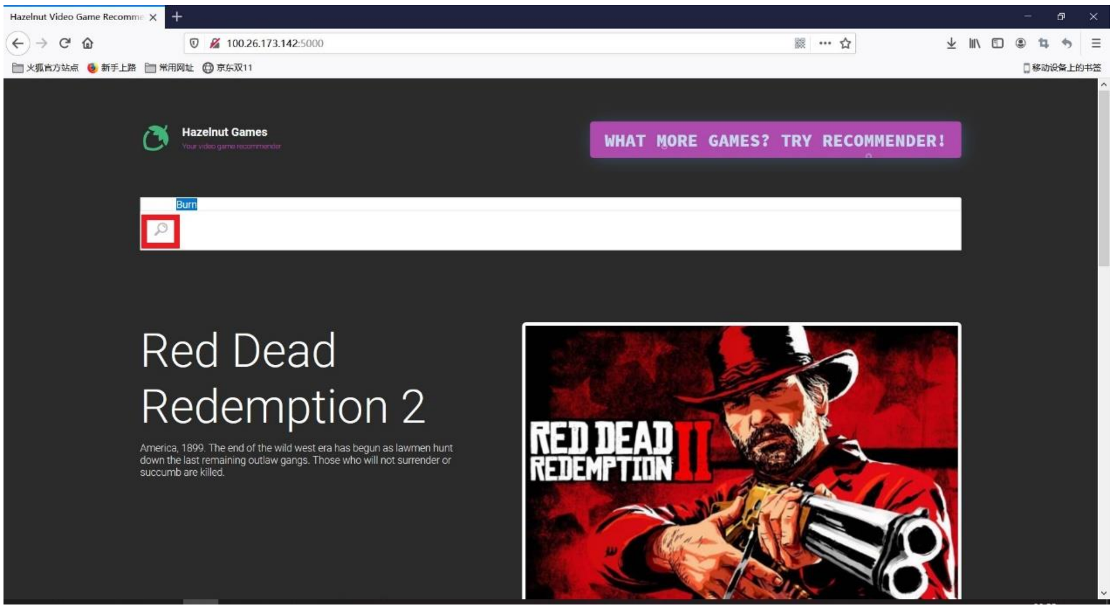

Hazelnut Video Game Recomme
100.26.173.142:5000/questionnaire
Puzzle Adventure
Linear Story Adventure
Clike Here! Get Recommended Games!

Click the icon that you can back to the first page



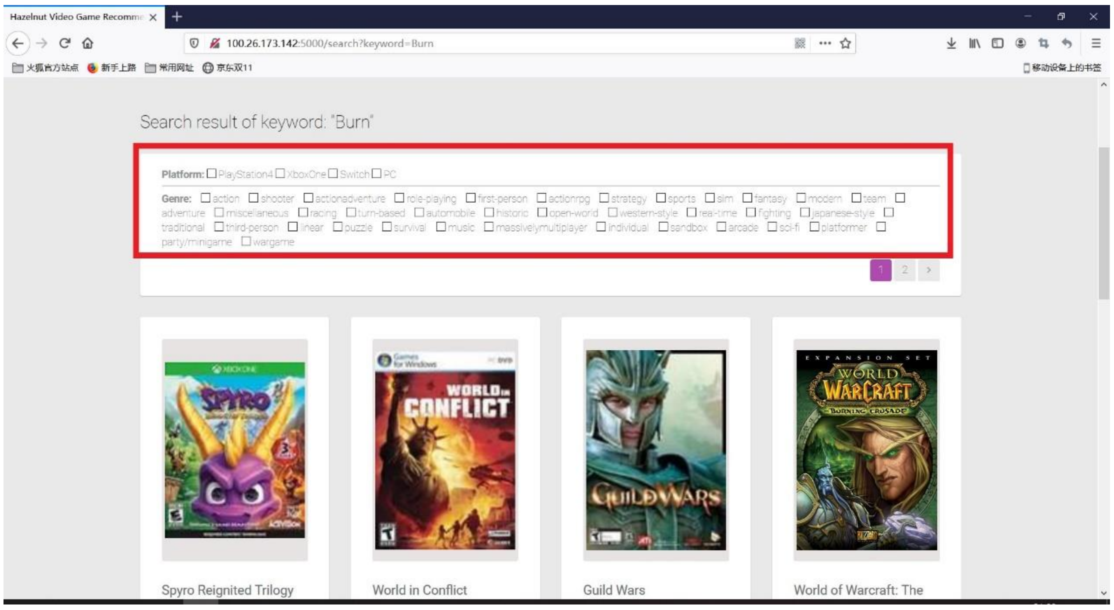
According your selection: **PlayStation4, PC, Platformer, Strategy, Linear Story Adventure, Sports(Competitive Games)** We recommend these games:

You can type the name of game you are interested in or just type some keywords, even type nothing. Then, press the search button you will get some result. You also can use the check boxes in red block to reduce the searching results.


