Boğaziçi University

Department of Management Information Systems

MIS 463 Decision Support Systems for Business

PROJECT FINAL REPORT

Just Pick It!

Project Team No: 5

M. Melih Doğan

Ezgi Dönger

Necdet Gülsever

Şevval Beyza Sezer

Metehan Uçan

Bihter Yalçın

FALL 2021

CONTENTS

1. INTRODUCTION

1.1 Decision Environment

Before the COVID-19 pandemic, the distribution channel of movies is limited. The movies were used to be watched at the movie theater or they were seen on television. Recently, especially with the COVID-19 pandemic, movies and content are mostly consumed on online platforms and independent from the environment. According to Full Circle Research Co., 70 percent of people prefer to watch movies at home. However, there are some downsides to watching a movie at home. According to another study, an average of 24 minutes is spent thinking about which movie to choose. Thus, we want to launch a platform to provide the most efficient recommendation before watching movies online at any streaming service.

Because of its complexity, the decision-making environment is uncertain. Users spend a lot of time looking for the right movie depending on their interests because there are so many possibilities on different platforms. Aside from that, movies have a variety of characteristics to consider. These characteristics, on the other hand, are not just used to draw an analogy between different movies. Our system will provide a more certain environment for decision-makers by filtering and prioritizing the movie list by decision variables which are an overview of the movie that is the summary of major plot points and main characters, key words, director, genre, spoken language, runtime, and IMDB rating. The goal of our system is to provide a suggestion platform that helps users to select movies according to their preferences.

Decision-makers are the ones who try to select movies which they might like. The decision period highly depends on users. The frequency of the decision changes according to the needs of users. For example, if you are a cinephile probably your decision period is on daily basis.

The main bottleneck of the decision is that people are unsure about the aspects of movies they appreciate. They may not know which exact feature attracts them to like the movie. Also, the preferences of people change over time. On the other hand, there is a huge amount of data while considering movie history. The main constraint of our system is the lack of accessibility. Because we only use a few different providers, the movie list is limited. The decision-making is complex because of the above-mentioned reasons.

The significance of the problem:

- **Cost of an erroneous decision:** Watching a movie is a spare time activity to enjoy for most people. Due to the requirement of having pleasure, people feel pressured while deciding how to spend their free time. As a result, if they are dissatisfied with the decision, they may feel stressed. People lose time and damage their morale when an erroneous decision occurs.
- **The current level of goal attainment in the existing system:** Although streaming services provide movie recommendations for their own platform, they have a limited movie list to give an accurate recommendation. The current level of goal attainment is not sufficient because the perfect movie recommendation might be on a different platform.

- **Sensitivity of the decisive goal to certain input parameters and the related risks and the necessity for a computerized information system for decision support:** Our system is designed to find the best and accurate movies according to users' preferences. We ask users to add some movies according to their preferences and then make them elaborate on these preferences by ranking some attributes related to the movie. Due to this fact, the input parameters of users are crucial for our system to work. Since our only source to generate these recommendations are users' conscious choices, the output is highly affected, thus very sensitive to user inputs. Thus, we need a computerized decision support system because of the complexity of the system.

1.2 Aim of Project

With the emergence of streaming services, the consumption of movies is fundamentally altered. In 2021, 78 percent of people in the United States have a subscription to streaming services, up more than 25% in only five years. (Stoll, 2021) Furthermore, this change is accelerated with covid-19. For example, between 2018 and 2020, there was a shift in preferences for seeing new movies for the first time. In 2018, 28 percent of consumers highly prefer seeing a movie for the first time in a theater, whereas in 2020 36 percent highly prefer streaming services. (Navarro, 2021) This shift causes a new problem which in a wide variety of content, people have to choose what to watch. The project aims to suggest people accurate movies according to their preferences to decrease time consumption on selecting what to watch.

1.3 Scope of Project

The scope of the project can be divided into three main categories:

1- **Functional scope:** The functionality of the project is to output a series of movie recommendations and their corresponding platform links. To achieve this output, the system needs two separate inputs. The first input is the movies they like in general or the movies that they want recommendations of, in other words, movies that are not necessarily similar but likable in the same context, most probably by the same person. The second input is the degree to which the user cares about the properties of the movie. These properties will be runtime, IMDB rate, or similar aspects. Properties will be shown in a slider view and the user will be able to select what properties they care about more.

2- **Technical scope:** The project will be available only on web platforms and will be usable for anyone with a stable internet connection. The interface will be coded on pure HTML5, CSS & JS. There will be a rest API that will supply the movies as a database. The movie recommendation calculation will be done with Python and the script will be connected with an API.

3- **Content scope:** The movie database as the main content will be limited to the movies available on popular platforms of the US and Turkey. If the movie is only and only available in a language other than English or the title of the movie is in another alphabet other than Latin, the movie will also be not available. Other contents will be scattered across the website. There will be a 'how it works tab' that includes content about the background of the system and an 'about us' tab which has information about group members.

1.4 Methodology

First of all, we fetch our movie dataset from an up-to-date movie API. As expected, there were null values, outliers, and inaccurate data, so we cleaned them. Therefore, our dataset became more accurate. After the data cleansing, we interview a Film Reviewer/Editor to better understand what people pay attention to when choosing a movie. According to the interview we prepared survey questions. We conducted a survey and analyzed the results. After an extensive literature review, we chose the content-based filtering approach as the best way to build our recommendation system. According to data, we gathered from the interview and survey we decided to go along with some attributes and eliminate some of the weak ones. Content-based filtering is a general approach that has many methodologies that can create different algorithms. The method we are planning to use is a space vector model. It estimates the similarities between word-based items which is fitting to our survey and interview data. In addition to content-based filtering, we opt to apply the analytic hierarchy process (AHP). AHP is wise to use. In our project, the AHP approach is used to optimize the selection process based on user preferences. We will gather a cluster of initial movie recommendations after content-based filtering. But since we don't want our recommendations to be bound only to word-based methodologies and want our users to have their own impact on what is actually important to them, we will use AHP to sort the data one more time. Thus, we will create more relatable recommendations with user preferences.

2. LITERATURE SURVEY

With the explosive growth in the usage of the Internet, the available digital information increases. This creates a potential problem of information overload. People must face many alternative courses of action which complicate the choosing process. (Isinkaye, Folajimi, & Ojokoh, 2015) Recommendation systems solve this problem by offering personalized suggestions according to the preferences and interests of the user. (Vidiyala, 2020)

2.1 Sources of the Recommendation Systems

Recommendation systems continually gather various data to make their recommendations. A recommendation system has 3 components as a data source. These are an item, user, and transaction. (Ricci, Rokach, & Shapira, 2015)

“Item” is the general term for what the system offers users. (Burke, 2007) A recommendation system, under normal conditions, focuses on that specific item. In our case, this kind of specific item is more likely a movie or a director. The ultimate goal of a recommendation system is to provide useful and effective recommendations tailored to that specific item, using its constituent components such as its algorithm or even its user interface.

The component of a recommendation system, other than an item, is a “user”. Generally, users provide data to the system by making choices, giving feedback, or just clicking ignore button. In a short, the user decides what item will be used by the recommendation system. Hence, we can say that recommendation is based on the interaction between item and user in the past. (Aggarwal, 2016)

If the system saves or records that kind of interaction to provide more accurate suggestions, a “transaction” can be classified as the third source of the recommendation system. The most popular example of a transaction is ratings and these ratings data can be collected in different ways by the system itself. (Schafer, Frankowski, Herlocker, & Sen, 2007)

2.2 Functions of the Recommendation Systems

There should be many reasons why businesses want to develop a recommendation system and use it for their customers. Even though there are more and more complex motivations using the systems, some significant reasons could be listed: increased the number of items sold, increased variety of range of the item sold, more positive user satisfaction, more user loyalty, and better understandıng of what the user desires for, exactly. (Ricci, Rokach, & Shapira, 2015)

2.3 Types the Recommendation Systems

There are several types of recommender systems. (Isinkaye, Folajimi, & Ojokoh, 2015)

2.3.1 Collaborative Filtering

With the growth of the Internet, the amount of information created has grown rapidly. Among all this information pollution, people started to lose a lot of time dealing with data. In other words, an overwhelming amount of data made the decision-making process more difficult. Collaborative Filtering was made to address this problem. (Mason, n.d.)

The first collaborating system was developed by Xerox PARC in 1992. The system is called Tapestry, it allows users to like or dislike and leave reviews on the electronic documents they are reading. Based on the information that was collected by the users, the system decided the relevance between users and documents. In 1995, MIT developed artificial intelligence that can make decisions by collecting information about music tastes from users. In the mid-ninety-nines Yahoo was the first company to take a CF approach to the Internet, except for systems that address specific points. Thanks to CF, they have made searching on the Internet much easier for users. (Mason, n.d.) Today CF is used to build an effective recommendation system that is preferred by very famous and high-paid companies such as Amazon, Facebook, Twitter, LinkedIn. (Apáthy, n.d.)

Collaborative Filtering collects user/item rating or preferences data and finds similarities between items/users to provide item recommendations. There are two types of CF approach: data user-based recommender and item-based recommender.

2.3.2 User-Based Recommender Systems

This approach focuses on similar user behaviors to recommend the best possible item. For Example, User 1 purchased products A and B, User 2 purchased products B and C. In this context they both purchased product B so the CF algorithm says that based on the similarity between user 1 and user 2 purchasing behaviors, user 1 is more likely to purchase product C and user 2 will purchase product A in the future. (Schafer, Frankowski, Herlocker, & Sen, 2007)

2.3.3 Item-based Recommender Systems

Item-based recommendation systems collect information about items rated by users and look at the similarity between the target product and those products. This algorithm tries to find and recommend the item most similar to the items rated by users. (Sarwar, Karypis, Konstan, & Reidl, 2001) It was developed by Amazon in 1998. This algorithm, which is behind the phrase of Amazon’s “who bought this, has also brought this” brings great success in Amazon sales. (Qutbuddin, 2020) One of the reasons why item-based recommendation systems are preferred can be shown as the fact that much more rational results can be achieved compared to user-based systems. Since user behaviors may be unexpected, looking for items that are more stable and rational compared to human behaviors will provide us with more accurate information. (Xue, et al., 2019)

2.3.4 Content-Based Filtering

Content-based systems use the features of the items in their recommendations along with user ratings. The history of content-based filtering dates back to the 1960s. The immature content-based filtering concept was called "Selective Dissemination of Information'' at that time. (Lops, Jannach, Musto, Bogers, & Koolen, 2019) We can simply explain the content-based systems step by step as follows; firstly, the items are grouped according to the keywords they have, what the users like or dislike is checked, the items are found according to the user preferences, and finally, the recommendation process is performed. (Mishra, 2021)

2.3.5 Hybrid System

Hybrid recommendation systems combine collaborative and content-based filtering approaches to make recommendations. These systems perform collaborative and content-based predictions separately, then integrate the results of both strategies to produce recommendations. (Isinkaye, Folajimi, & Ojokoh, 2015)

2.3.6 Knowledge-Based

Unlike others, knowledge-based systems evaluate what the user wants within the framework of existing rules. It can be very useful in situations where there is no regular data flow from the users. (Jannach, Zanker, Felfernig, & Friedrich, 2010) In knowledge-based recommendation systems, we can say that there is no transaction component since they do not use past interactions. (Aggarwal, 2016)

2.3.7 Demographic

Demographic systems make suggestions based on the user's demographic preferences such as age, sex, or country. The main assumption is that people who have similar demographic backgrounds also have similar preferences. (Bobadilla, Ortega, Hernando, & Gutiérrez, 2013)

2.4 Analytic Hierarchy Process

Analytic hierarchy process (AHP) is commonly used when there are multiple criteria to prioritize and select. While offering the best option, AHP uses the concept called pairwise comparisons which is comparing two criteria at once instead of multiples. AHP evaluates the outcomes of each pairwise comparison. Every criterion is assigned a weighting based on its importance. The more weight a criterion has, the more essential it is to the ultimate choice. Furthermore, AHP can provide the ranking and how these alternatives are accurate according to the finest choice thanks to this evaluation procedure.

2.5 Factors That Affect People's Decision-Making

People often go through decision-making processes in their daily lives. Many scientific theories, especially in the cognitive psychology area, have been put forward to explain how people make their decisions and which factors affect the change in their decision-making processes. (Dietrich, 2010)

In the experiments conducted by Juliusson, Karlsson, and Gärling (2005), it was determined that past losses and gains, that is, past experiences, have a serious effect on people's decision-making. In this manner, decision-making is directly related to people's emotional states. (Naqvi, Shiv, & Bechara, 2006)

Another study states that people's cognitive levels and biases have a serious impact on their decision-making processes, especially when making rational decisions. (Stanovich & West, 2008) A series of studies on people from different demographic backgrounds are conducted. The study has shown that socioeconomic status, cognitive ability, age, and other personal differences are related to the way people make decisions. (Bruine de Bruin, Parker, & Fischhoff, 2007)

In another study, the concept of "belief in personal relevance" is encountered. If people believe that the issue they are deciding about is important regardless of what the others think, they are much more likely to decide it. (Acevedo & Krueger, 2004)

2.6 Interview Results

We also had an interview with the movie critic and asked him some questions about effective movie prediction.

Levent is an accountant who grew up in Kadıköy and is passionate about theatre and film. During his high school years, he played in various theatres and was the head of the cinema club at the university. Currently, she takes part in the organization of various events at the Mithat Alam Film Show Center and is also the editor of Altyazı. We asked questions about the movie selection process and the indicators which are crucial while determining the movie. Our expert said that the key indicators in the film determination process are the production year, color, time, cast, director, and language. He also added that the environment is an important factor before the pandemic. Now, every movie can be watched at home. We also asked him if he would use it if there was a DSS application that can help while find the best movie. He indicated that searching the movie is the most time-consuming part of the process and using this kind of DSS is more beneficial and time-efficient, also it would be better for anyone who wants to improve themselves in the field of language. Thus, while learning a new language, people want to increase their familiarity with this language by watching similar movies or TV series.

The interview's original transcription could be found in the Appendix A section of this report.

2.7 Survey Study

For the sake of reaching the highest possible surveyors, we decided to ask the least number of questions possible, to do that we considered the interview results. Keep the survey compact and get answers to the questions we seek to develop our model. The result of this approach was positive, we reached 284 total surveyors from different demographic backgrounds who enjoy watching movies.

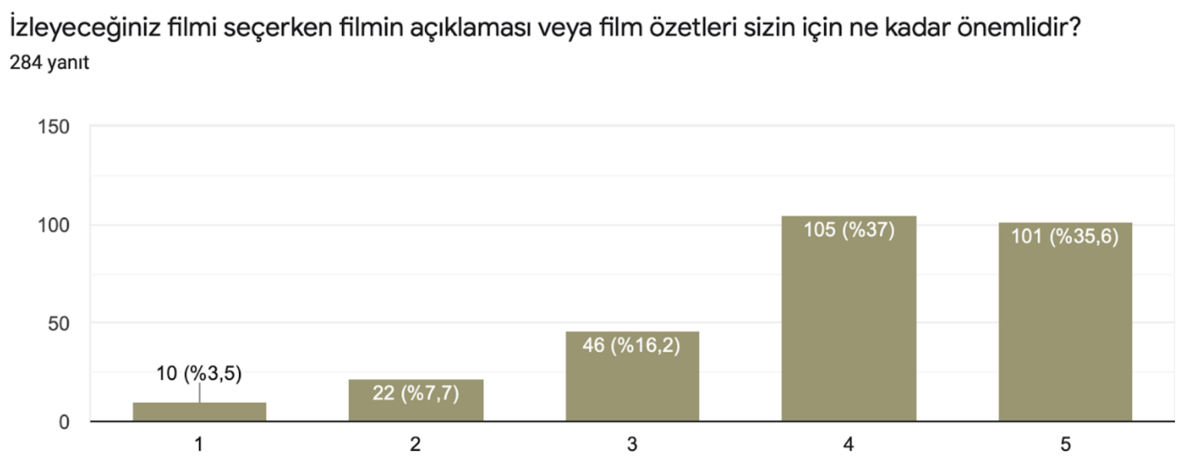

Figure 1 - While selecting a movie to watch, how important are the movie overviews or explanations? (1- Not important at all, 5- Crucially important)

To test our hypothesis of content-based filtering is one of the best approaches while making a movie recommendation system. Since content-based filtering uses overview to make predictions, we asked how much the viewers care about the overview while selecting a movie. Figure 1 shows that 72,6% of the viewers find it important.

