Optimal Search Strategies for the Parlor Game "Twenty Questions"

Jialin Guo
Department of Applied Math and Statistics
Johns Hopkins University
Baltimore, MD 21218
jguo81@jhu.edu

Abstract

The classic parlor game "20 Questions" challenges players to deduce a specific object by asking a series of yes-or-no questions. It is a competition between two players, and the objective is to guess the object correctly within limited amount of questions. In this project, we developed a strategy to ensure a dynamic and optimal selection of questions at each step. The focus of our interest is name of "famous" people. To collect data, we randomly generated a list of famous names and their attributes through GPT-4, and represented those data as a large binary matrix with 1,016 rows(names) and 146 columns(attributes). Given the assumption that each name is equally likely to be initially selected, we applied the concepts of *Information Pursuit*(IP) and maximized the *Conditional Mutual Information*(CMI) at each step. This approach ensured that each selected attribute provided the most information in terms of reducing uncertainty. As a result, we found that the best questions were those that could partition the remaining population into two almost equal parts, and over 85% of names were identified within 10 questions. Meanwhile, we developed an online webpage for playing the game interactively and visualizing the narrowing down process.

1 Introduction

"20 Questions" is a classic parlor game enjoyed by people of all ages. Simple yet engaging, it is a face-to-face guessing game between two players, where the first player thinks of an object (which could be a person, a place, or a concept), and the second player attempts to guess it by asking a series of non-trivial "yes-no" questions. The first player responds accurately to these inquiries. Each question should be strategically chosen based on the responses to previous ones. The aim of the game is to identify the target object using as few questions as possible.

In this project, the focus is on the names of "famous" people. The target person could be from any profession, dead or alive, and is recognized as a well-known historical figure or celebrity. For instance, Taylor Swift is a renowned American singer. The series of "yes-no" questions are centered around the attributes of a person, such as: "Dead?", "Female?", "American?", "Singer?", etc. Our objective is to develop a strategy that uses as few questions as possible to identify the target name.

2 Data Collection

Scripting names one by one from the internet is slow and inefficient. Therefore, we utilized the recent Large Language Model, GPT-4, to generate a substantial list of famous people (1,016 in total) across 12 different categories (e.g., Music, Tech, Politics, Literature, etc.). We collected basic attributes and specific descriptors for each individual. To minimize potential biases, we diversified the characteristics within names, considering aspects such as professions, genders, nationalities, etc., throughout the collection process. We also replaced some less popular names based on the number of search hits using `pytrend`. After creating our original dataset, we manually grouped some specific entries in columns into more general categories. Subsequently, we represented the data as a large binary matrix, with the first column being the target names and all remaining columns representing binary attributes containing only "0" and "1". The final dataset consists of 1,016 rows and 146 columns.

2.1 Original Dataset

The original dataset contains 1016 rows and 23 columns[Figure 1] and its data are consisted of two parts: names and attributes.

	Name	Gender	Group	Nationality	Status	Birth Year	Death year	Age	Marriage	Education Level	...	Gold Metals	Active Player	Political	Forbes	Grammy
0	Virginia Apgar	1	Medicine	United States	1	1909	1974	65	0	D	...	0	0	NaN	0	0
1	Florence Nightingale	1	Medicine	United Kingdom	1	1820	1910	90	0	SS	...	0	0	NaN	0	0
2	Alice Ball	1	Medicine	United States	1	1892	1916	24	0	G	...	0	0	NaN	0	0
3	Patricia Bath	1	Medicine	United States	1	1942	2019	77	1	D	...	0	0	NaN	0	0
4	Linda Buck	1	Medicine	United States	0	1947	2023	76	0	D	...	0	0	NaN	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1011	Federico Fellini	0	Film	Italy	1	1920	1993	73	1	U	...	0	0	NaN	0	0
1012	Akira Kurosawa	0	Film	Japan	1	1910	1998	88	1	C	...	0	0	NaN	0	0
1013	Ingmar Bergman	0	Film	Sweden	1	1918	2007	89	1	C	...	0	0	NaN	0	0
1014	John Huston	0	Film	United States	1	1906	1987	81	1	H	...	0	0	NaN	0	0
1015	Sergio Leone	0	Film	Italy	1	1929	1989	60	0	U	...	0	0	NaN	0	0

1016 rows × 23 columns

Figure 1: Original Dataset

2.1.1 Name

The first column of our dataset is labeled "Name," and it contains names collected from 12 categories: *Medicine, Sports, Politics, Law, Military, Technology, Finance, Art, Literature, Music, Fashion,* and *Film*. To populate this column, we utilized GPT-4, prompting it with: "Within the X category, generate 100 different famous people from around the world, both males and females." However, we noticed that for some fields, the generated names were not widely familiar. Since the game's enjoyment hinges on players' familiarity with the chosen person, obscure names could potentially lessen the experience.