Hazelnut Games
Your video game recommender
WHAT MORE GAMES? TRY RECOMMENDER!
Burn

Red Dead Redemption 2

America, 1899. The end of the wild west era has begun as lawmen hunt down the last remaining outlaw gangs. Those who will not surrender or succumb are killed.


Hazelnut Video Game Recomme
100.26.173.142:5000/search?keyword=Burn
火狐官方站点
新手上路
常用网址
京东双11
移动设备上的书签

Search result of keyword: "Burn"

Platform: ☐ PlayStation4 ☐ XboxOne ☐ Switch ☐ PC

Genre: ☐ action ☐ shooter ☐ actionadventure ☐ role-playing ☐ first-person ☐ actionrpg ☐ strategy ☐ sports ☐ sim ☐ fantasy ☐ modern ☐ team ☐ adventure ☐ miscellaneous ☐ racing ☐ turn-based ☐ automobile ☐ historic ☐ open-world ☐ western-style ☐ real-time ☐ fighting ☐ japanese-style ☐ traditional ☐ third-person ☐ linear ☐ puzzle ☐ survival ☐ music ☐ massivelymultiplayer ☐ individual ☐ sandbox ☐ arcade ☐ sci-fi ☐ platformer ☐ party/minigame ☐ wargame

Spyro Reignited Trilogy

World in Conflict

Guild Wars

World of Warcraft: The

Individual reports

Zhang Zekun

1. Your personal contribution to the project.

In this project, I responsible for the backend programming and data collection. My contribution includes:

- Construct this system’ s backend with python, flask and SQLite.
- Deploy the system on AWS.
- Write program on the basis of Scrapy to crawl video game data from Internet.
- Create an RPA agent to find high resolution images by reversing image search with low resolution images.
- Provide professional advice about the questionnaire design in our system. (I was a former game developer before study in NUS-ISS)

2. What you have learnt from the project.

- Use flask to implement web server.
- Rule-based (rule-engine) and similarity-based (TF-IDF, Count Vectorizer) recommendation algorithm.
- Utilize crawler to collect data from Internet.
- Let RPA agent to deal with repetitive works.

- Deploy my own backend on cloud server like AWS.
- System integration among machine learning model and web application.
- Manage data with SQLite, and enable full-text search with fts4.
- How to program with jQuery and Ajax

3. How you can apply this in future work-related projects.

- In this project I became more skillful on implementing web applications. Web development is very popular today. And in my former job, we occasionally communicated with web servers in our mobile game clients. Because these functions were seldom used and not been tested comprehensively. Sometimes they were not very stable and had subtle bugs. Unfortunately, I could not fix all of them thoroughly at that time. With the knowledge I learnt from this project, I think I can handle such bugs in the next time.
- I used SQLite's full-text search function – fts4 in this project. The knowledge of how to utilize utf4 can be very useful when I need to create a search function in mobile games without network connection. It enables users to search local data (e.g. inventory system in games) just like searching on Google or Bing.
- In order to help me deal with repetitive works, I developed an RPA agent to find high resolution images. And I also find RPA can be very useful in game industry. For example, if we need multiple software in sequence to build an android package. We can use RPA to replace human to operate them one by one. It will save a lot time.
- I learned how to use crawler in this project. I think it's really a powerful tool to collect necessary data when I want to do some analysis but have no ready-to-use data. For instance, crawling chats from a video game forum around a certain game can help us analysis its gaming experience.

Shi Zhaoheng

1. Your personal contribution to the project.

- Choose algorithm for recommendation
- Design recommendation system
- Complete recommendation system with Python
- Finish recommendation Game Recommender part of the report
- Data preprocessing to build a good construction dataset
- Do clustering to reduce the number of genres for personalized recommendation.

2. What you have learnt from the project.

In this project, the first thing I have learnt is to use Python to do programming. I never code by using Python before this project and Python is the most popular language for machine learning, which is critical for me to learn.

Then, I know how to build a website. Generally, there are two parts to design a website: frontend and backend.