Figure 2 - While selecting a movie to watch, how important is the genre? (1- Not important at all, 5- Crucially important)

As seen in figure 2 genre is considered an important factor for movie selection. Only 1.1% of the viewers do not consider genre as a differentiator. This attribute will have a strong impact on our system.

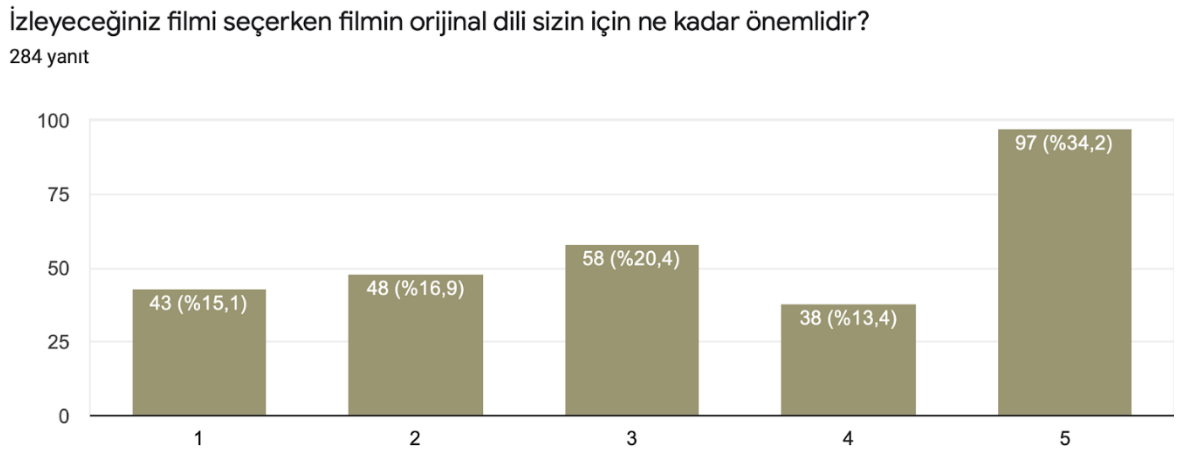

Figure 3 - While selecting a movie to watch, how important is the original language? (1- Not important at all, 5- Crucially important)

Figure 3 also shows us that the majority of the people consider the original language while selecting the movie. We thought of adding this but did not at the final product because the most majority of movies are already in English. It did not make sense to put this attribute into our model because it did not differentiate the recommendations.

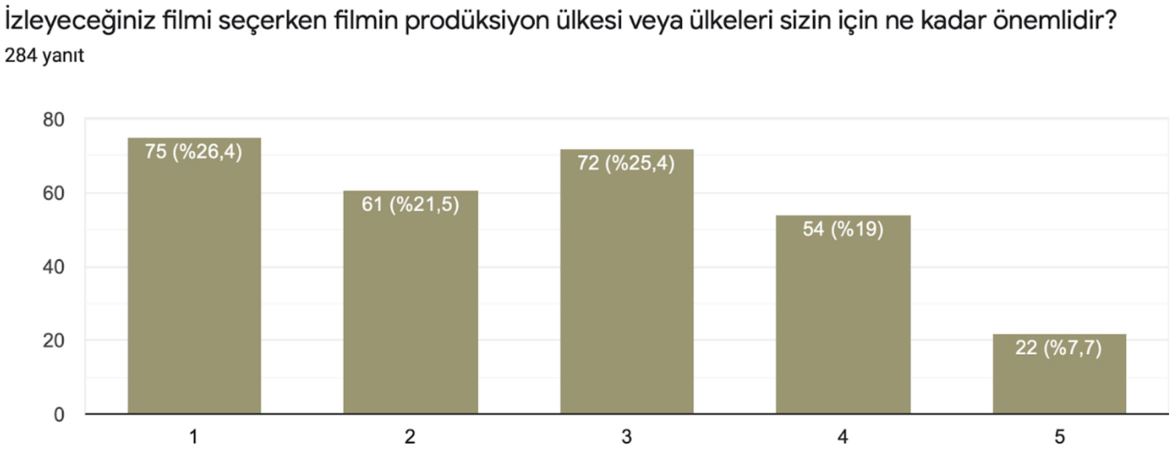

Figure 4 - While selecting a movie to watch, how important is the production country or countries? (1- Not important at all, 5- Crucially important)

We asked viewers if they would consider a production country a criterion for their selection. As seen from figure 4, 47,9% of the viewers do not and 25,4% of the viewers have an indifferent opinion. Since we want to keep the weak criteria out to make our system strong, we will not add this criterion to our system.

Figure 5 - While selecting a movie to watch, how important is the release year? (1- Not important at all, 5- Crucially important)

As Figure 5 shows, the release year is also on the lower side of the importance scale. Only 23.9% of people find it important, therefore we will not be using this criterion.

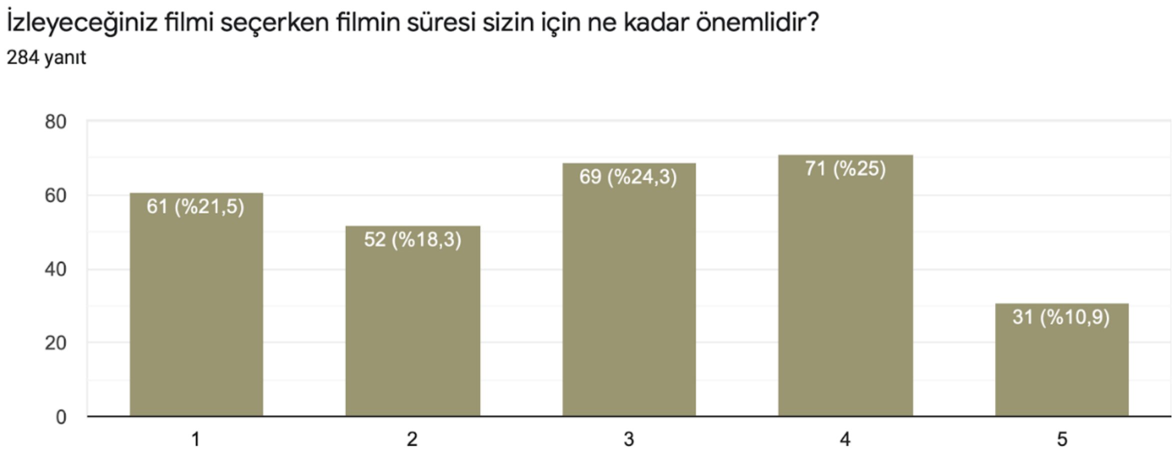

Figure 6 - While selecting a movie to watch, how important is the runtime (length of the movie)? (1- Not important at all, 5- Crucially important)

The runtime of the movie has 39,8% viewers on the lower side and 35,9% viewers on the higher importance scale. 24,3% of viewers have indifferent opinions. Since upper and lower levels have close values, we will consider this criterion as a weak on the weak side but we will put it to our model.

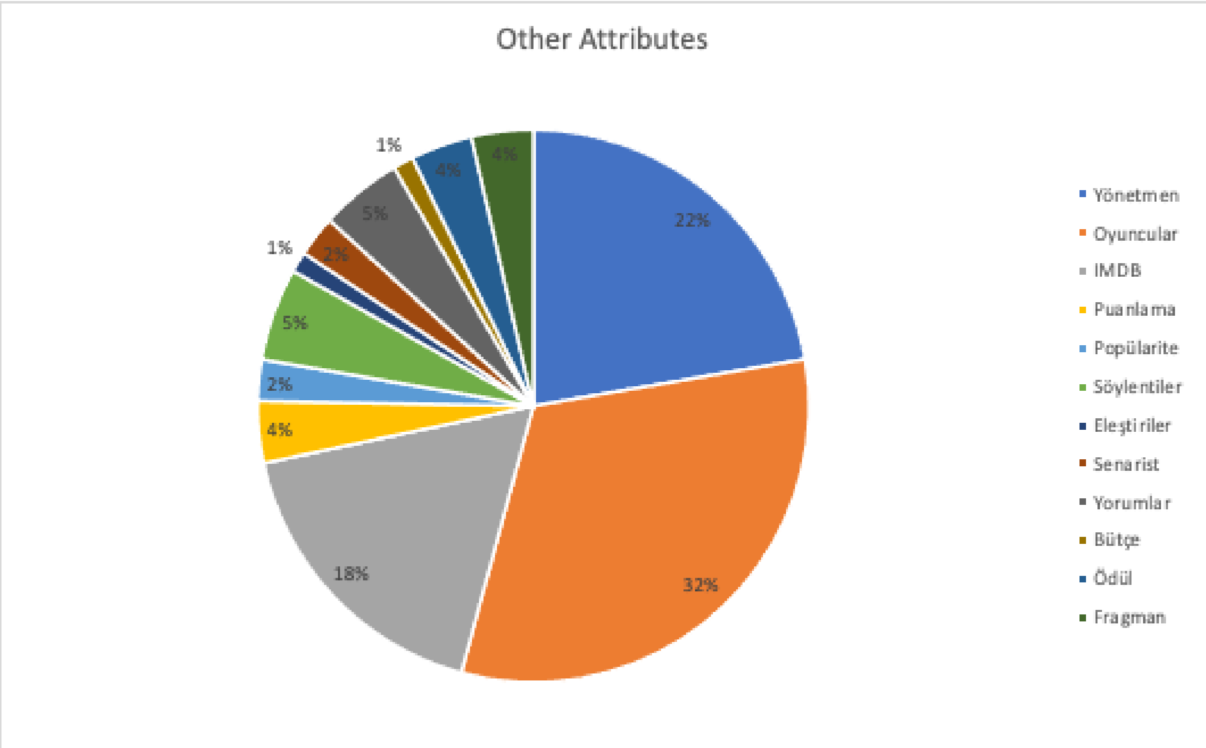

Figure 7 - What are the other attributes that you consider while selecting a movie?

We also asked viewers if there are any attributes they would consider valuable and out of 89 responses found out that most of the viewers would like to select their movies based on the director and the cast. After that IMDB rating comes. Our dataset did not contain cast information. After seeing how important the cast, rating, and the director is we added these to our dataset and our system because of the high demand.

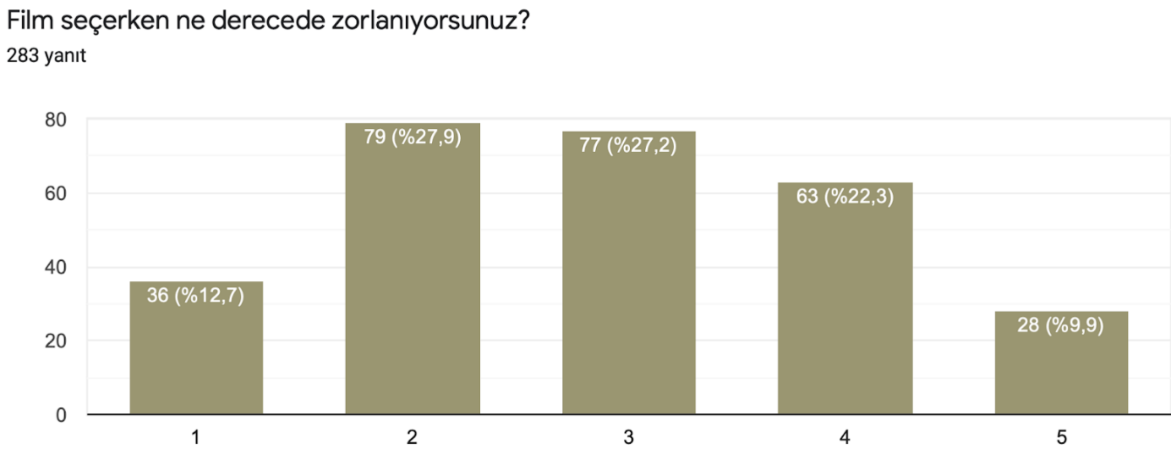

Figure 8 - How hard is it for you to select a movie? (1- Not hard at all, 5 - Very hard)

We asked viewers about their experience while selecting a movie to watch. We found out that 40,9% of the people are having a hard time, 27,2% of people are having a somewhat hard time and 32,2% of people do not have a hard time.

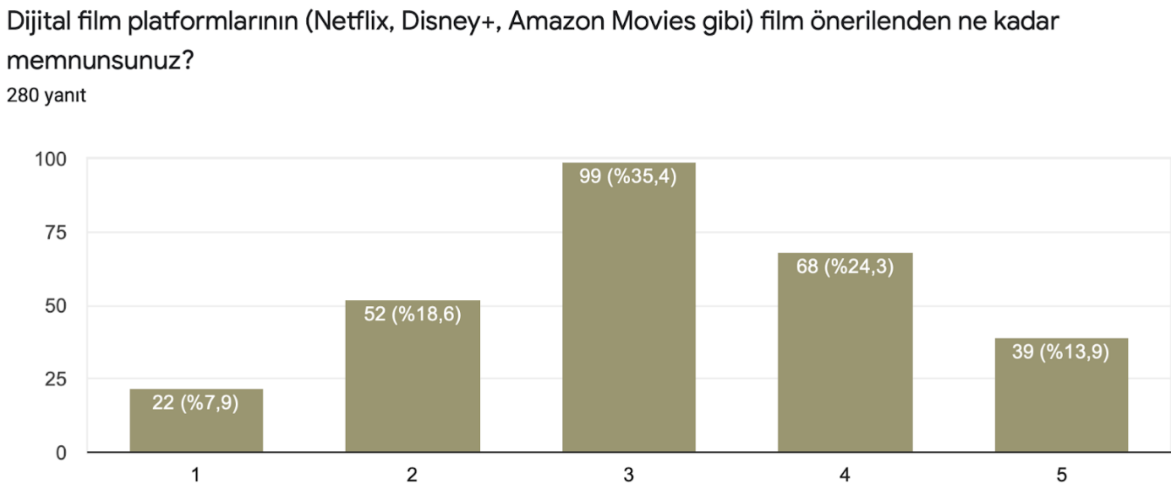

Figure 9 - How happy are you with the digital film platforms' film recommendations? (1- Not happy at all, 5- Very happy)

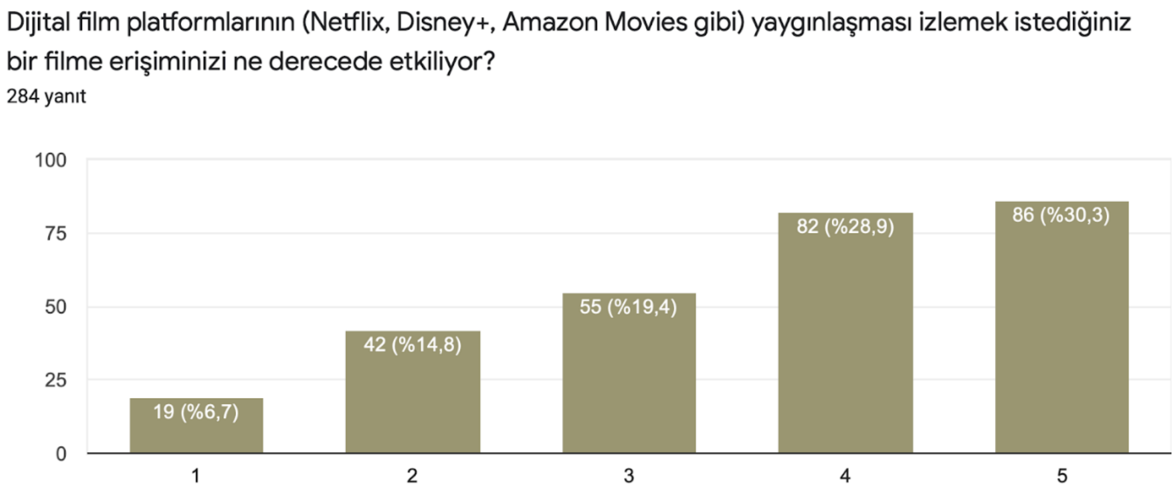

Figure 10 - To what extent does the proliferation of digital film platforms affect your access to a movie you want to watch? (1 – Less , 5 - More)

Figure 10 was asked to confirm our hypothesis on the pain point. It is harder to select a movie when there are many platforms with different movie content. Our system is aiming to bring a solution to this by sharing the platforms.