To address this, we integrated the name column into the Google Trend API, leveraging the `pytrend` package to collect the number of search hits for these names over the past five years. This step ensured that the names included in our dataset were recognizable and relevant to a broad audience. Below are some sample distributions of the hits within the Politics and Music categories, demonstrating the variance in public interest and recognition for individuals in these fields.[Figure 2]

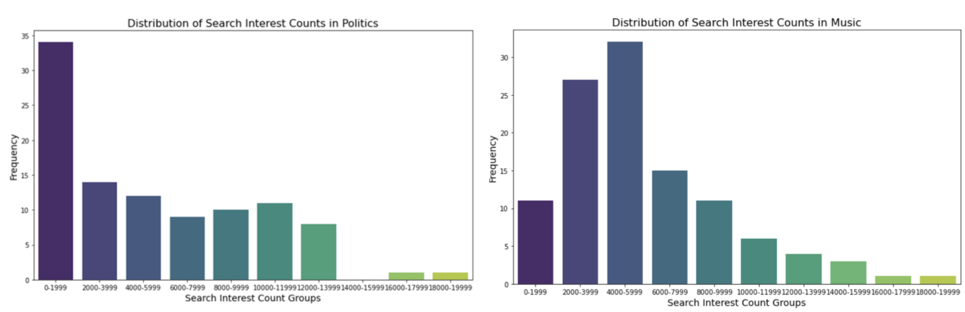

Figure 2: Search hits for Politics and Music

Names with below *2000* hits were replaced with popular ones or deleted. The final list contains 1016 names of famous people and the amount of names within categories varies. [Figure 3]

Category	Frequency
Medicine	81
Sports	100
Politics	98
Law	64
Military	60
Technology	75
Finance	72
Art	81
Literature	82
Music	109
Fashion	81
Film	113

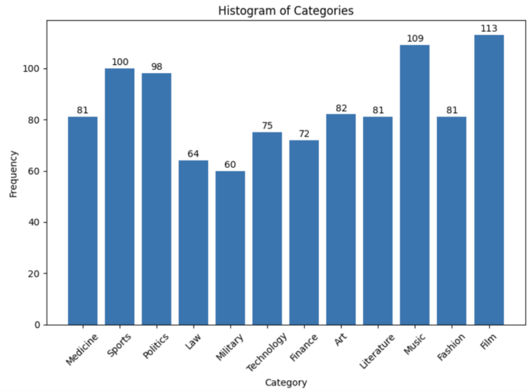

Figure 3: Category Distribution

2.1.2 Attributes

All remaining twenty-two columns are attributes of individuals. Some of those attributes are general information (e.g., Gender, nationality, education) and others are specific descriptors (e.g., Profession, prize winning, service year).

2.2 Final Dataset

To facilitate the posing of yes-or-no questions, the entries within the columns of our dataset are exclusively binary (either 0 or 1). After building the original dataset, we transformed it into a large binary matrix. In this matrix, the rows are indexed by names, encompassing a total of 1,016 rows, and the columns are indexed by attributes, amounting to 146 columns. Within this matrix, an entry of "1" indicates that a person possesses a particular attribute, while an entry of "0" signifies the absence of that attribute. This binary representation is essential for efficiently structuring the data to align with the yes-or-no format of the "20 Questions" game. [Figure 4]

	Name	Female?	Dead?	Married?	Arts and humanities?	Fine Arts?	Humanities?	Gov?	Industry?	Medicine?	...	Biologist?	British?	Chinese?
0	Virginia Apgar	1	1	0	0	0	0	0	1	1	...	0	0	0
1	Florence Nightingale	1	1	0	0	0	0	0	1	1	...	0	1	0
2	Alice Ball	1	1	0	0	0	0	0	1	1	...	0	0	0
3	Patricia Bath	1	1	1	0	0	0	0	1	1	...	0	0	0
4	Linda Buck	1	0	0	0	0	0	0	1	1	...	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1011	Federico Fellini	0	1	1	1	0	1	0	0	0	...	0	0	0
1012	Akira Kurosawa	0	1	1	1	0	1	0	0	0	...	0	0	0
1013	Ingmar Bergman	0	1	1	1	0	1	0	0	0	...	0	0	0
1014	John Huston	0	1	1	1	0	1	0	0	0	...	0	0	0
1015	Sergio Leone	0	1	0	1	0	1	0	0	0	...	0	0	0

1016 rows × 146 columns

Figure 4: Final Dataset

The conversion contained two steps: grouping and encoding.

2.2.1 Grouping

In some columns, the entries of attribute are highly unique. For example: in col Sport Profession, players play all kinds of sports, like tennis, badminton, swimming, fencing etc. To avoid data fragmentation, we manually grouped those entries into broader groups.

Again with the example of Sport Profession, players were further separated to four types of professions.[Figure 5]

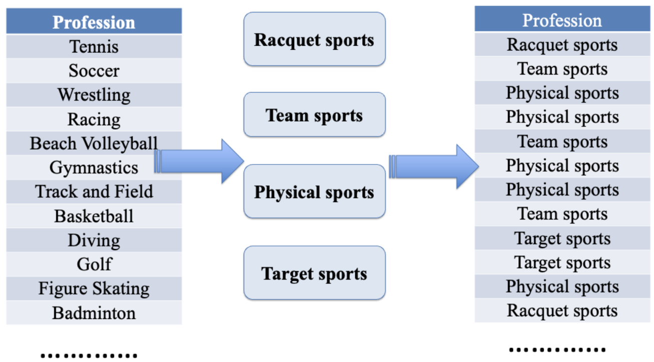

Figure 5: Restructured Group

2.2.2 Encoding

After we restructured some of the columns, our next step was to simply apply one-hot encoding to all cols and turned them from categorical values to binary values. Then, we renamed columns names to be short and precise[Figure 6]. Finally, we obtained our dataset of 1016 rows and 146 cols. [Figure 4]

Profession
Racquet sports
Team sports
Physical sports
Physical sports
Team sports
Physical sports
Physical sports
Team sports
Target sports
Target sports
Physical sports
Racquet sports

Racquet sports?	Team sports?	Physical sports?	Target sports?
1	0	0	0
0	1	0	0
0	0	1	0
0	0	1	0
0	1	0	0
0	0	1	0
0	0	1	0
0	1	0	0
0	0	0	1
0	0	0	1
0	0	1	0
1	0	0	0

Figure 6: One-Hot Encoding

2.3 Disclaimer

There are two points worth mentioning.