Normally, frontend means Web page design that is a weak point for me. I have no idea how to make a good-looking Web page. Because people always feel happy to see something have a good appearance, no matter how powerful your system is, if it looks bad, users may not be willing to use it. As a result, to make a beautiful Web page becomes dramatically important. However, I know nothing about the design, so the only thing I can do is to find some website which have the same theme with my website and I will treat it as a template to modify it to fit what I need, but it is necessary to keep the main style of the template. Imitation is the start point of learning everything, I believe that when I learn many templates, I finally can build my own style for Web page design.

Let us talk about backend. We choose a Python web framework called flask that is a lightweight WSGI web application framework to build backend. The main function of backend is to process the request of user. The frontend always sends some information from users to backend and backend will do something according to what information it gets. When it finishes processing, it will return to frontend (Web page). In other word, frontend is the bridge linking users with backend.

Also, I learn some NLP algorithms.

- TF-IDF An algorithm discards those words that appear in every document and focus on the unique words
- CountVectorizer An algorithm is used to transform a given text into a vector on the basis of the frequency (count) of each word that occurs in the entire text.

I use these two algorithms as the core of my recommendation system which is called content-based recommendation system.

And I learn how to build a package for others to use. According to my experience, it should be a black box, which means that I only provide a function to the user, he/she does not need to know anything I implement inside. What the user should know is what can my function complete and how to apply it.

Finally, I know how to assign work to members. The work of each member should not have a strong connection between each other, which will make system integration easier.

3. How you can apply this in future work-related projects.

Firstly, nowadays, many systems present as websites, so the demand for people who can build website in the Internet industry increases dramatically. And I have learnt how to build a website from this project. What’s more, I am not only able to design Webpages, but also able to complete a backend system by myself, while many companies only require a person who knows one of them. Because of the huge demand in the market, I have many opportunities to use my Web knowledge to build some Web systems for other projects. For example, many models need the user interface to let users input information the model requires. Only after those models receive the information, they can show the powerful function of them, just like our system. So, Website that can be simply found some templates from the Internet will be a good choice for the user interface.

Secondly, NLP techniques are also applied in many fields, such as real-time translation, automatic article generation and so on. So, the skill of NLP can also help me to complete related jobs in the future. For example, I can apply TF-IDF to implement an article classification system. I will put the contents of the articles in this algorithm, then it will generate some feature vectors about those articles. Finally, I will apply cosine similarity function to calculate the similarity score about those articles. If the score is larger than a threshold, those articles will be considered as the same type.

Next, k-means is a well-known unsupervised learning algorithm for classification. Just like the example above, I do not need to calculate the similarity score among those articles. I can apply k-means to deal with those feature vector and those vectors can be assigned into several clusters that are defined by me. Finally, I can summarize the theme of each cluster and name them. After that, the classification is finished.

Also, those techniques that I apply in this model can help me to build other recommendation systems, such as movie recommendation. The description of the movie will be addressed by TF-IDF, while the genre will be processed by CountVectorizer, and the system will also be shown as a Website.

Tao Xiyan

1. Your personal contribution to the project.

- Participate in Project topic selection;
- Crawling data (game videos links)；
- Participate in the overall design of the first version of the project;
- Participate in Project schedule planning;
- Complete the search function(full-text search), based on FTS4, a tech in SQLite;
- The overall design of the final version of the project;
- Video split design, video script & narration writing;
- Design the PPT for display our algorithms and system design and for the video;
- Report writing (Business Case / Product Plan / Market Research/Full-text search/etc.)

2. What you have learnt from the project.

- How to write a crawler and crawl data (Data is very important for machine learning.)
- How to write front-end html5 and CSS.
- How to use SQLite to realize the search function of background data.
- How to integrate the front and back ends.
- How to look at the entire project (business value, business outlook, business goals) with leadership thinking.
- How to design advertising video, write advertising video script, design advertising video storyboard.

Now, I have some experience in participating in the establishment of the entire project to the later publicity. After all, I was only involved in back-end development or algorithms in my previous work and study, and my vision was very cramped. This project brought me a different learning experience for computer students.

3. How you can apply this in future work-related projects.

In this project, I used a crawler to crawl online game data. In the process of completing the crawler code, I have a deeper understanding of the structure of the web page. This technology will be applied to future big data related work.

Then, for the obtained user data, we used the K-means algorithm for clustering. This machine learning algorithm is simple but effective. It is a good choice for solving the classification problem of big data in future work.

At the same time, the construction of the front-end and the backend of the webpage are both skills that are often used in Internet companies. In this project, I learned and completed the tasks of these two parts. I believe it will be of great help to future work.

Finally, we jointly produced the project’s promotional video, completed the business plan, etc., which allowed me to understand the entire process of a project from development to landing. I believe this project experience will lay a good foundation for my future Internet-related work.