Figure 11 - Do you use a system that gives you the movies you might like, considering the movies you enjoy watching and the features mentioned above, with information on which platforms they are available on? (Yes is blue – No is red)

Lastly, Figure 11 asks about our system idea in general. If people would want to use the system we created. This question is crucial because it validates our projects' existence reason with 86,6%.

2.7.1 Additional Survey

Figure 7 gave us critical information about the users' preferences, therefore we wanted to make sure the info given in the open questions matter so we conducted an additional survey.

Figure 12 - While selecting a movie to watch, how important is the cast? (1- Not important at all, 5- Crucially important)

Figure 13 - While selecting a movie to watch, how important is the director? (1- Not important at all, 5- Crucially important)

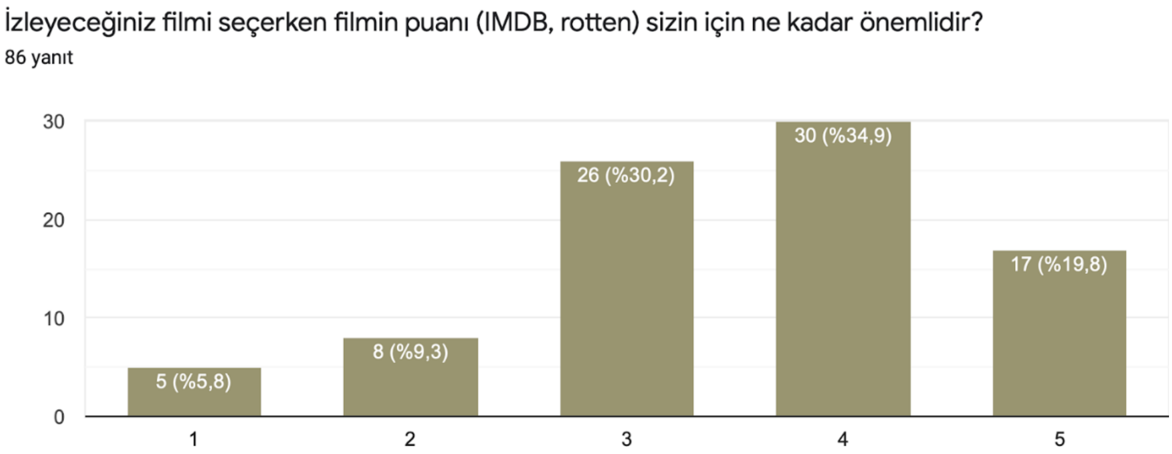

Figure 14 - While selecting a movie to watch, how important is the rating (IMDB, Rotten tomatoes)? (1- Not important at all, 5- Crucially important)

3. INPUT DATA ANALYSIS

Data Manipulation Process

1- Added providers column, added cast & director
2- We removed the movies that do not have 'RELEASED' status because we want to recommend movies that are available to watch. We reduced rows from 478.857 to 475.796.
3- We removed movies that had a runtime of 0. Thus, the number of rows decreased from 475.796 to 376.730.
4- We deleted movies that have less than 50 characters. Number of rows decreased from 376.730 to 368.396.
5- We filtered the movies that are available in America & Turkey because we want them to be available in this region. Thus, the number of rows decreased from 368.396 to 135.147.
6- Finally, we checked if there were any null or empty variables in other dimensions, there were none.

Data Visualization

We analyzed our data to have a deeper understanding of our dataset. We used Tableau and python to visualize our data. We had to use the Atlas MongoDB BI connector to convert unstructured JSON data to structured relational data.

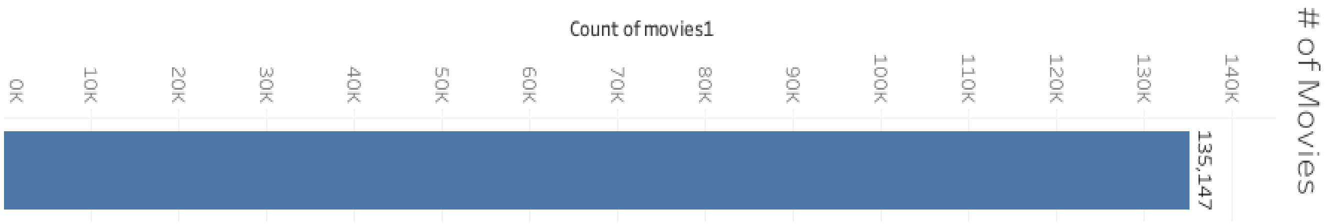

Figure 15 - Number of movies, which are our rows.

Since we base the first part of our system on words with content-based filtering, we wanted to visualize our data with words. The best way to do this is a word cloud, the higher the number they appear in the rows the bigger they look.

The first word cloud is from the cast. The biggest word is Frank Welker. Welker is an American voice actor, he has contributed over 860 films. He has voiced many characters such as Scooby Doo and Garfield himself. He is followed by more publicly known actors. Samuel L. Jackson, Morgan Freeman, Danny Trejo, Bess Flowers, and Bruce Wills.

Figure 16 - Cast

The second word cloud is the overview. These words are used to describe the movie. This criterion will have a huge impact on our system. For our system, we will not use the stopwords such as the, to, and, it. They have no value while describing the contents of the movie. Therefore, in this word cloud, we also wanted to exclude them. The most appearing words are the world, life, find, family, take, and one.

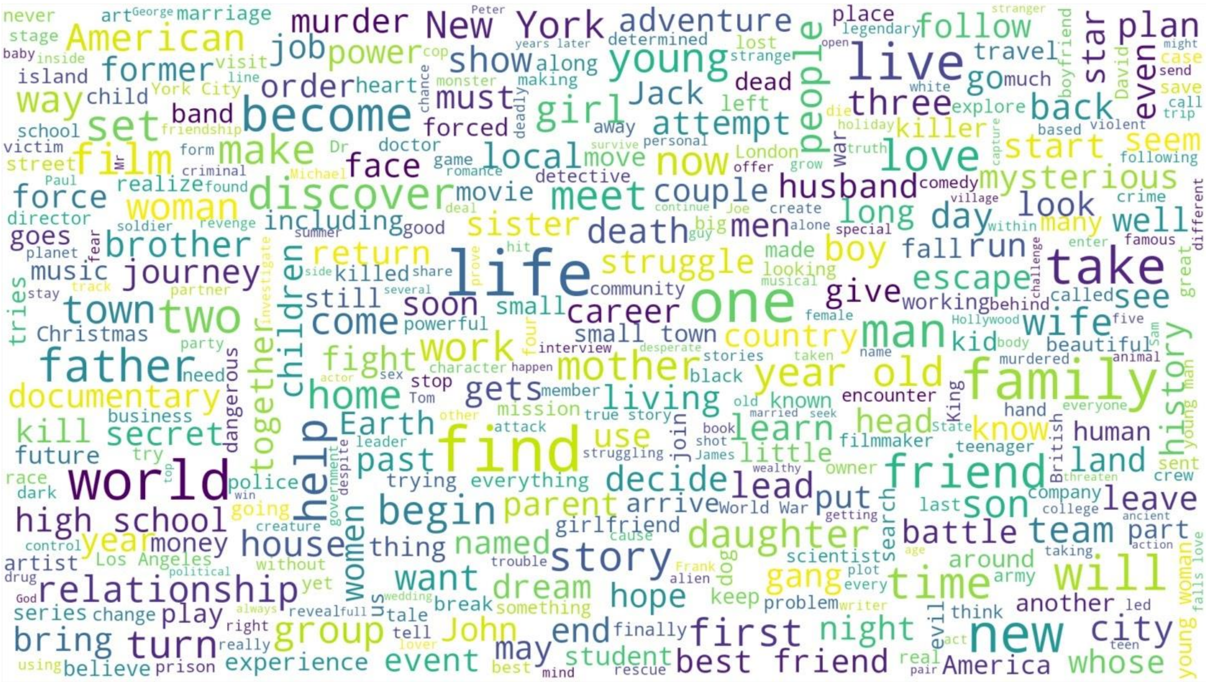

Figure 17 - Overview

The third one is the keywords. A keyword is a word or concept which has high importance. They are used frequently by viewers while searching for a movie on digital platforms. The most frequent keyword appearing in our movie set is woman director, followed by murder and based on novel or book.

Figure 18 - Keywords

The last one of the word clouds is directors. The most apparent names are Sam Newfield and Dave Fleisher. Sam Newfield is an American director who has over 250 movies between 1923-1958. Dave Fleischer is also a silent era (1920-40) director who has his studio and directed some iconic animations such as Betty Boop, Popeye the Sailor, Gulliver's Travels (1939), and Superman (1941).

Figure 19 - Directors

Our movie dataset has many genres available and it is an important factor in movie selection decisions. With this visualization, we can observe that most of the movies are in the drama genre. Note that one movie can have more than one genre. Drama is followed by comedy, thriller, and action.

Figure 20 - Genres

We also wanted to see how popular different genres are. Even though most of the movies had the drama genre, it is not the most popular among users. Adventure and animation are the most popular ones.

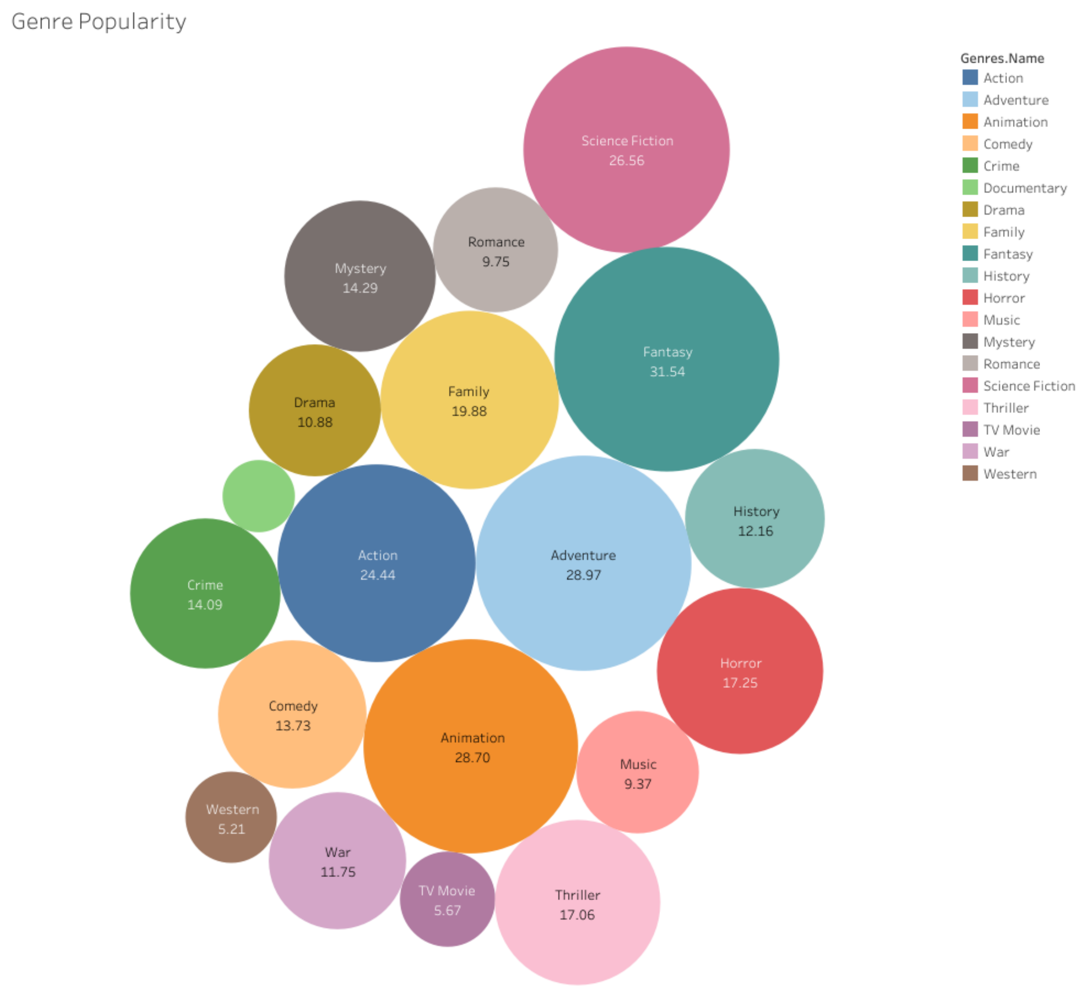

Figure 21 – Genre Popularity

Our main pain point is the increasing number of movie streaming platforms. We visualized this data with a tree map. There were 118 different platforms in the US and 16 platforms available in Turkey.

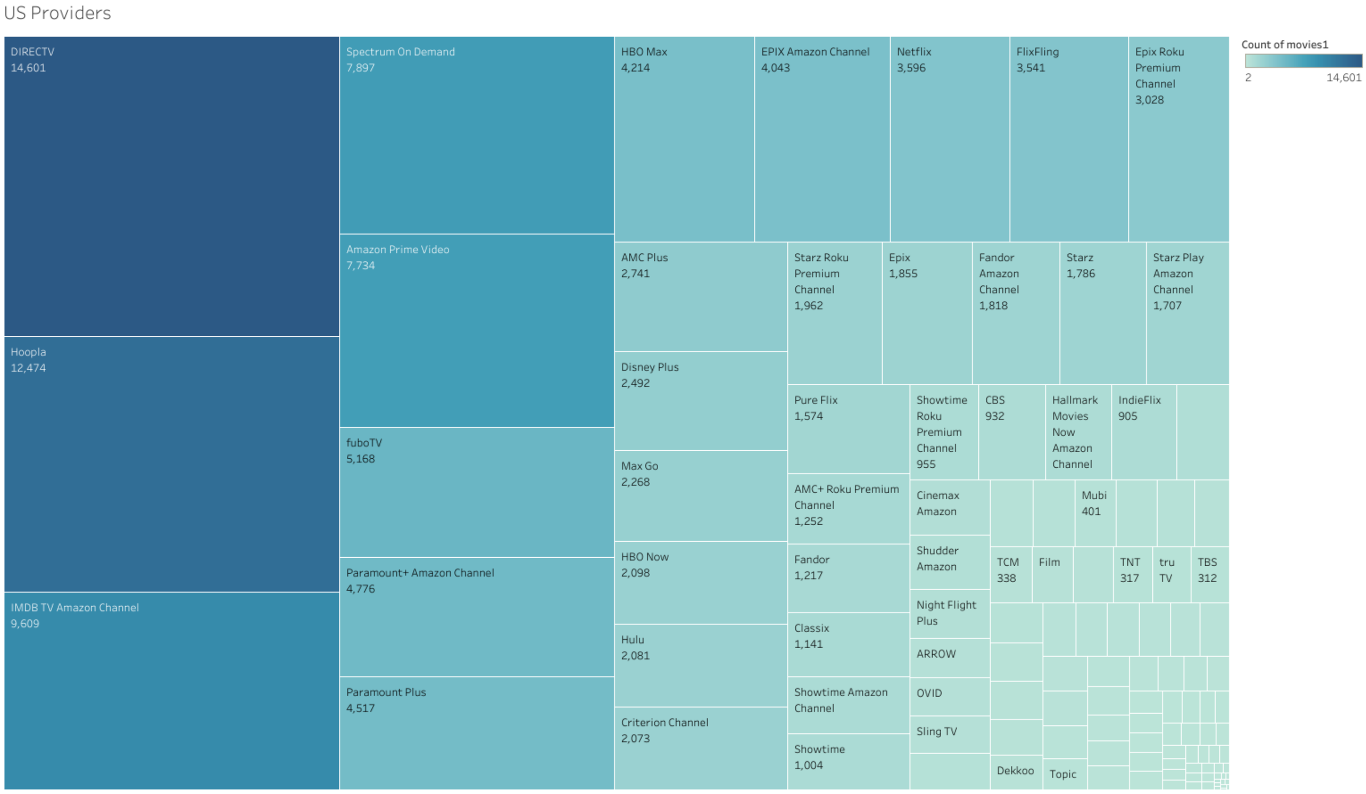

Figure 22 - US providers tree map

DIRECTV has the lead in the US. Netflix is in the 10th order.