- All names and attributes collected by GPT4 are up to date as of 2022.
- About 10% of the names have identical attribute entries; therefore, the model is not able to find the chosen name in these cases. It will find a group of names instead, and the chosen name is within that group.

3 Approach

For efficient search, at each step, we searched for the optimal question to ask in order to gain the most information in the sense of reducing uncertainty in an information-theoretic sense. Building on the assumption of uniform distribution among all names, we applied the *Information Pursuit*(IP) algorithm and designed a search strategy that dynamically iteratively chooses the next question to be the one with maximum *Conditional Mutual Information*(CMI) at each time.

- *Information Pursuit*(IP)[1]: a sequential algorithm that iteratively selects the most informative attributes(question) based on *Conditional Mutual Information*(CMI). Since finding the globally optimal strategy may be conceptually challenging, but we can certainly learn a greedy approximation. IP works best under this circumstance.
- *Conditional Mutual Information*(CMI) [2]: $I(Q;Y|R) = H(Q|R) - H(Q|Y,R)$ Given Y = target, R = questions and answers to date, Q = next question(attribute) to be selected which maximizes $I(Q;Y|R)$ at each step. To maximize CMI, we wanted to maximize $H(Q|R)$ and minimize $H(Q|Y,R)$.
  - $H(Q|R)$: conditional entropy of next question(Q) given previous responses(R). We maximized this.
  - $H(Q|Y,R)$: conditional entropy of Q given the target Y and R. This is 0 since knowing the target(Y) leaves no uncertainty about its question(Q).

It is known that this strategy yields a mean search time within one unit of the best possible mean time which is the entropy of Y.

At each step, a dictionary was created with keys = attribute and value = CMI. The attribute with the maximum CMI was selected as the *Best Question* and proceed to retrieve the response. If the values tied, then the attribute appeared first according to the original sequence of columns would be selected. Then, the question and answer were appended to R.[Figure 7]

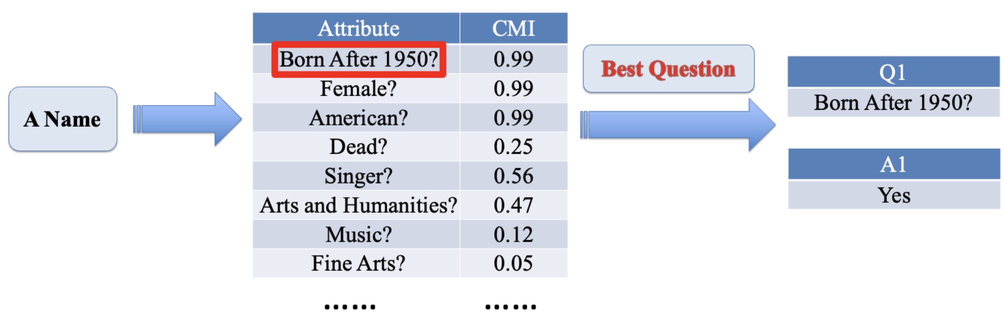

Figure 7: Select Best Question

We updated the remaining matrix with the responses(R) obtained and removed the used attribute from remaining columns. This iterative process continued until we reached the end condition: either only one name remained or no other information could be gained. Then we returned the results.[Figure 8]

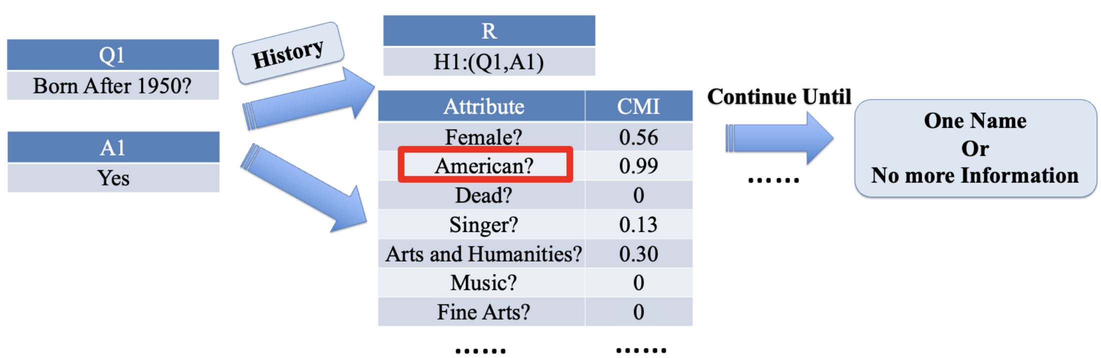

Figure 8: Iterative Process

Throughout the process, we used matrix and function operations to improve efficiency and save computation power. Sanity checks were also implemented to check if $H(Q|Y,R) = 0$ to ensure the accuracy of calculation.

4 Results

We fed our dataset into the search algorithm and created the game-play. At the beginning, the game took in a name from player and generated the narrowing down process including selected questions, answers and history of responses. Finally, it returned with the resulted name. The result contained two cases. We build a representation of a question history to keep track of the narrowing process.