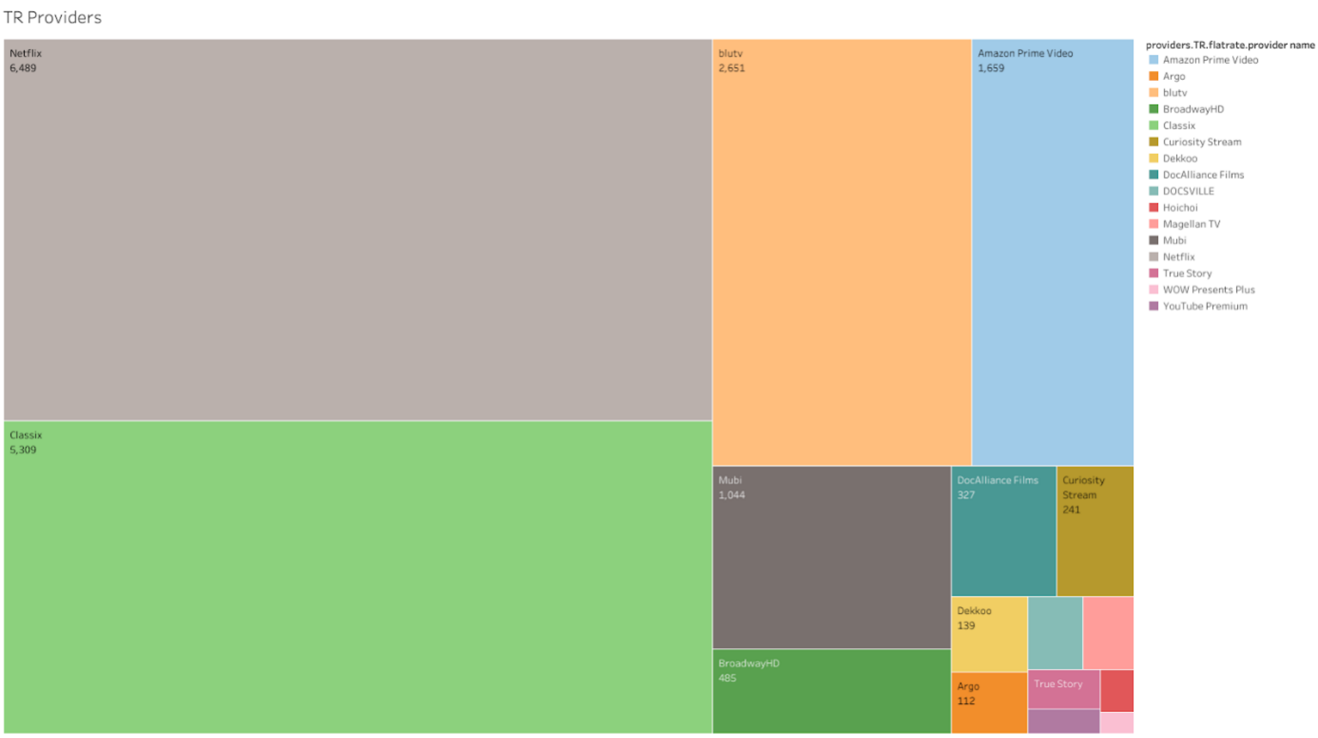

Figure 23 - Turkey tree map

Netflix has the lead in Turkey, followed by Classix, Blutv & Amazon Prime Videos

To check if our runtime has outliers, we visualized this data to a box & whisker graph.

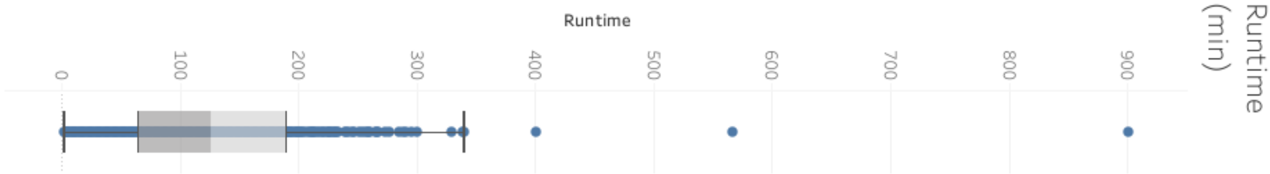

Figure 24 - Box & Whisker Graph

The original language of the movie is highly English. This is expected because we took the movies that are only available in the US and Turkey.

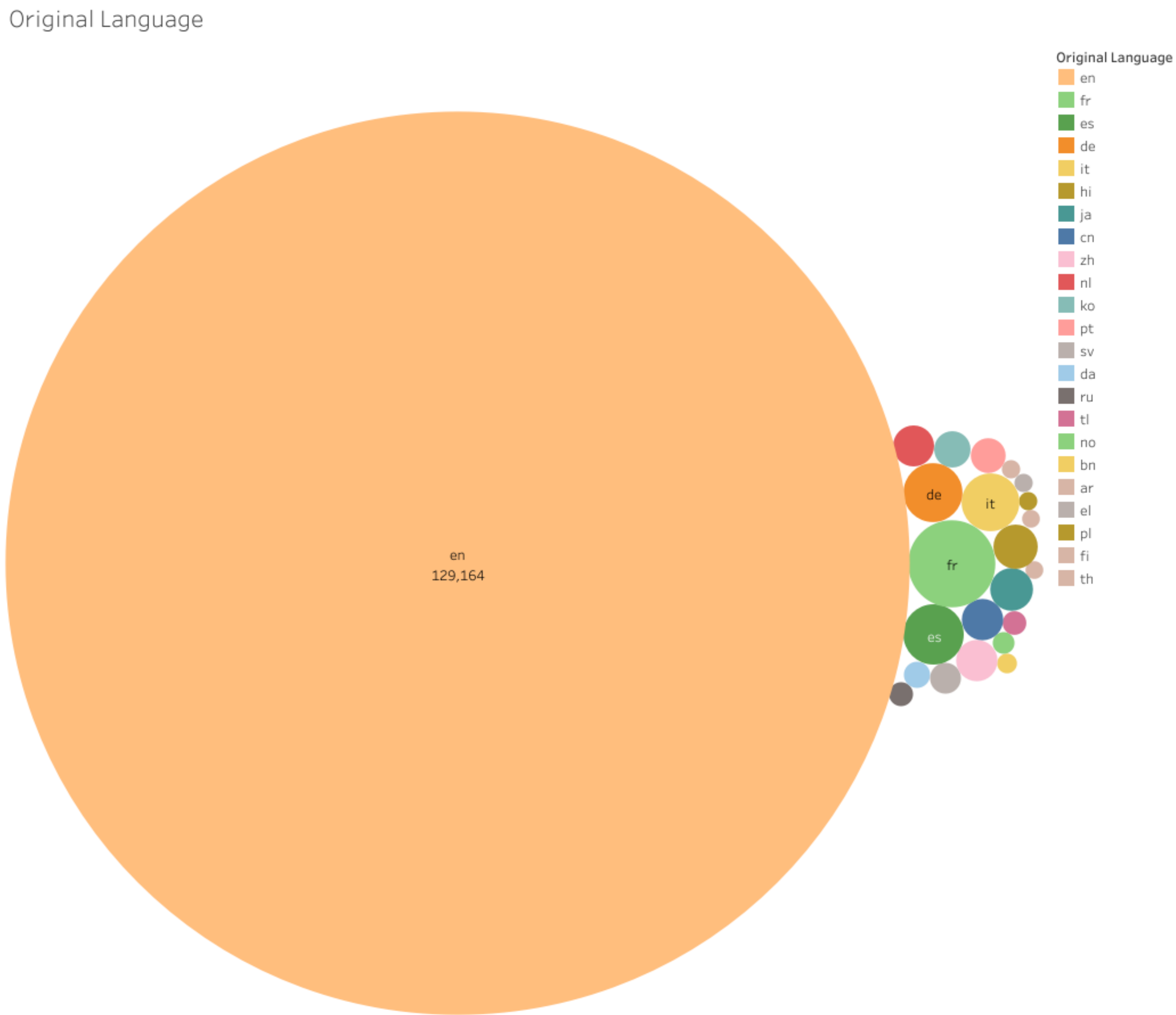

Figure 25 – Languages

We analyzed how the genre affects the runtime of the movie. History was the genre with the highest average runtime, followed by war and action. Animation is on the lower side of the runtime average.

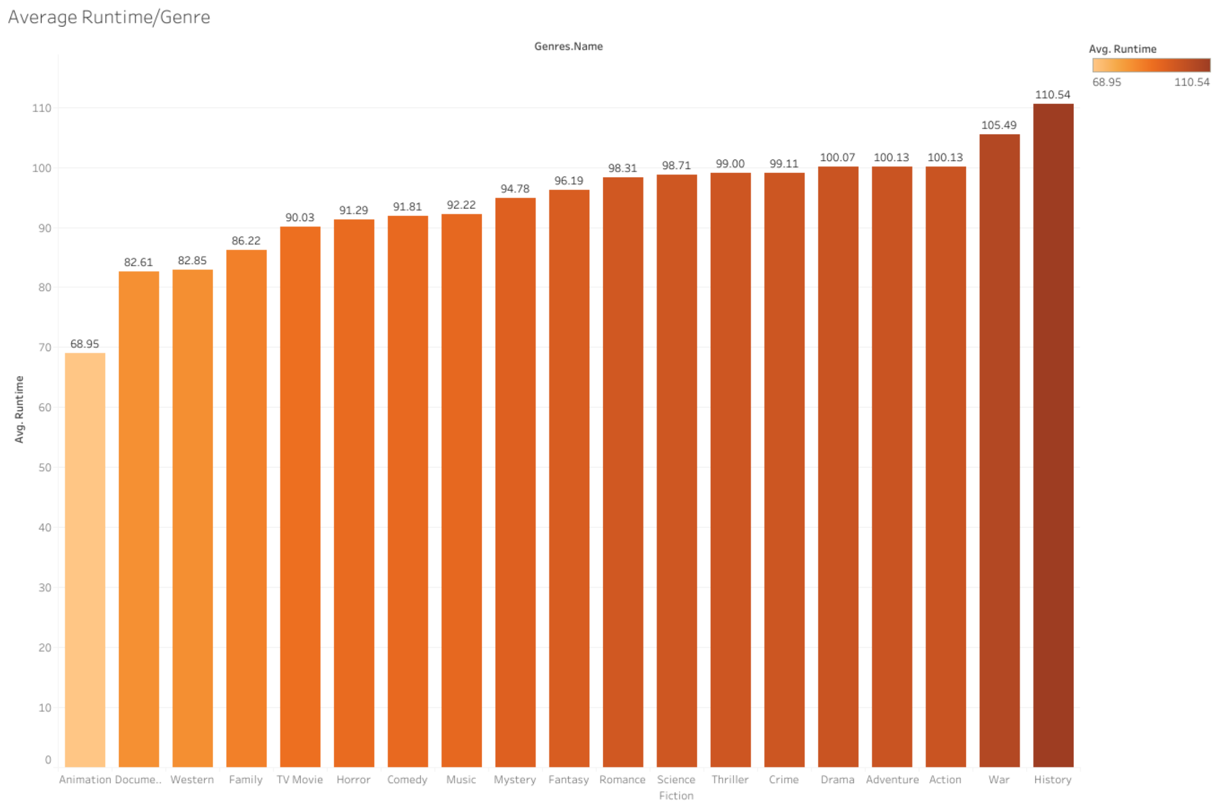

Figure 26 – Average Runtime/Genre

We wondered about which release year has the highest average revenue. The year 1997 has the most average revenue followed by 2001.

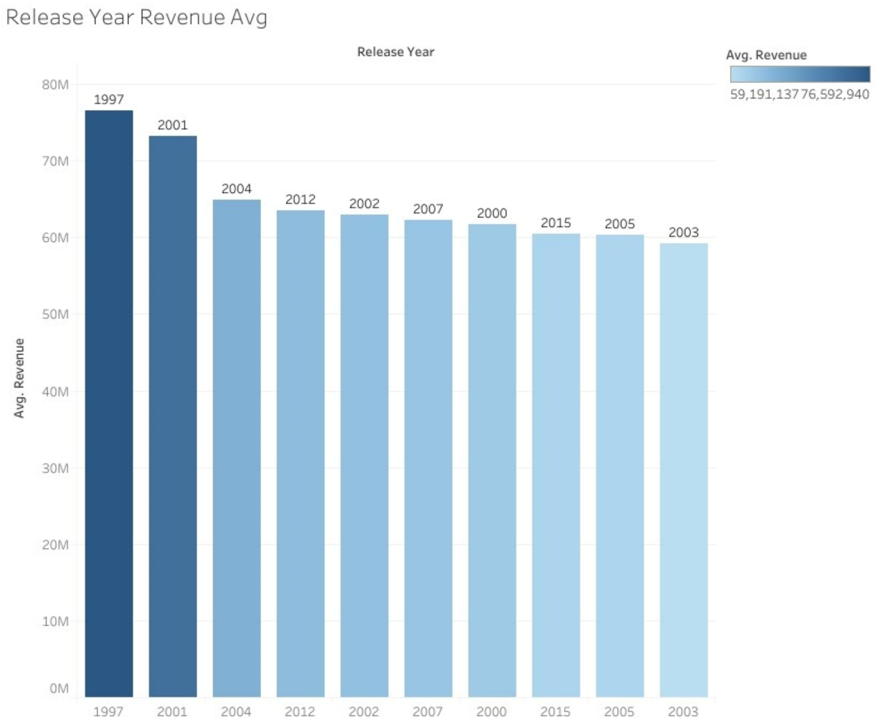

Figure 27 – Release Year Revenue Average

Release year number of the graph of the movie shows us that the year 1997 has the most number of movies released.

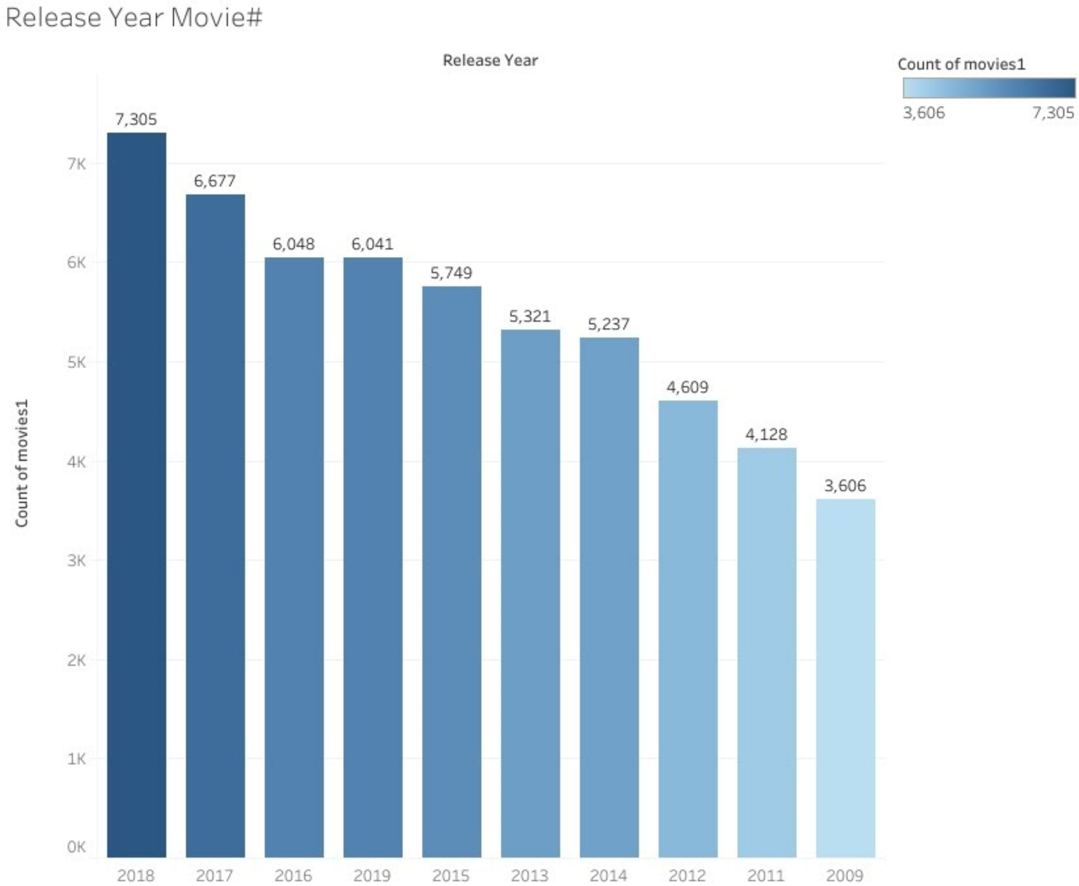

Figure 28 - Release year number of the movies

Lastly, we see that our dataset has 2.03% short films and 97.7% non-short films. The popular movie platforms have little to no short films.

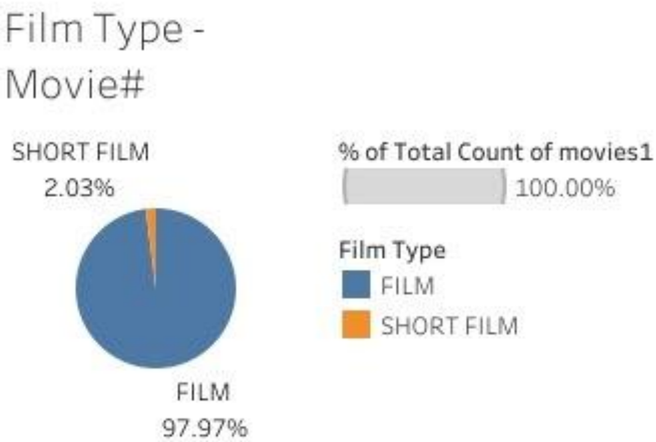

Figure 29

4. DEVELOPMENT OF THE DECISION MODEL

We aimed to offer the most accurate movie recommendations to our users according to their movie preferences. To achieve this, we used machine learning algorithm in our problem. The machine learning algorithm we chose was Content-based Filtering, which is also frequently used in recommendation systems. Basically Content-based Filtering is a Machine Learning technique that uses similarities in features to make decisions. Using the content-based approach in our own data, we determined our attributes that we looked at for similarities.

4.1. Data selection and preparation

We concatenated **overview, title, genres, cast** and **director** into **description** column for each movie.

```
df['description'] = df['overview'] + df['title'] + df['genres'].fillna('').str.join(' ')  +
df['cast'].fillna('').str.join(' ') + df['crew'].apply(lambda d: d if isinstance(d, list) else
[]).apply(lambda x: list(filter(lambda a: a['job'] == 'Director', x))).apply(lambda d:
list(map(lambda c: c['name'],d))).str.join(' ')
```

While doing this we replaced every null items in those columns with “ ” via fillna() function. Secondly we used str.join() function to join them.
Also we create and use tokenizer function to,

- Remove non-alpha characters
- Handle lower-case
- Split words
- Clear stop words
- Remove common words
- Handle stem words

```
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
import re
stemmer = PorterStemmer()
def stem_words(words_list, stemmer):
  return [stemmer.stem(word) for word in words_list]

def tokenizer(text):
  letters_only = re.sub("[^a-zA-Z]",' ', text)
  words = letters_only.lower().split()
  stopwords_eng = set(stopwords.words("english"))
  useless_words = ['film','movie','available','theater','directed']
  stopwords_eng.update(useless_words)
  useful_words = [x for x in words if not x in stopwords_eng]
  useful_words_string = ' '.join(useful_words)
  tokens = nltk.word_tokenize(useful_words_string)
  stems = stem_words(tokens, stemmer)
  return stems
```

In order to best extract the similarity between the selected films, which is our main goal, we examined which algorithms are used in the filtering systems.

As a result of our research, we decided to apply cosine similarity, a simple method that gives strong connection and similarity between two vectors, to our model.

To briefly explain cosine similarity, it is a metric that measures how similar two items are. Similarity values range from 0 to 1. If the cosine similarity value between two vectors is close to 0, it is considered slightly similar, if it is close to 1, it is considered very similar. (Javed, 2020)

(O'Reilly Media, Inc., n.d.)

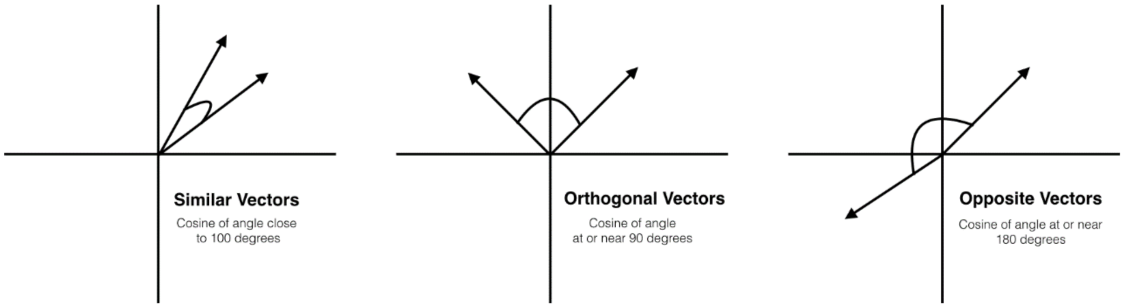

Figure 30

We decided to implement cosine similarity to our model but in order to use this model we need vectors. We decided to use TF-IDF vectorizer to convert **description** column to vectors.

4.2. Vectorizing Data

This method will calculate how often the words in the description occur. It will give less weight to words that are repeated a lot, and more weight to words that are less repetitive. In this way, it will find the importance of the word for the whole document. For example " is", "are" words are probably occur in the most of the movies' description those words will be less weighted (Reikeras, n.d.). Except for the whole document, TF-IDF will also give weight according to the importance of the words, based on each film.

```
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf = TfidfVectorizer(
    tokenizer = tokenizer,
    stop_words = 'english'
)
tfidf_matrix = tfidf.fit_transform(df['description'])

// cosine similarity calculation
// example output: [
// [1.0, 0.2, 0.3], [0.2, 1.0, 0.4], [0.3, 0.4, 1.0]
// ]
```

4.3. Cosine Similarity

After vectorizing our data, we use the cosine similarity method we explained above to compare the movies selected by the user with all the movies in our data set and find the most similar ones.

```
from sklearn.metrics.pairwise import cosine_similarity
cosine_sim = cosine_similarity(tfidf_matrix, tfidf_matrix)

// create index for movie ids

indices = pd.Series(df.index, index=df['id']).drop_duplicates()

// package cosine similarity result with indices into a file

import pickle
with open('data.pkl', 'wb') as f:
   pickle.dump(cosine_sim, f)

with open('indices.pkl', 'wb') as f:
   pickle.dump(indices, f)
```

4.4. Criteria Tree

After cosine similarity, remaining movies are ready to be evaluated by AHP methodology. To accomplish this, we must first present the criteria to the user and obtain input on the weight of each criteria. The criteria tree is provided below; there are three criteria to consider while selecting a movie that is appropriate for the user. The duration, rating, and number of providers are the three factors.

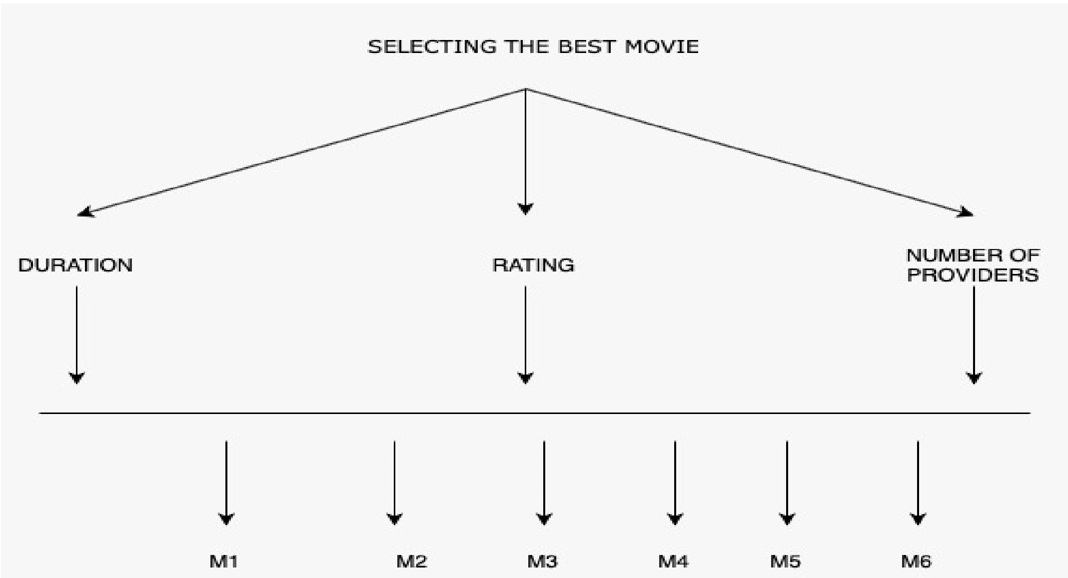

Figure 31 – Criteria Tree

4.5 AHP

A pairwise comparison of criteria is performed based on the user's input. This side of the project was coded in JavaScript and its React framework. We also did calculations on Microsoft Office Excel to give an example and clarify the code.

```
export const Rating = ({ onChange, left, right, defaultSelected }) => {
  const [selected, setSelected] = useState(defaultSelected);
  const options = [-4, -3, -2, -1, 0, 1, 2, 3, 4].map(value => ({ id: value, default: value === 0,
max: value === -4 || value === 4 }));


  const onClick = (id) => {
    onChange(id);
    setSelected(id);
  }

  return (
    <RatingContainer>
      <OptionLabel>{left}</OptionLabel>
      {options.map(option => <RatingOption key={option.id} onClick={() =>
onClick(option.id)} selected={option.id === selected} id={option.id}>{option.max ?
<MaxLabel>MAX</MaxLabel> : option.default ? <MaxLabel>0</MaxLabel> : ''}
</RatingOption>)}
      <OptionLabel>{right}</OptionLabel>
    </RatingContainer>
  );
}
    <CenterColumn>
        <StyledTitle>Which is more important?</StyledTitle>
        <StyledText>Which aspects should be closer to the movies you
choose?</StyledText>
        <Rating left={'IMDB'} right={'Runtime'} onChange={(id) => {
priorityState.imdbRuntime = id }} defaultSelected={priorityState.imdbRuntime} />
        <Rating left={'Runtime'} right={'Availability'} onChange={(id) => {
priorityState.runtimeAvailability = id }} defaultSelected={priorityState.runtimeAvailability} />
        <Rating left={'IMDB'} right={'Availability'} onChange={(id) => {
priorityState.imdbAvailability = id }} defaultSelected={priorityState.imdbAvailability} />
      </CenterColumn>
```

An example comparison as like this:

A			
Criteria	**Duration**	**Rating**	**Number of providers**
Duration	1	3	4
Rating	0.33	1	4
Number of providers	0.25	0.25	1
	1.583333333	**4.25**	**9**

Then we normalized this matrix:

```
export const normalize = (matrix) => {
  let normalized = Array(matrix.length).fill(0).map(() => Array(matrix[0].length).fill(0));
  [...Array(matrix.length).keys()].forEach(i => {
    const colsum = [...Array(matrix.length).keys()].map(j => matrix[j][i]).reduce((a, b) => a +
b);
    [...Array(matrix.length).keys()].forEach(j => {
      normalized[j][i] = matrix[j][i] / colsum;
    });
  });
  console.log("Normalized matrix:", normalized);
  return normalized;
}
```

Normalised A			
Criteria	**Duration**	**Rating**	**Number of providers**
Duration	0.631578947	0.71	0.44
Rating	0.210526316	0.24	0.44
Number of providers	0.157894737	0.06	0.11

Then we found the average of each row to find the priority vector:

```
export const createPriorityMatrix = (imdbRuntime, runtimeAvailability, imdbAvailability) => {
  let imdb = [1, priorityScore(imdbRuntime), priorityScore(imdbAvailability)];
  let runtime = [1 / priorityScore(imdbRuntime), 1, priorityScore(runtimeAvailability)];
  let availability = [1 / priorityScore(imdbAvailability), 1 / priorityScore(runtimeAvailability),
1];

  const selections = { imdbRuntime, runtimeAvailability, imdbAvailability };
```

```
    console.log("Priority selections:", selections);

    const priorityMatrix = [imdb, runtime, availability];
    console.log("Priority matrix:", priorityMatrix);
    return priorityMatrix;
}
```

Normalised A					X (Priority Vector)
Criteria	Duration	Rating	Number of providers		
Duration	0.631578947	0.71	0.44		0.59
Rating	0.210526316	0.24	0.44		0.30
Number of providers	0.157894737	0.06	0.11		0.11

After that we checked the consistency of the values. First, to find consistency, we found the lamda max. After calculations of CI and CR, we observed that this system is consistent.

```
export const extractEigenVector = (matrix) => {
    let eigenVector = matrix.map(row => row.reduce((a, b) => a + b) / matrix.length);
    console.log("Priority vector:", eigenVector);
    return eigenVector;
}

export const validate = (matrix, eigenVector) => {
    const Ax = multiply(matrix, eigenVector);
    const lambdaMax = Ax.map((x, idx) => x / eigenVector[idx]).reduce((x, y) => x + y) / 3

    const ci = (lambdaMax - 3) / 2;
    console.log("Consistency index:", ci);

    const cr = ci / 0.58;
    console.log("Consistency ratio:", cr);
    return cr < 0.1;
}
```

X	AX	lamda	RI	0.58	
0.59	1.93	3.27	n	3	
0.30	0.94	3.12	CI	0.069355989	
0.11	0.33	3.02	CR	0.119579292	< 0.1
		3.14	Consistency Check	Consistent	