First Case: enough information was obtained, and a single name was found. In this example, Harry Styles is a famous British singer, and we narrow down his case and find him successfully.[Figure 9]

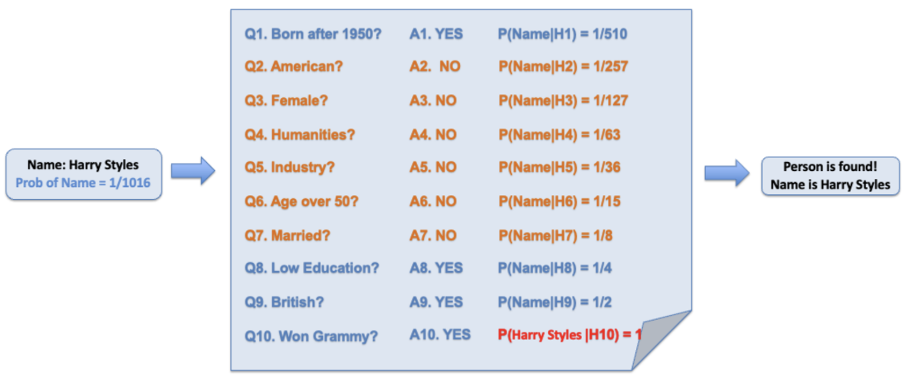

Figure 9: Sample Run - Case One

Second Case: remaining characters had the same information gain, and exact name can't be found. Then the chosen name was in a group of names. In this example, Taylor Swift is a famous American singer, and she shares the same attribute entries with another American singer, Christina Aguilera. Therefore the output contains both names. [Figure 10]

Figure 10: Sample Run - Case Two

We recorded the selected attributes during each question step. The plots indicated: for the first and second questions, the algorithm always asked "Born after 1950?" and "American?"; the attributes varied staring on the third question and so forth. Similar to decision tree, the selected attributes branched out further as more questions were asked.[Figure 11, Figure 12, Figure 13, Figure 14, Figure 15, Figure 16, Figure 17, Figure 18]