Since rating and number of providers are quantitative, we directly normalized them. However, duration was calculated by the distance to the chosen movie. After normalized the ranking of alternatives, we multiplied it by the criteria weight to find the output of the AHP. Because we would like to include the cosine similarity to get an accurate result, we multiplied the AHP output by the cosine similarity score.

```
export const calcAHP = (movies, recommendations, priorityState) => {
  const { imdbRuntime, runtimeAvailability, imdbAvailability } = priorityState;
  const priorityMatrix = createPriorityMatrix(imdbRuntime, runtimeAvailability,
imdbAvailability)
  const priorityVector = extractEigenVector(normalize(priorityMatrix));

  const imdbRatings = recommendations.map(movie => movie.imdb_rating);
  let imdbMatrix = Array(imdbRatings.length).fill(0).map(() =>
Array(imdbRatings.length).fill(0));
  for (let i = 0; i < imdbRatings.length; i++) {
    for (let j = 0; j < imdbRatings.length; j++) {
      imdbMatrix[i][j] = imdbRatings[i] / imdbRatings[j];
    }
  }
  console.log("IMDB matrix:", imdbMatrix);
  const imdbPriorityVector = extractEigenVector(normalize(imdbMatrix));
  console.log("IMDB priority vector:", imdbPriorityVector);

  const idealRuntime = movies.map(movie => movie[0].runtime).reduce((x, y) => x + y) /
movies.length;
  const runtime = recommendations.map(movie => movie.runtime === idealRuntime ? 0.1 :
abs(movie.runtime - idealRuntime));

  let runtime_matrix = Array(runtime.length).fill(0).map(() => Array(runtime.length).fill(0));
  for (let i = 0; i < runtime.length; i++) {
    for (let j = 0; j < runtime.length; j++) {
      runtime_matrix[i][j] = runtime[j] / runtime[i];
    }
  }
  console.log("Runtime matrix:", runtime_matrix);
  const runtimePriorityVector = extractEigenVector(normalize(runtime_matrix));
  console.log("Runtime priority vector:", runtimePriorityVector);

  const availability = recommendations.map(movie =>
movie.us_providers?.providers?.length || 0 + movie.tr_providers?.providers?.length || 0);
  let availability_matrix = Array(availability.length).fill(0).map(() =>
Array(availability.length).fill(0));
```

```
  for (let i = 0; i < availability.length; i++) {
    for (let j = 0; j < availability.length; j++) {
      availability_matrix[i][j] = availability[i] / availability[j];
    }
  }
  console.log("Availability matrix:", availability_matrix);
  const availabilityPriorityVector = extractEigenVector(normalize(availability_matrix));
  console.log("Availability priority vector:", availabilityPriorityVector);

  const alternativesPriorityMatrix = recommendations.map((_, i) => [imdbPriorityVector[i],
runtimePriorityVector[i], availabilityPriorityVector[i]]);
  console.log("Alternatives priority matrix:", alternativesPriorityMatrix);

  const rankings = multiply(alternativesPriorityMatrix, priorityVector);
  const movieRankings = rankings.map((ranking, i) => ({ movie: recommendations[i], ranking
}));
  console.log("Movie rankings:", movieRankings);

  return movieRankings;
};
```

To make the cosine similarity more accurate, we take the mean of the movies entered. Lastly, after calculating the ranking of alternatives and the mean of cosine similarity scores, we multiply both of them with 0.5 and sum it to reach our custom sort value.

```
export const sortByWeighted = (movieRankings) => {
  return movieRankings.map((movie) => ({ ...movie, weightedRanking: movie.ranking * 0.5 +
movie.movie.score * 0.5})).sort((a, b) => b.weightedRanking - a.weightedRanking)
};
```

By doing this, we got the final results and “The Lord of the Rings: The Return of The King” is selected as the best movie for the user in terms of his or her chosen movie at the first stage, which is “The Lord of the Rings: The Fellowship of the Ring”

Chosen Movie	Duration	Rating	Number of Providers
The Lord of the Rings: The Fellowship of the Ring	179	8.8	3

Alternatives	Duration	Rating	Number of Providers
The Lord of the Rings: The Two Towers	1.00	8.7	3
The Lord of the Rings: The Return of the King	0.89	8.9	3
King Kong	0.96	7.2	7
Star Wars: Return of the Jedi	0.75	8.3	7
Saints and Soldiers	0.50	6.7	3
Peterloo	0.86	6.5	1
	4.97	46.30	24.00

Normalized Alternatives	Duration	Rating	Number of Providers		Priority Vector	Result
The Lord of the Rings: The Two Towers	0.20	0.19	0.13		0.59	0.188951
The Lord of the Rings: The Return of the King	0.18	0.19	0.13		0.30	0.177241
King Kong	0.19	0.16	0.29		0.11	0.192482
Star Wars: Return of the Jedi	0.15	0.18	0.29			0.175483
Saints and Soldiers	0.10	0.14	0.13			0.116909
Peterloo	0.17	0.14	0.04			0.148933
	1.00	1.00	1.00			

Result	Score	Final Results
0.1759707	0.461731	0.081251141
0.190276	0.43878	0.083489381
0.1893155	0.148197	0.028056025
0.1656934	0.132582	0.021967921
0.1103828	0.127648	0.014090161
0.1683616	0.122639	0.020647633

5. DEVELOPMENT OF THE DSS

5.1. DSS Architecture

"Just Pick It!" is a web-based application. Our model consists of a model base, backend, and frontend. We took our data from two sources the first one is "The Movie Database" it consists of many movies with their attributes, but it does not include the IMDB values. However, we need these values for an effective DSS because our survey results indicate that viewers care about IMDB points. Therefore, we wrote a web crawler and got the IMDB data of the related movies. We combined and cleaned these two data and put them in mongoDB. This database sends movies to two systems. One is the cosine similarity vector space. With content-based filtering, we selected the attributes and did cross cosine similarity for each movie, got the scores for the movies selected. This score feeds the model. The other one is the website, where the user selects the movie by searching them. After the selection of the movies, user also enters the importance of set of criteria, there we make a consistency check and give warnings if the result of AHP is not consistent. The selected movies and the criteria weights goes to the model base calculates the ranking of alternative through AHP, uses the cosine similarity scores, makes a custom score, and re-sorts the movies to recommend the most relevant movie. Movie recommendations are showed through website again with their links.

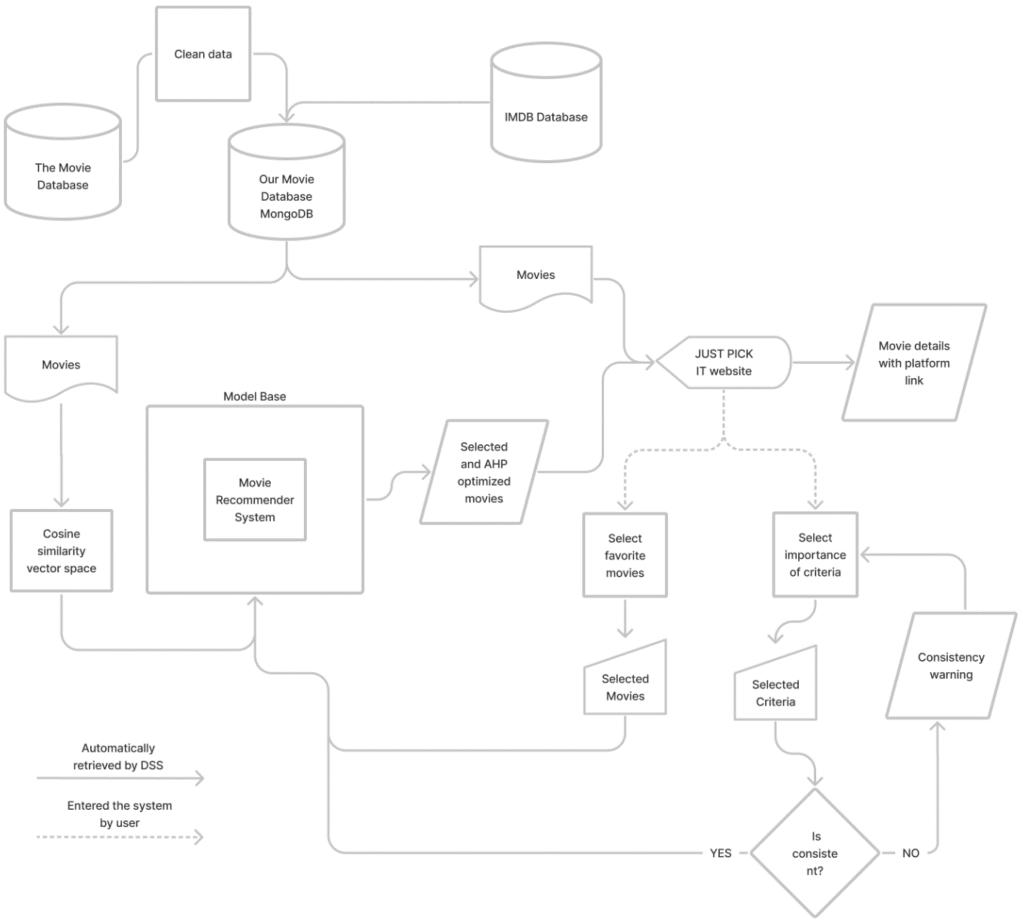

Figure 32 – DSS Model

5.2. Technical

5.2.1. Model

For our model we mostly used python. The cosine similarity calculation is written on it and stored in mongoDB. We also used two libraries to make the machine learning algorithm work. First one is pandas for easier data manipulation and the second one is skitlearn to actually make the calculation.

5.2.2. Back-end

To connect our frontend to backend we again used python. These connections were made by using the framework of FastAPI. MongoDB is feeding the movies to the website for users to select on. For deployment we used uvicorn for server implementation and docker for container.

5.2.3. Front-end

To write our frontend, we used modern frontend libraries besides HTML5, CSS and JavaScript which is the core web-application languages. We used Bootstrap for easier structuring. React and NodeJS as libraries and MongoDB for selection of movies.

5.3. User Interfaces

This is our home page where the users can go to the process with one click. If they want more info, they can use the top bar to navigate.

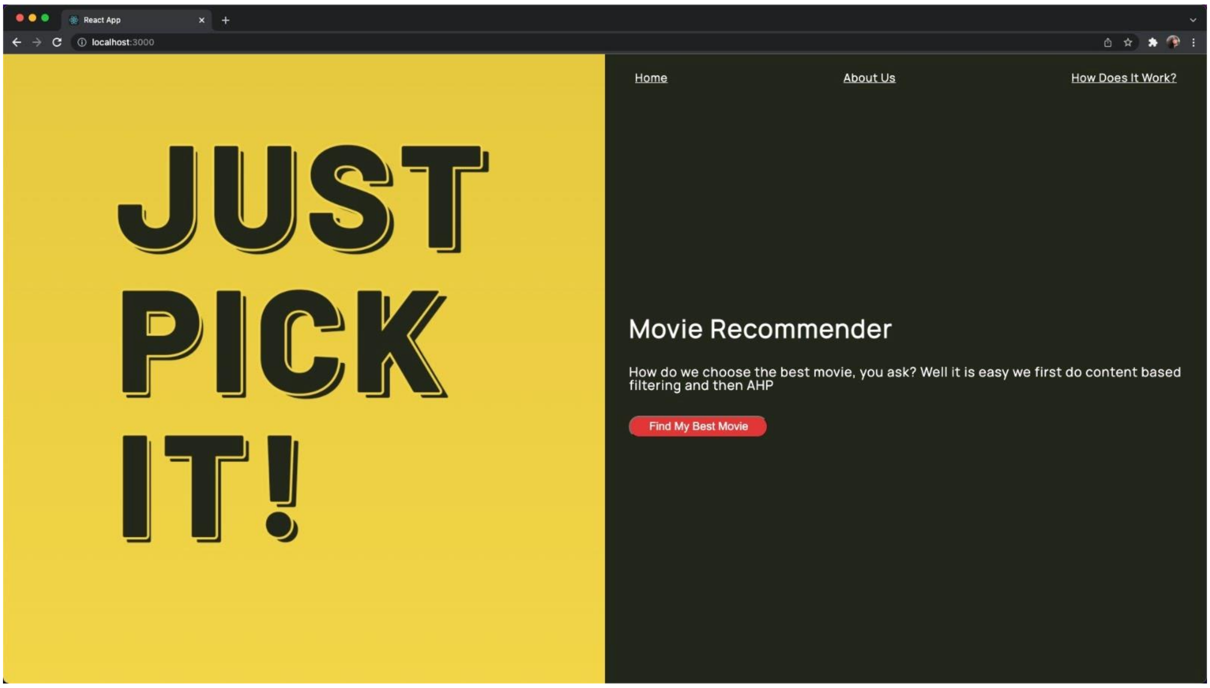

Figure 33 - Home Page

How does it work section where users can read about how the system works.