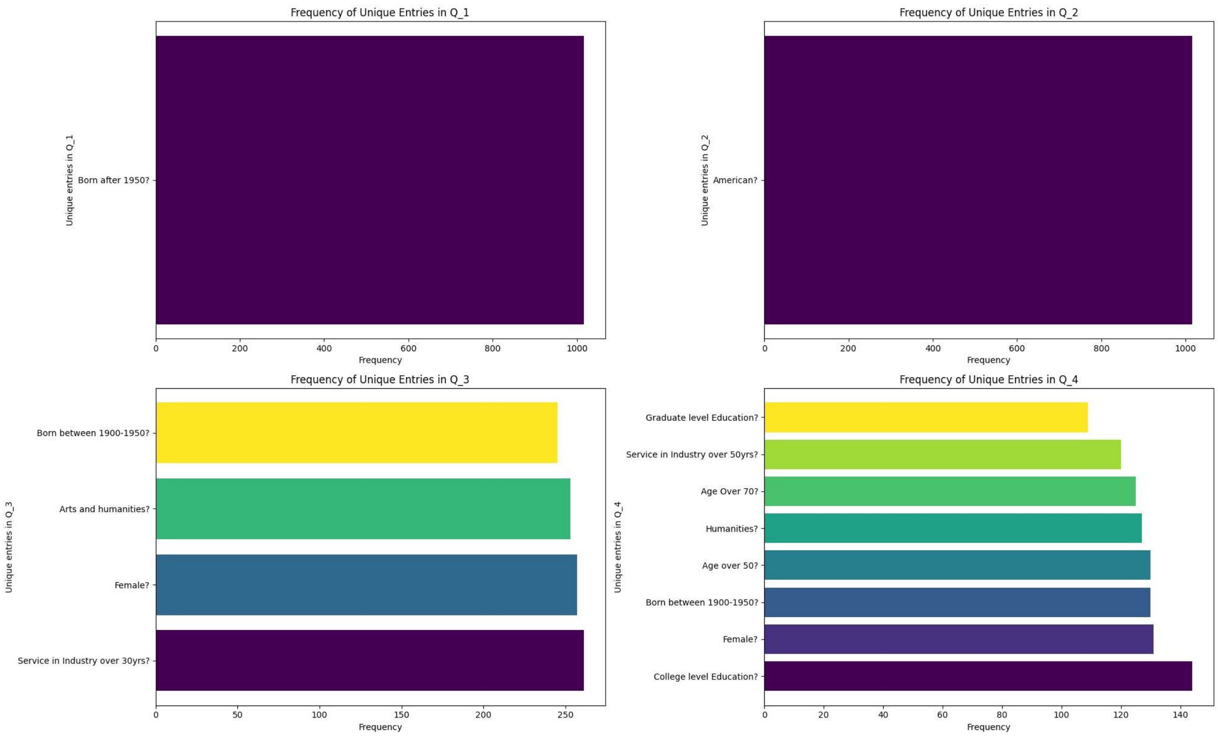

Figure 11: Distribution of selected attributes for best question at step 1-4

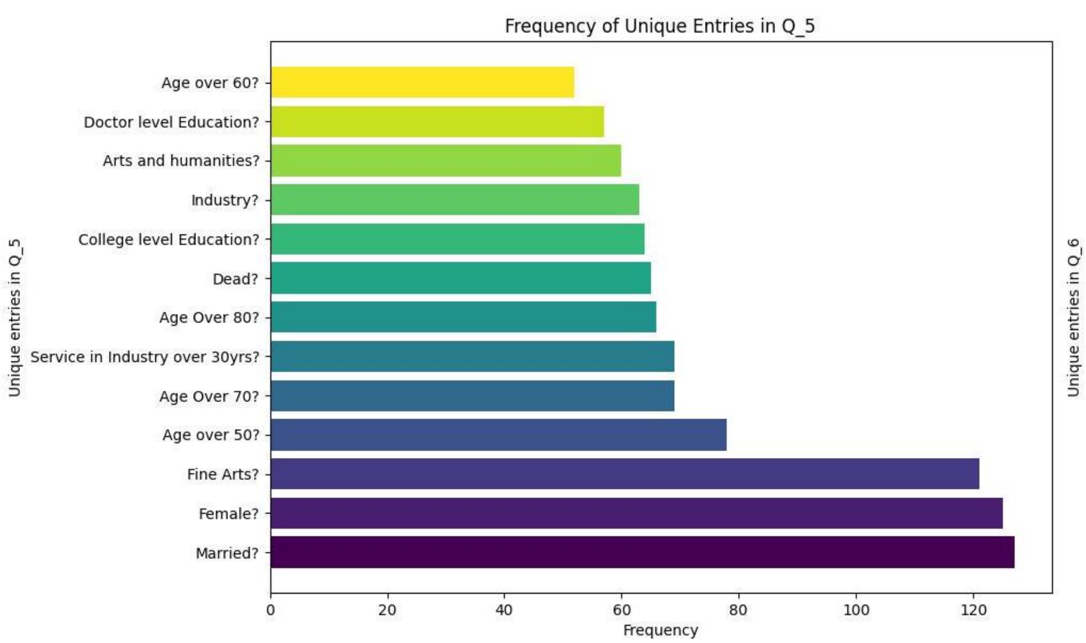

Figure 12: Distribution of selected attributes for best question at step 5

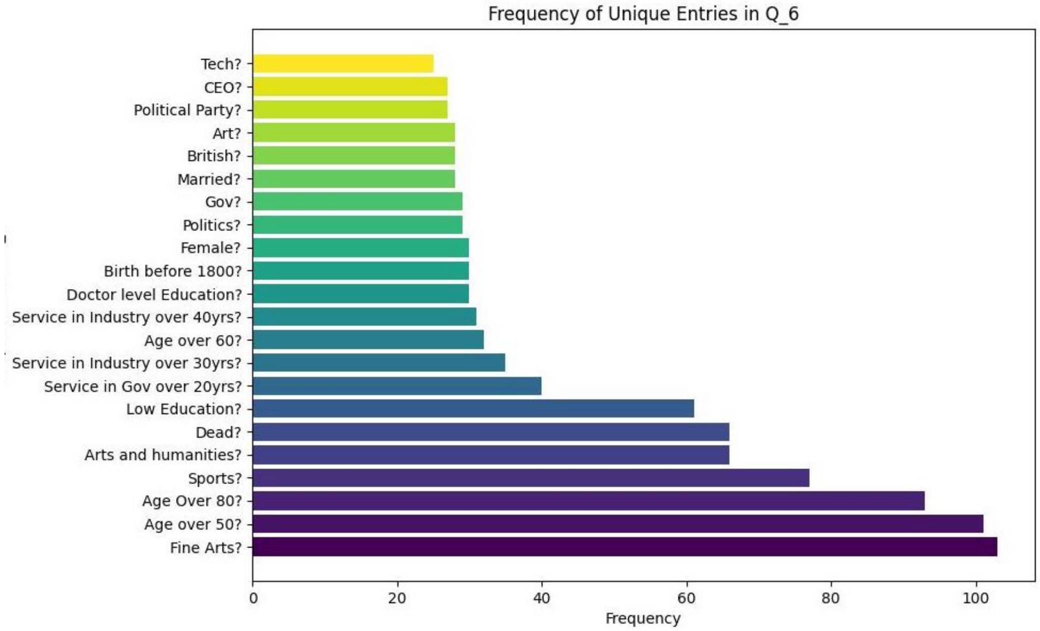

Figure 13: Distribution of selected attributes for best question at step 6

Due to different answers to previous questions, there are quite a number of different next questions to ask. Therefore, the x-axis may be blurry and hard to read due to the large amount of different questions present.

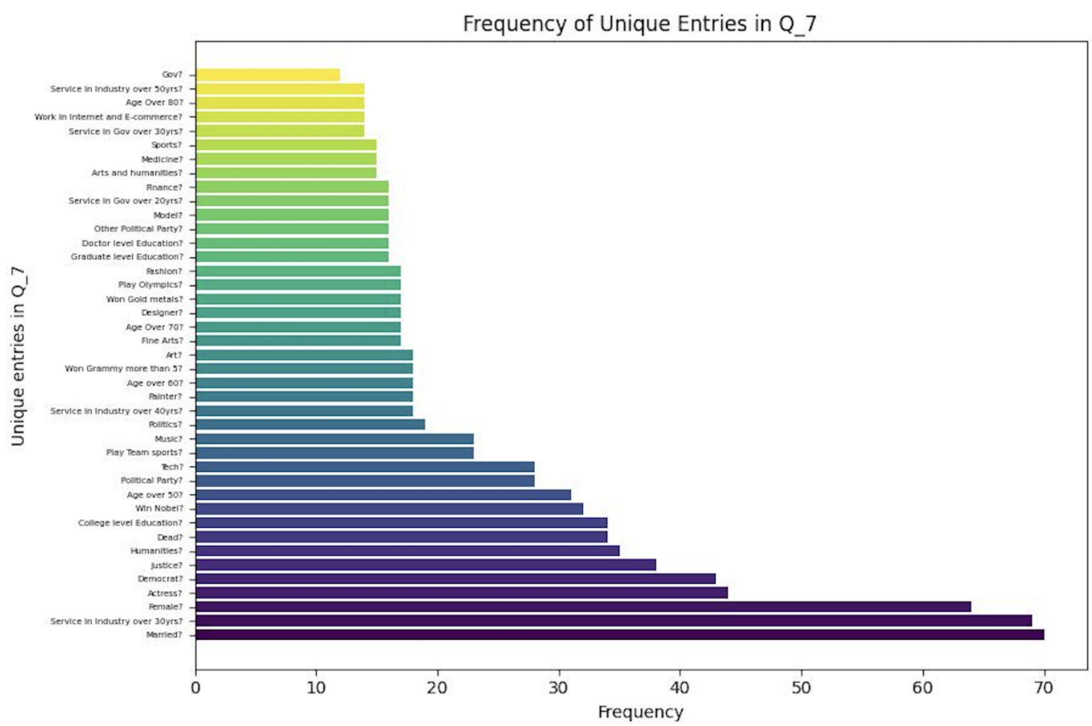

Figure 14: Distribution of selected attributes for best question at step 7

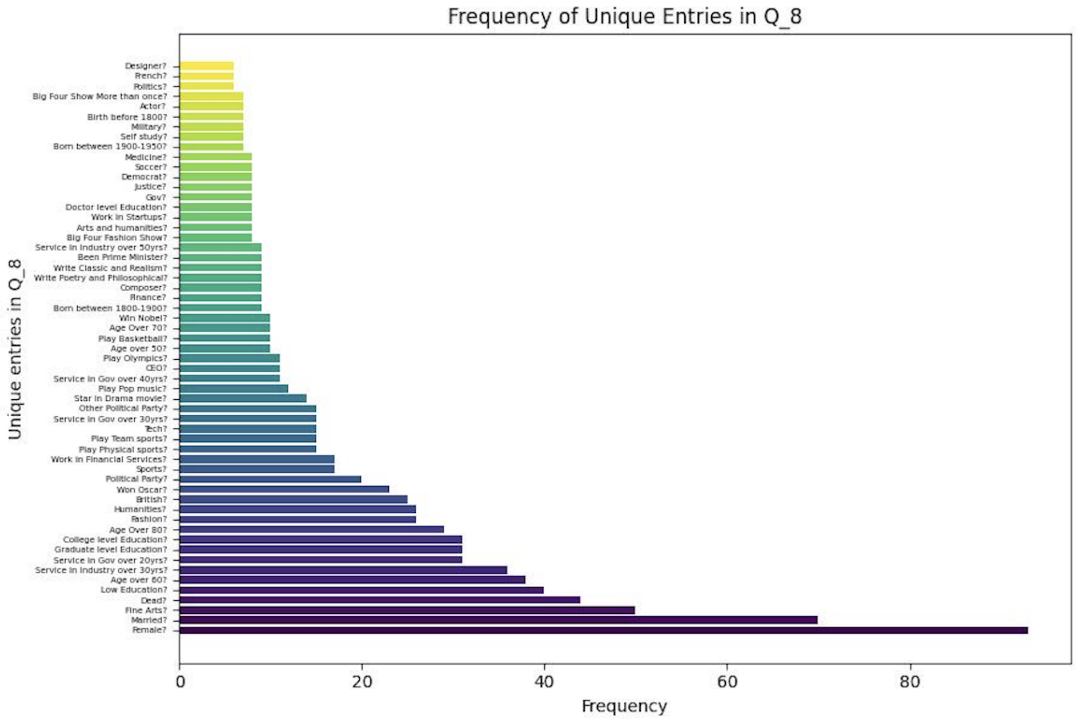

Figure 15: Distribution of selected attributes for best question at step 8

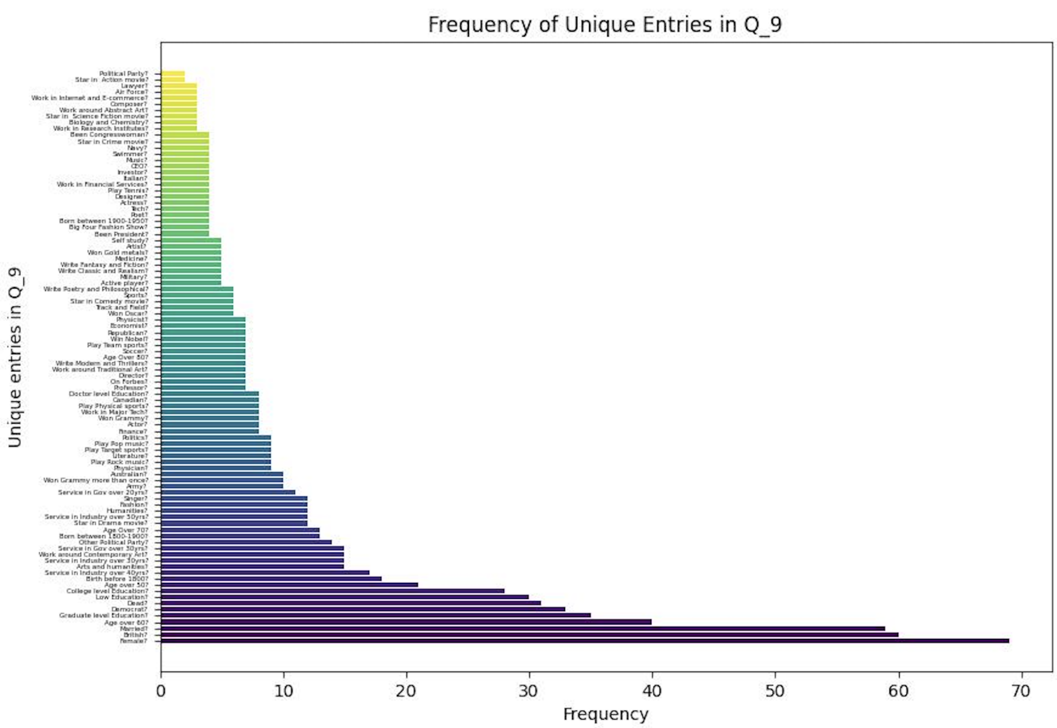

Figure 16: Distribution of selected attributes for best question at step 9