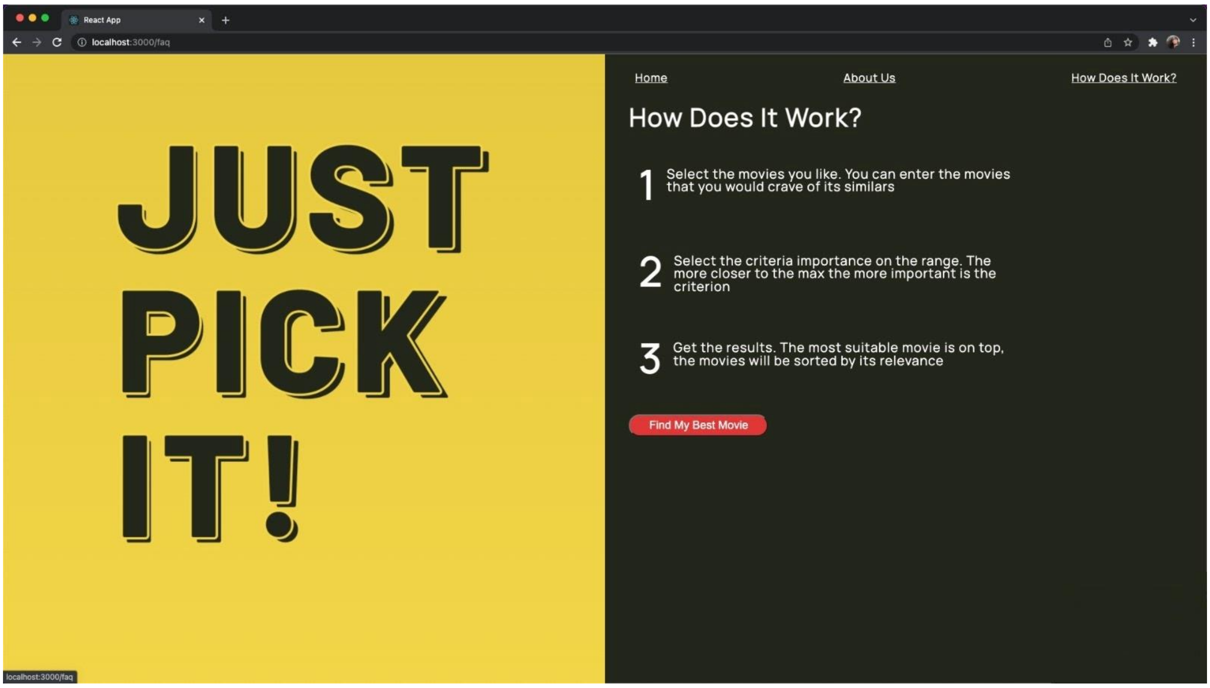

Figure 34 – "How Does It Work?" section

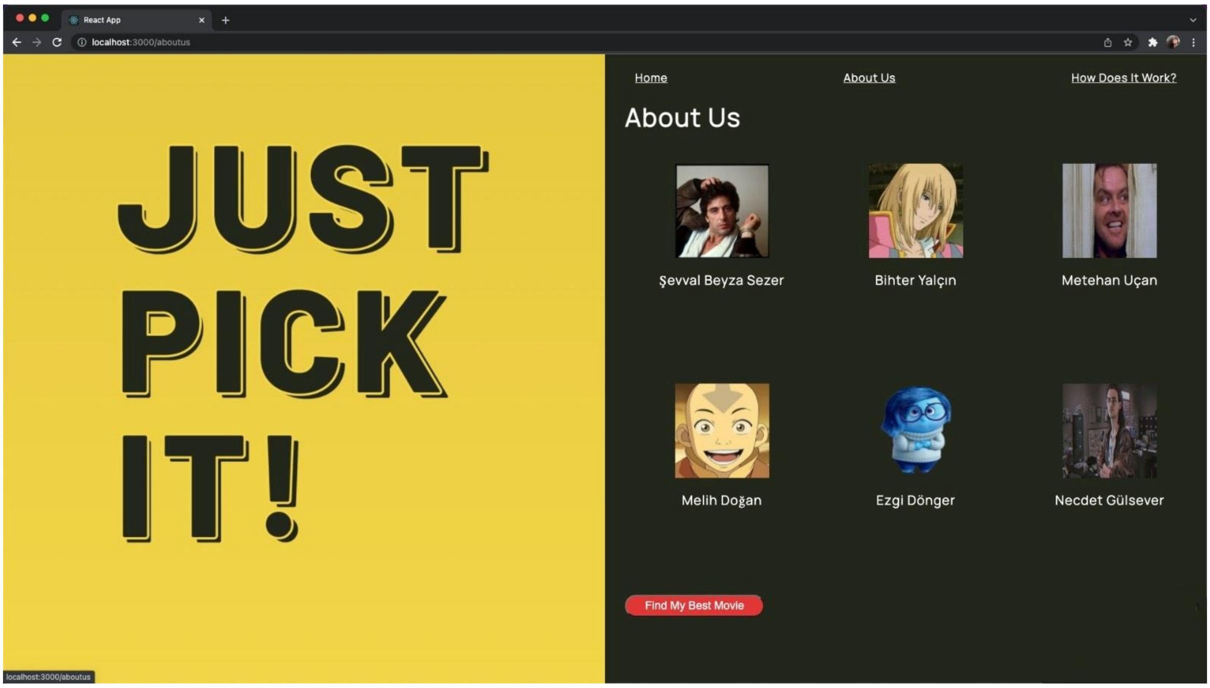

Figure 35 - "About Us" page

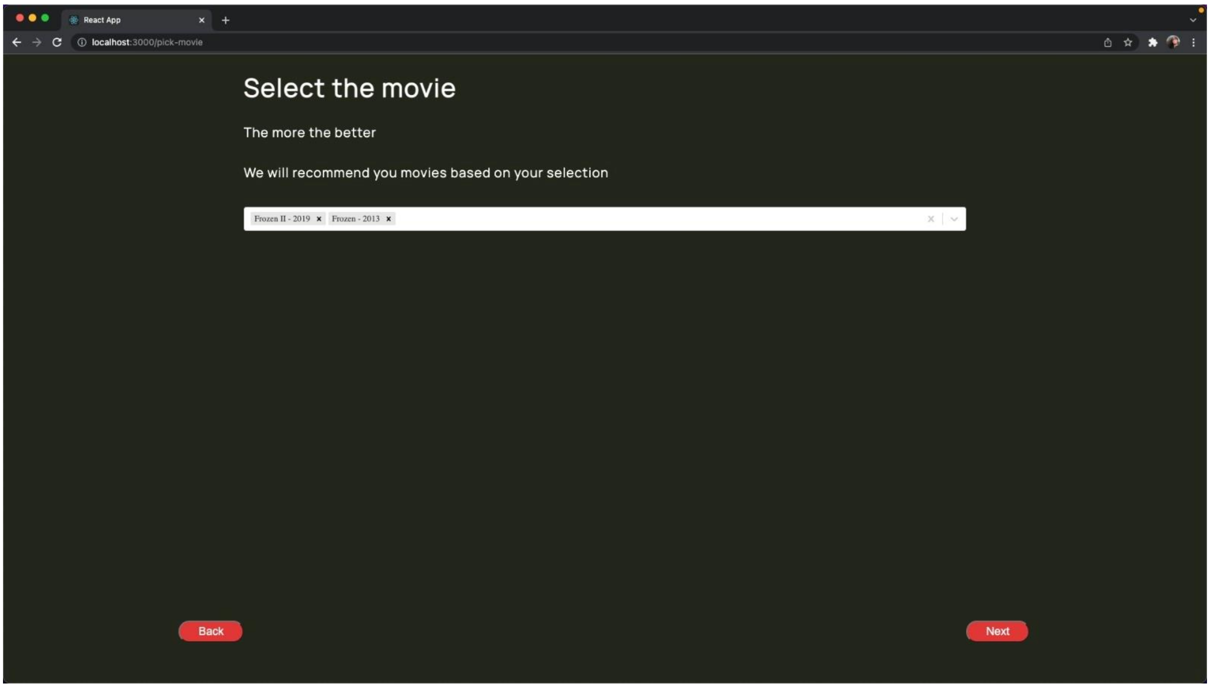

Figure 36 - Selection of the movie

This is the first step where the users select the movies that they want to base their selections on. They can search for a movie and the dropdown will show the movies that are existent in our database. We also show the release year of the movie for distinction.

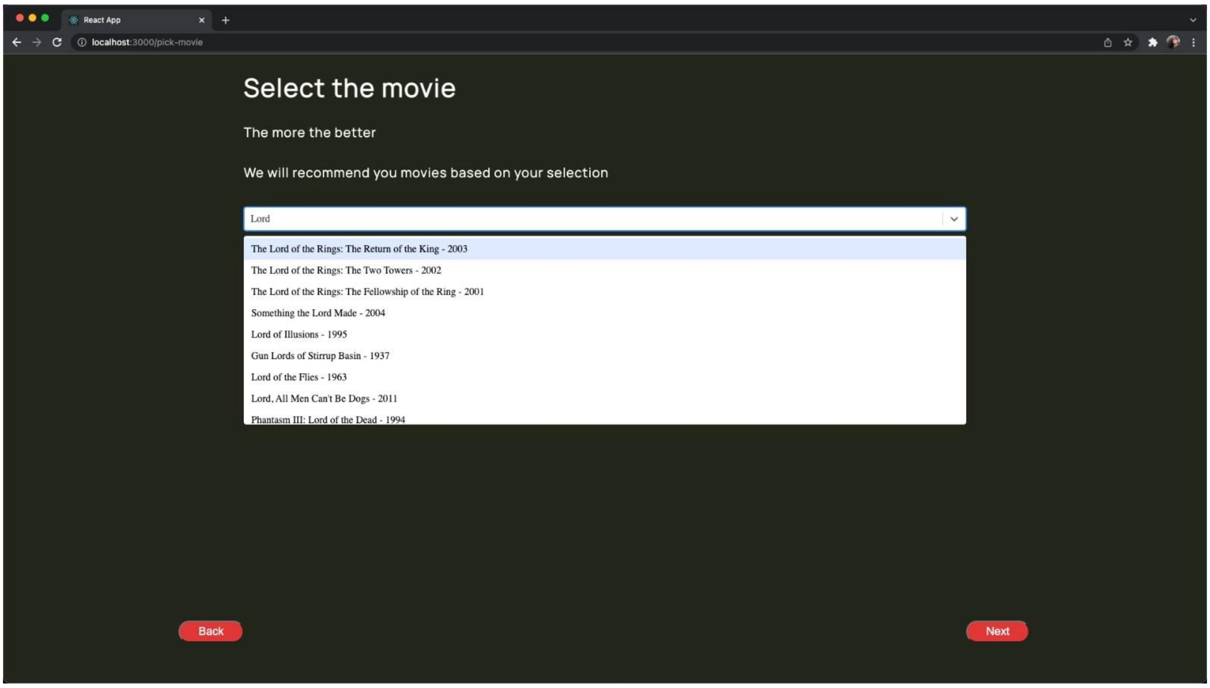

Figure 37 - Selection of the movies where users can search and select movies

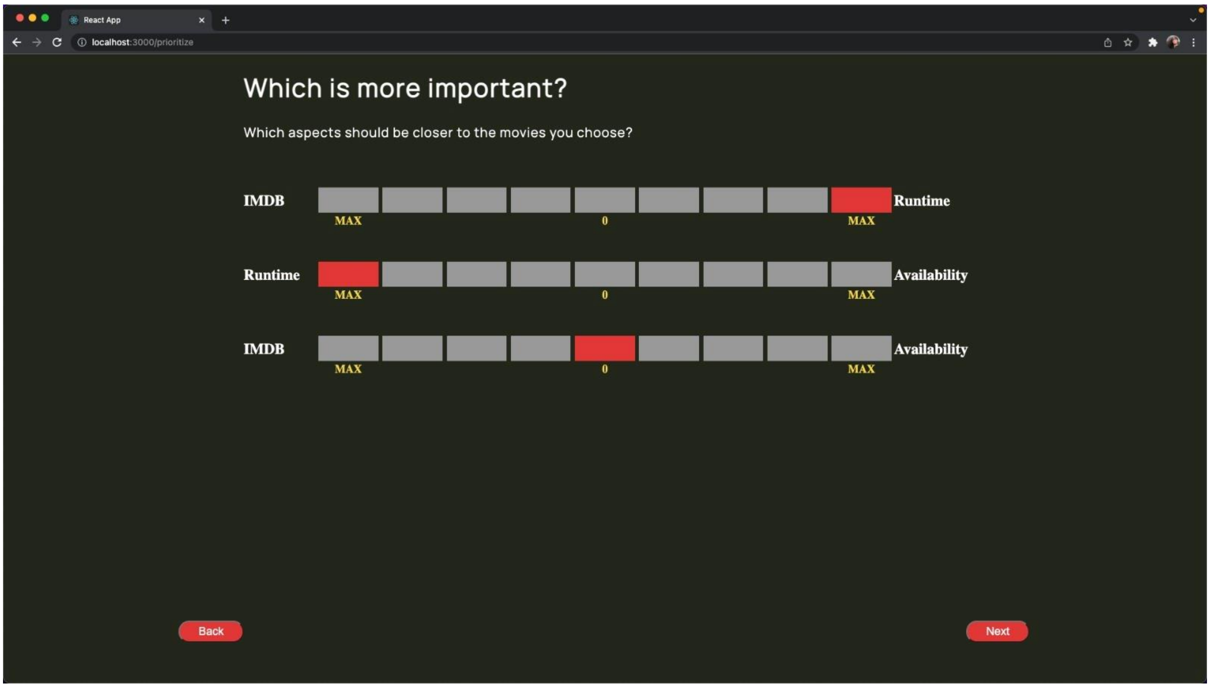

Figure 38

This is the second section where they assess the importance of criteria. There are 3 sliders, these sliders show IMDB, Runtime, and Availability. Users can select what is important for them. If they make an inconsistent assessment, we will warn them with a pop-up.

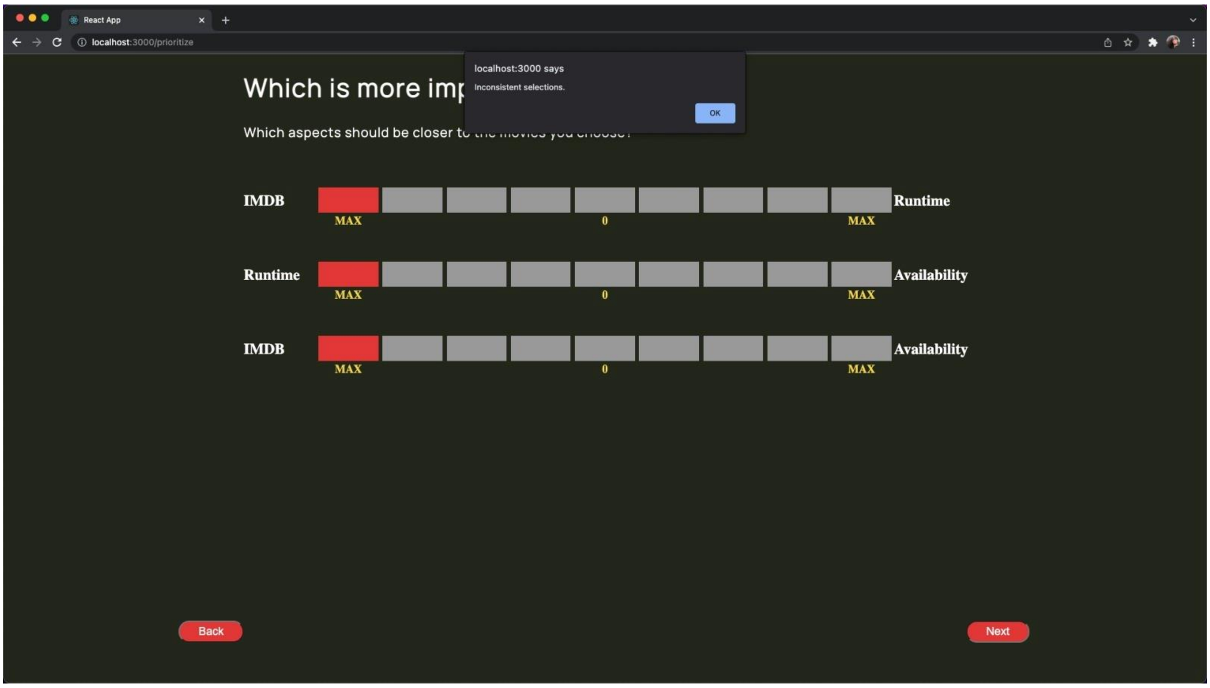

Figure 39 - Inconsistent selection warning

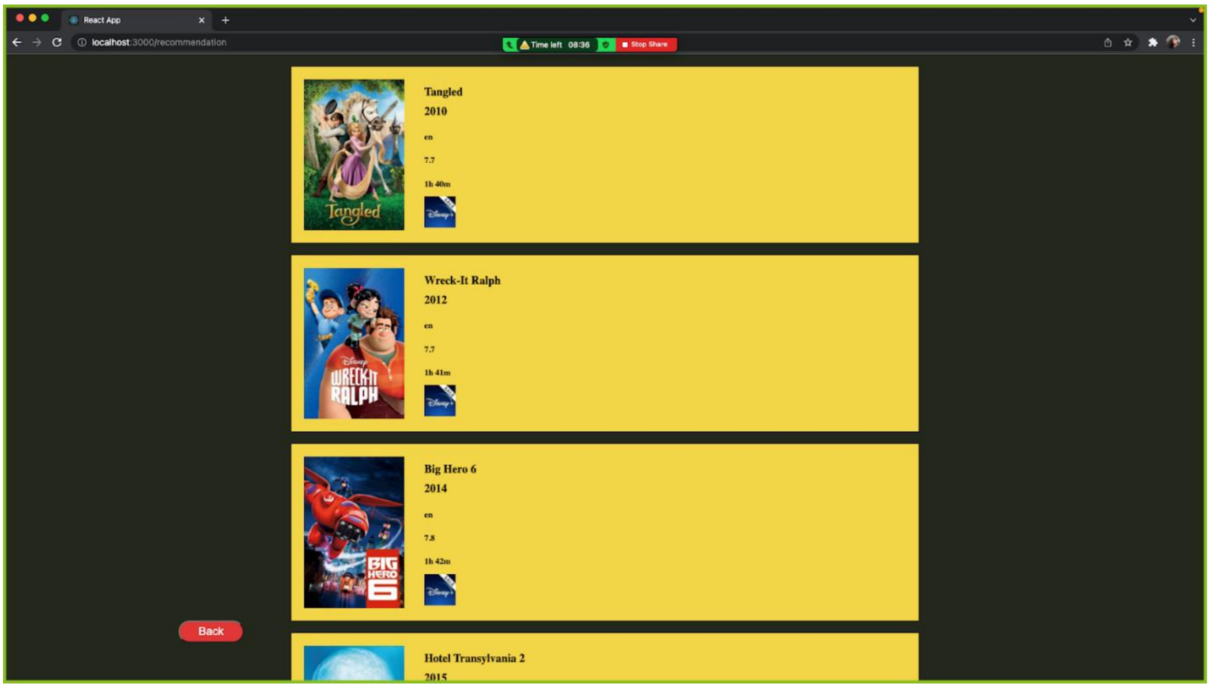

Figure 40

This is the recommendations page, most relevant to less, top to bottom. Users can see the name of the movie, its release date, original language, available platform, and its picture. Here, they can go back to do what-if analysis and see the new results.

6. PROJECT PLAN

As Project Team 5, every member of the team has a unique ability and profession. So, we can say that our project team represents the variety of skills and knowledge to successfully finish the project. We came together regularly and show excellent collaboration during the project development. Each member did not hesitate to take initiative and had an equal contribution to the project. We follow the common team virtues during the discussions and brainstorming.

After the mid-report submition and its presentation, we did a one-week break for the project development because we have already did data collection and data cleaning before the mid-report deadline.

Then, we started to established our final model and concluded it.

Basic steps of our project and time plan:

1. Deciding on which decision environment we want to work on
2. Conducting surveys, interviews and literature review
3. Collecting and cleaning data set
4. Analyzing the data
5. Choosing and developing the analytical model we will employ
6. Build our interface with ReactJS
7. Performing system validation tests

Even though we achieve meeting regularly for the project development, it is hard to find the optimum time slot for each member since every team member has different schedules. Also, each member lives in different locations in Istanbul. We overcame this challenge by being flexible with the meeting method.

Meeting Place: Zoom, Discord, and face-to-face meetings in different locations

Meeting Time: Two or three times a week

Coordinator: Necdet Gülsever

Task Allocation: Worked collaborative

7. CONCLUSION

When we launched this project, we wanted to make sure that users were getting reliable movie recommendations based on their tastes so that they could spend less time deciding what to watch. We conducted a survey and found that to solve this issue, people require a movie recommendation system.

First, to understand what users value the most about a movie, we analyzed the survey and conducted an interview with an expert. These research methods help us to discover that users care about the overview of the movie, original language, rating, and cast. Then we decided to make our decision-making system a web-based platform that takes user-submitted movies as an input and recommends movies based on them.

We got our data from TMDB (The Movie Database is a community-driven database for movies and TV shows.) We cleaned irrelevant or null data and added columns based on the results of our survey and interviews.

To decide which movies to recommend, we used content-based filtering to calculate the cosine similarity. Cosine similarity calculates the similarity distances between movies, to implement this we had to vectorize our word-based data, we used the TF-IDF method for that. This gave us the top n similar movies from similarity property, relevant to the movies entered. We also did AHP and took the importance of some criteria and use the top n similar movies to rank them again. After having these two similarity values (ranking of alternatives & cosine similarity), we made a cross sorting to achieve peak relevance. Finally, we coded an interface with HTML, CSS, JS, and React to take the necessary inputs for the model and show the output which is the movie recommendation.

One of the limitations of our system is that the algorithm is based on an input from users, which is a movie. However, the user must choose from the dataset that we have provided.

The strength of our decision support system is that we use a variety of approaches to provide accurate suggestions. As mentioned, we employed cosine similarity to find similarities between movies, followed by AHP. This way, both movie features and user preferences are taken into account.

For future research, real-time data may be utilized to increase the dataset and make it more current. Additionally, the user history data can be kept to increase the range of recommendations.

REFERENCES

Acevedo, M., & Krueger, J. I. (2004). Two Egocentric Sources of the Decision to Vote: The Voter's Illusion and the Belief in Personal Relevance. *Political Psychology, 25*(1), 115-134. doi:10.1111/j.1467-9221.2004.00359.x

Aggarwal, C. C. (2016). An Introduction to Recommender Systems. In C. C. Aggarwal, *Recommender Systems* (pp. 1-28). Cham, Switzerland: Springer. doi:10.1007/978-3-319-29659-3_1

Almeida, F., Duarte Santos, J., & Augusto Monteiro, J. (2020). The Challenges and Opportunities in the Digitalization of Companies in a Post-COVID-19 World. *IEEE Engineering Management Review, 48*(3), 97-103. doi:10.1109/EMR.2020.3013206

Apáthy, S. (n.d.). *History of recommender systems: A brief and superficial overview*. Retrieved November 2021, from Onespire: https://www.onespire.net/news/history-of-recommender-systems/

Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-Based Systems, 46*, 109-132. doi:10.1016/j.knosys.2013.03.012

Bruine de Bruin, W., Parker, A. M., & Fischhoff, B. (2007). Individual differences in adult decision-making competence. *Journal of Personality and Social Psychology, 92*(5), 938-956. doi:10.1037/0022-3514.92.5.938

Burke, R. (2007). Hybrid Web Recommender Systems. In B. P., K. A., & N. W., *The Adaptive Web. Lecture Notes in Computer Science* (Vol. 4321, pp. 377-408). Berlin, Heidelberg: Springer. doi:10.1007/978-3-540-72079-9_12

Dietrich, C. (2010). Decision Making: Factors that Influence Decision Making, Heuristics Used, and Decision Outcomes. *Inquiries Journal, 2*(02).

Dworkin, S. L. (2012). Sample Size Policy for Qualitative Studies Using In-Depth Interviews. *Archives of Sexual Behavior, 41*(6), 1319-1320. doi:10.1007/s10508-012-0016-6

Isinkaye, F., Folajimi, Y., & Ojokoh, B. (2015). Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal*, 261-273. doi:10.1016/j.eij.2015.06.005

Jannach, D., Zanker, M., Felfernig, A., & Friedrich, G. (2010). Knowledge-based recommendation. In D. Jannach, M. Zenker, A. Felfernig, & G. Friedrich, *Recommender Systems: An Introduction* (pp. 81-123). Cambridge University Press. doi:10.1017/CBO9780511763113.006

Javed, M. (2020, November 4). *Using Cosine Similarity to Build a Movie Recommendation System*. Retrieved from Towards Data Science: https://towardsdatascience.com/using-cosine-similarity-to-build-a-movie-recommendation-system-ae7f20842599

Juliusson, E. Á., Karlsson, N., & Gärling, T. (2005). Weighing the past and the future in decision making. *European Journal of Cognitive Psychology, 17*(4), 561-575. doi:10.1080/09541440440000159

Krathwohl, D. R. (1998). *Methods of educational & social science research: An integrated approach* (2nd ed.). Longman/Addison Wesley Longman.

Lops, P., Jannach, D., Musto, C., Bogers, T., & Koolen, M. (2019). Trends in content-based recommendation. *User Modeling and User-Adapted Interaction, 29*, 239-249. doi:10.1007/s11257-019-09231-w

Mason, M. K. (n.d.). *Short History of Collaborative Filtering*. Retrieved November 2021, from Moyak: https://www.moyak.com/papers/collaborative-filtering.html

Mishra, U. (2021, May 29). *What is a Content-based Recommendation System in Machine Learning?* Retrieved November 2021, from Analytics Steps: https://www.analyticssteps.com/blogs/what-content-based-recommendation-system-machine-learning

Naqvi, N., Shiv, B., & Bechara, A. (2006). The Role of Emotion in Decision Making: A Cognitive Neuroscience Perspective. *Current Directions in Psychological Science, 15*(5), 260-264. doi:10.1111/j.1467-8721.2006.00448.x

Navarro, J. G. (2021, August 12). *Preference for watching a movie for the first time at a theater instead of via a streaming service in the United States from November 2018 to June 2020*. Retrieved November 2021, from Statista: https://www.statista.com/statistics/947757/theaters-streaming-watching-movies/

O'Reilly Media, Inc. (n.d.). *Cosine similarity*. Retrieved December 19, 2021, from DeepAI: https://deepai.org/machine-learning-glossary-and-terms/cosine-similarity

Qutbuddin, M. (2020, May 7). *Comprehensive Guide on Item Based Collaborative Filtering*. Retrieved November 2021, from Towards Data Science: https://towardsdatascience.com/comprehensive-guide-on-item-based-recommendation-systems-d67e40e2b75d

Reikeras, H. (n.d.). *How to Build a Content-Based Recommender System For Your Product*. Retrieved December 25, 2021, from The OfferZen Community Blog : https://www.offerzen.com/blog/how-to-build-a-content-based-recommender-system-for-your-product

Ricci, F., Rokach, L., & Shapira, B. (2015). Recommender Systems: Introduction and Challenges. *Recommender Systems Handbook*, 1-34. doi:10.1007/978-1-4899-7637-6_1

Šakan, D., Žuljević, D., & Rokvić, N. (2020). The Role of Basic Psychological Needs in Well-Being During the COVID-19 Outbreak: A Self-Determination Theory Perspective. *Frontiers in Public Health, 8*, 713. doi:10.3389/fpubh.2020.583181

Sarwar, B., Karypis, G., Konstan, J., & Reidl, J. (2001). Item-based collaborative filtering recommendation algorithms. *Proceedings of the Tenth International Conference on World Wide Web - WWW '01*. doi:10.1145/371920.372071

Schafer, J. B., Frankowski, D., Herlocker, J., & Sen, S. (2007). Collaborative Filtering Recommender Systems. In B. P., K. A., & N. W., *The Adaptive Web. Lecture Notes in Computer Science* (Vol. 4321, pp. 291-324). Berlin, Heidelberg: Springer. doi:10.1007/978-3-540-72079-9_9

Stanovich, K. E., & West, R. F. (2008). On the relative independence of thinking biases and cognitive ability. *Journal of Personality and Social Psychology, 94*(4), 672-695. doi:10.1037/0022-3514.94.4.672

Stoll, J. (2021, October 22). *SVOD service user shares in the U.S. 2015-2021*. Retrieved November 2021, from Statista: https://www.statista.com/statistics/318778/subscription-based-video-streaming-services-usage-usa/

Vidiyala, R. (2020, 10 2). *How to Build a Movie Recommendation System*. Retrieved 11 10, 2021, from Towards Data Science: https://towardsdatascience.com/how-to-build-a-movie-recommendation-system-67e321339109

Xue, F., He, X., Wang, X., Xu, J., Liu, K., & Hong, R. (2019). Deep Item-based Collaborative Filtering for Top-N Recommendation. *ACM Transactions on Information Systems, 37*(3), 1-25. doi:10.1145/3314578

APPENDIX A: INTERVIEW

Kendinizden kısaca bahsedebilir misiniz?

İsmim Levent, 32 yaşındayım. Doğuş Teknoloji'de Budget Planning and Financial Control Specialist olarak çalışıyorum. Bunun yanı sıra Altyazı Sinema Derneğinin kurucularındanım. Çocukluğumdan beri sinemaya ve sahne sanatlarına ilgiliyim. İlk gittiğim film Tarzan, zamanında mahalle arası sinema salonları vardı. Artık hepimizin evi kendi özel sinema salonumuz. Üniversite yıllarımda Sinema Kulübü Başkanlığı yaptığım ve tiyatrolarda ışıkçı olarak çalışmışlığım var. Nuri Bilge Ceylan en sevdiğim yerli yönetmen, en sevdiğim yabancı yönetmen ise Wes Anderson. Mithat Alam Film Merkezi'nde her cumartesi film yorumları üstüne çalışmalar yapıyoruz.

Sizce film seçerken insanlar açıklamalara/film özetlerine dikkat ediyor mu / ne kadar dikkat ediyor?

Çok ediyor. Sinapsis her şeydir. Kitapçıklardaki açıklamalar ayrı bir pazarlama ayrı bir sanat eseridir. Şahsi olarak tercih sebebim değildir ama insanların algısını yönlendirir. film kapakları gibi düşününce arka kapak gibi. Ön kapak afiş, arka kapak kitapçık.

Film seçerken internette ne gibi anahtar kelimeler tercih ediliyor?

Spesifik bir muhabbet için olur. Flörte atmalık filmleri. Key value gibi. Özel konseptlere göre yönelir ama dram, nuri bilge bunlar genre. Bunlar için anahtar kelime demek doğru değil.720 p yazmak filmin kalitesini belirtir. Filmi bulmadan 720p izle diye direkt film aranmaz. İzlenecek film bulunduktan sonra filmin sitesi de pek mühim değildir. Önemli olan doğru filmi bulabilmektir.

Seyirci için film seçerken belirli kriterler var mı?

Kişisel olarak yönetmene bakarım. Genel olarak millet oyuncuya bakar. Yönetmene bakmazlar. Oyuncu her şeydir. Film biter " aa Çağan Irmak'ın filmiymiş" derler fakat Farah Zeynep oynuyor diye arayıp buluyorlar.

Evde izlenecek filmler ile sinemada izlenecek filmler birbirinden farklı mı?

Pandemiden sonra fark kalmadı. Bunların hepsi artık evde de izlenebilir. Sinema bir endüstri ama ayrım kalmadı.

Film seçimi yaparken film bütçesi veya gişe hasılatı ne kadar etkili?

Bunlar kriter değil. Yani içeriği de beslemez. Kaliteyi de etkilemez.

Genre'lar film seçiminde etkili mi?

Kendi adıma olmaz. Yorgunsun kafa dağıtmalık film istersin ama genre tek başına bir belirleyici değil. Mesela polisiye seversin ama her polisiye film de izlenmez. Bir oyuncunun her filmi izlenir lakin bir genrenin her filmi izlenmez.

Dil bariyeri izleyicileri film seçiminde yönlendiriyor mu?

Dili bilip bilmemek değil, günlük hayatta sese aşina olup olmamak gibi. Yine insanın modu bir parametre. Hafif uykuluyken İsveççe film izlemek size kolay veya çekici gelmez. Örneğin bir filmi altyazısını olmadığı için izlemeyi tercih etmeyecek onlarca insan vardır. Bu filmin altyazısı Portekizce bir sitece yapılabiliyordur fakat siteye aşina olmayan birisi bu sitede nasıl çeviri yapacağını bilmediği için yine aynı şekilde dil bariyerinden dolayı filmi tercih etmeyecektir. Bu sebepten streaming platformlar çevirili içeriklere ev sahipliği yaptığı için çoğu insanın ilk tercihi. Bunun yanı sıra çevirinin kalitesi çok etkiliyor. Profesyonel sitelerin çeviri kalitesi de izleyiciyi cezbediyor.

Filmin renkli veya siyah-beyaz olması seçerken bir kriter mi ?

Color önemli bir faktör. en az dil kadar önemli. Filmin çekildiği yer, yapımcı şirketi, dağıtım şirketi ve senarist gibi etkenler film seçiminde ne kadar etkili. Seyirci açısından yapım şirketi, dağıtım şirketi ve senarist önemsiz ama filmin çekildiği yer tek başına etken. Kartal Tibet çekici değildir fakat filmin Fransa'da geçmesi etkileyicidir.

Oyuncu mu yönetmen mi film seçiminde etkili?

Ben yönetmen seçerim ama ekseriyetle oyuncu seçer. Mood da önemli bir etken. Canım sıkkın ise ağır film çeken yönetmenin filmi yerine iyi oyuncunun eğlenceli filmini seçerim mesela.

Filmin yapım yılı etki seçime etki eder mi ?

Evet, etkili. Mesela renkli film bulmak için yıl bir kriter, eskiden siyah beyazdı.

Film süresi ile filmin tercih edilmesi arasında bir ilişki var mı?

Filmin süresi etkilidir. çok önemli. Mesela Vine diye bir app vardı. İçerikler 6 sn idi ama insanların bunu bile izlemeye tahammülü yoktu, tek bir parmak hareketi ile değiştiriyorlardı. Bazen kısa filmler geliyor Mithat Alam'a. 15dakikalık film için uzundu diyor etkinliğe gelen senaristler bazen de seyirciler. Mesela bir konu var ağır ağır da işlersin 15 dakikada da… "bunu anlatmak için bu kadar zaman harcamaya gerek var mı? " düşüncesi seyircide genel anlamda önemli bir yere sahip. Özellikle film seçerken insanlar çok vakit kaybettiği için filmi izlerken de vakit harcamak istemiyorlar. Bu sebeple film seçiminde geçen süre de film süresi kadar önemli.

APPENDIX B: SOURCE CODE

The full source code of our project can be found here: https://gitlab.com/mis463

APPENDIX C: MASTER PLAN

Project Code	5
Project Title	Just Pick It!
Team Members	M. Melih Doğan, Ezgi Dönger, Necdet Gülsever, Şevval Beyza Sezer, Metehan Uçan, Bihter Yalçın

Phase	Planned		Actual		**Complete%**	**Problems**
	Start	**Finish**	**Start**	**Finish**		
Team Formation	6.10.2021	13.10.2021	6.10.2021	7.10.2021	100%	*None*
Project Proposal Report	-	-	-	-	-	-
Data collection	20.10.2021	27.10.2021	28.10.2021	5.11.2021	100%	*Determining the subject was the hard part. Also, retrieving data took time.*
Literature Review (Library, Web, former studies)	1.11.2021	12.11.2021	1.11.2021	18.11.2021	100%	*It is finished but additional technical review could be done for the development process*
Interviews with experts, decision makers in the related area	1.11.2021	1.11.2021	2.11.2021	3.11.2021	100%	*Interviewee's plan is changed. Also, transcription of the interview took one more day.*
Development of the model	1.11.2021	13.11.2021	1.11.2021	13.11.2021	100%	*None*
Project Midreport	15.11.2021	20.11.2021	15.11.2021	21.11.2021	100%	*None*
Coding interfaces	1.12.2021	18.12.2021	1.12.2021	20.12.2021	100%	*None*
Tests for verification	3.12.2021	20.12.2021	4.12.2021	21.12.2021	100%	*None*
Project Final Report	21.12.2021	23.12.2021	23.12.2021	26.12.2021	100%	*None*