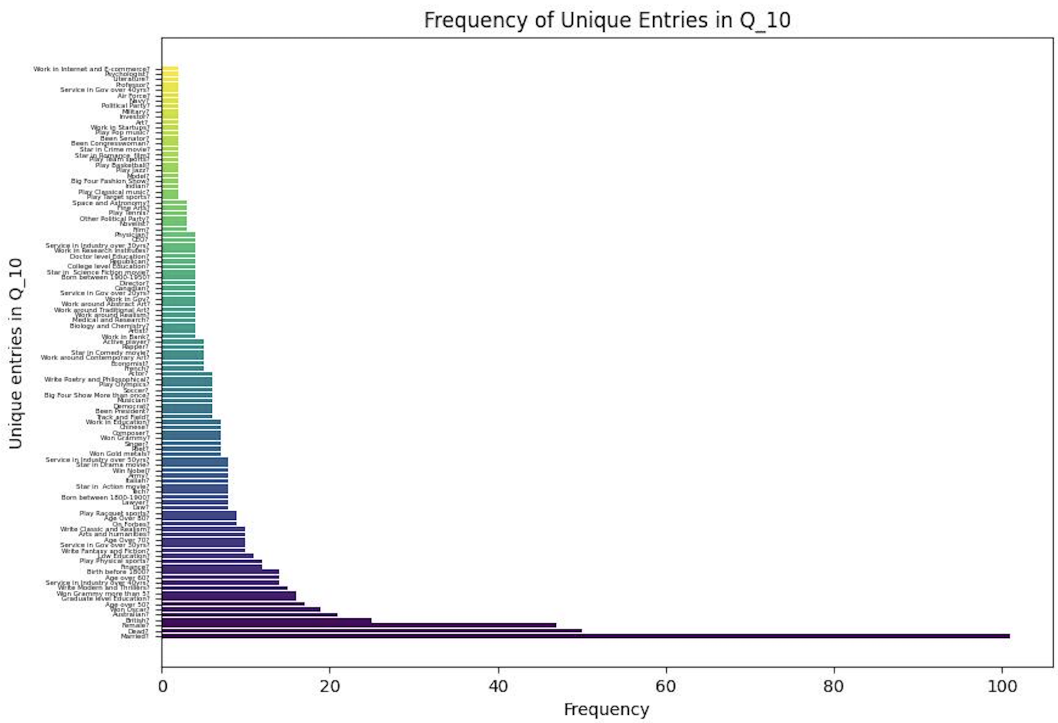

Figure 17: Distribution of selected attributes for best question at step 10

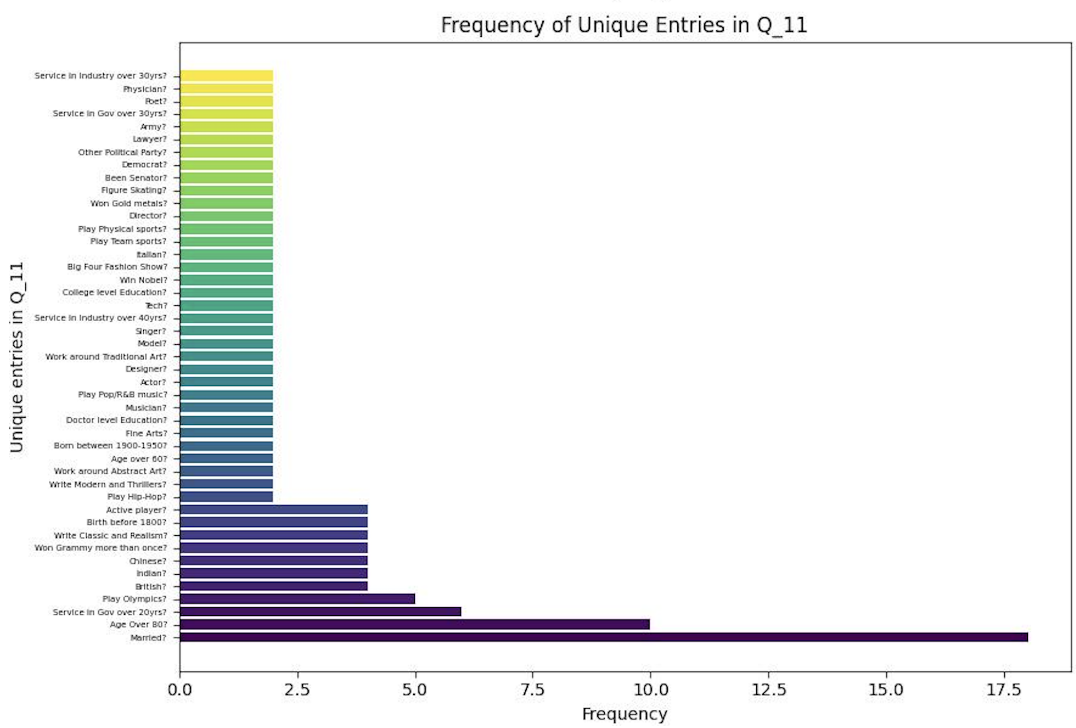

Figure 18: Distribution of selected attributes for best question at step 11

Finally, we recorded the general distribution of the number of questions that each name used and plotted in histogram. [Figure 19, Figure 20, Figure 21]

	Mean	Median	Std
Num of Q	9.9459	10	0.06054

Figure 19: General information of number of questions

Num of Questions	Frequency
8	14
9	160
10	707
11	133

Figure 20: Table: number of questions vs. Frequency

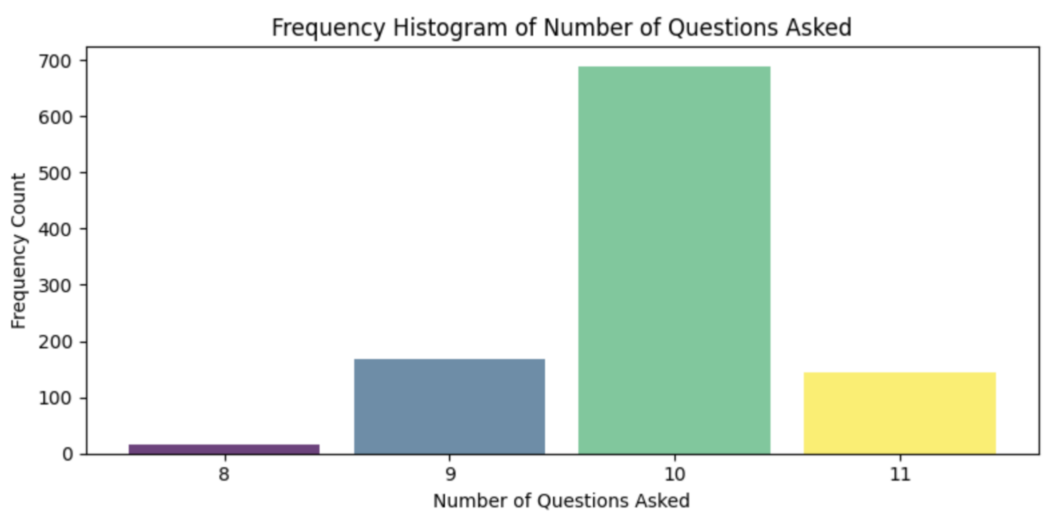

Figure 21: Histogram: number of questions vs. Frequency

For about about **67%** of population, it takes 10 questions to identify the names; and for about **16%**, it takes 9 or 11 questions; for very few cases(about **1.6%**), it takes only 8 questions. Over **84%** population uses fewer than **10** questions to get to the names.

With all the results above, we develop an online webpage for playing the game interactively, allowing players to play against each other. It is created through an interaction with python notebook and the `flask` package. The website: 20 Question Game. [Figure 22]

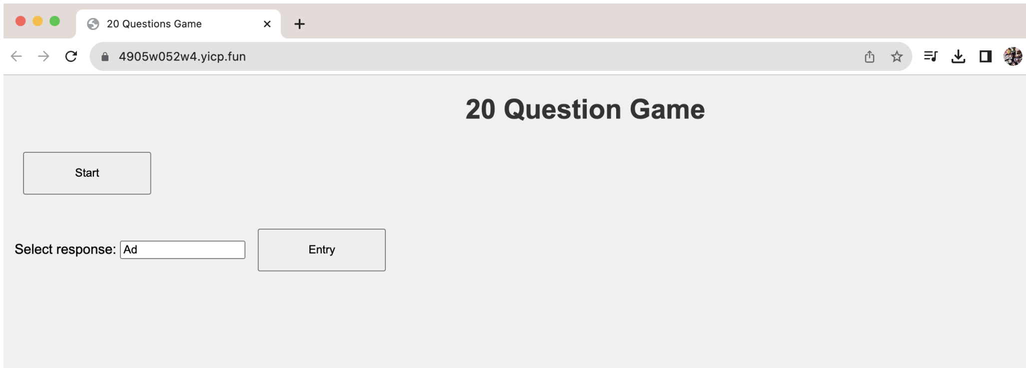

Figure 22: Online 20 Question Game

It works by first clicking `Start` to initialize the game. Then select the chosen name through the drop-down menu of `Select Response`. Names are ranked with alphabetical order and allows associative inputs. After the selection, hit `Entry` to visulize the narrowing down process and results.

5 Conclusion

The implementation of the Information Pursuit (IP) Algorithm in our search process ensures that our strategy is both efficient and dynamic. The algorithm continuously updates the population of potential answers based on previous responses, and selects the next question or attribute accordingly. The most effective questions are those that can divide the remaining population into two nearly equal parts. Examples of such questions are the questions "Born after 1950?" and "Female?" at the first step.

Despite our efforts to randomize the data generation process, biases are still present. This is partly because GPT, the model used for generating data, relies on heuristic approaches and may not fully capture cultural biases. As a result, in certain categories, the majority of names might share similar backgrounds. For instance, in the politics group, most people are Americans. Consequently, our search algorithm may inadvertently leverage these biases, affecting the results.

In this project, we assumed a uniform distribution across all names. For future research, we could consider assigning different weights to individuals based on their popularity, as indicated by online search frequencies. Additionally, we could explore the use of generative neural network models, such as Variational Autoencoders (VAEs), in the process of identifying optimal questions. This would allow us to compare the results with our current findings and potentially enhance the algorithm's effectiveness and fairness.

6 Acknowledgment

Huge thanks to Prof. Geman for guiding the project and providing support during the entire Capstone experience. Thanks also to Aditya Chattopadhyay for providing help with codes and concepts, and to An Wang for providing suggestions on data collections.

References

[1] Chattopadhyay, A., Chan, K. H. R., Haeffele, B. D., Geman, D., and Vidal, R. (2023a). Variational information pursuit for interpretable predictions.

[2] Chattopadhyay, A., Slocum, S., Haeffele, B. D., Vidal, R., and Geman, D. (2023b). Interpretable by design: Learning predictors by composing interpretable queries. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(6):7430–7443.