A close up of a logo

Description automatically generated

ITC 6460 Cloud Analytics

Faculty: Prof. Sergiy Shevchenko

Module 3: Project Proposal

Date: 23rd February 2021

Submitted by:

Steve Duncan

Zachary Farrar

Mohit Manohar Manjaria

Vamshi Munagandla

Sakshi Suri

# Abstract

In the past twenty years, global media consumption has been revolutionized. The proliferation of high-speed internet connectivity and accelerating pace of technological development in computing has shifted the landscape of media consumption from limited content broadcast-based television and radio to on-demand streaming services that have access to an increasingly vast amount of content. With this shift in media delivery can content availability, new players have arisen in the industry that are competing with traditional media delivery networks. In order to expand their market share, provide differentiation, and facilitate their end users’ ability to access these huge content catalogs, media streaming companies have increasingly focused on improving their end users’ experience. A key aspect of this focus is to facilitate their users’ ability to find relevant and interesting content, and many have invested heavily on recommender systems in order to match users with relevant content. This project will explore the practical aspects of media recommender systems, provide a machine learning based prototype, and identify key aspects for future consideration of similar technologies.

# Background

In 2007, Netflix began streaming television shows and movies over the internet, sparking a revolution in media content delivery that has redefined the industry. Netflix has since grown into a media giant, spending more than twelve billion on programming in 2019 for its 158 million subscribers worldwide.[[1]](#footnote-1) Netflix is certainly not the only player in the market as new media content production and delivery companies continue to enter the market. Flixed.com recently reported that there are more than two hundred online streaming services available[[2]](#footnote-2), and many of the more traditional media giants, such as Disney, WarnerMedia, and NBCUniversal, have also launched streaming services.[[3]](#footnote-3) All these services represent a huge market, valued at more than $50 billion in 2020 and projected to achieve a compound annual growth rate of twenty one percent (21%) from 2021 to 2028.[[4]](#footnote-4)

With this shift from a broadcast-based content delivery system to an interactive, on-demand, internet-based delivery system, there has been an explosion of available content. Netflix, the market leader in terms of subscribers reportedly had nearly 6,000 movies and series (more than 50,000 individual titles) in 2020.[[5]](#footnote-5) Some of Netflix’s other competitors: Amazon Prime, Hulu, and others, have content libraries nearly as large.

With such a vast selection of content available on-demand, the need for services that help to match users to the content that they want to consume has become a major feature of the overall user experience for online content streaming platforms. By maximizing their users’ ability to quickly and efficiently find content that they want is a huge point of differentiation in a savagely competitive market. These conditions, in parallel with similar trends in the ecommerce marketplace, have led to the rise of increasingly sophisticated content recommendation services using advanced data science techniques, Machine Learning (ML), and big data.

In very basic terms, content recommender systems are algorithms that are designed to match users with the things they want. Those things could be physical products at an ecommerce marketplace (e.g., Amazon.com), music from one of the audio streaming services (e.g., Spotify.com), or movies from a video content provider (e.g., Netflix.com). There have arisen two primary categories of recommender systems: Collaborative and Content, and various hybrid models that try to balance the benefits of both. A very basic depiction of recommender system categorization is shown in Figure 1 and discussed in more detail below.

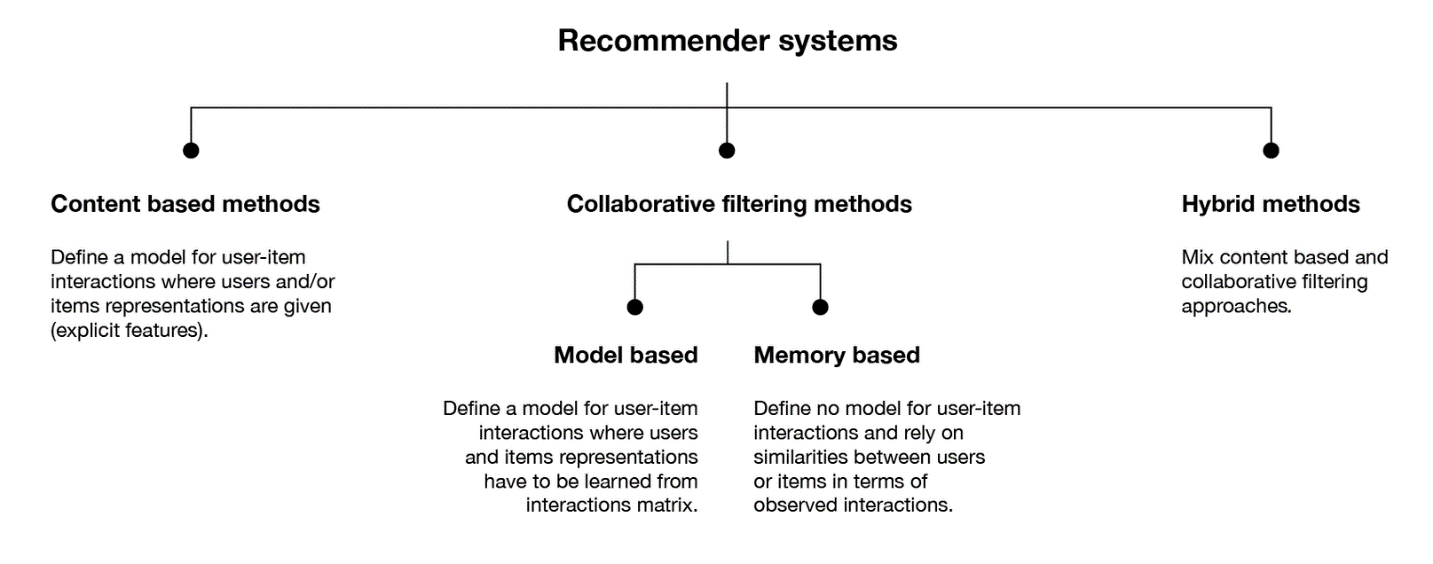


Figure 1 Recommender System Types (Rocca, 2019)

## Content Based Recommender Systems

Content based filtering is one of the simplest recommender systems to implement. The premise of content-based systems is that models can be created from knowledge of user and item attributes that allow for matching based on a variety of criteria. The user-based data can be explicit (an adult male user submits a preference: “I like action movies”) or implicit (action movies are a very popular genre for adult men).[[6]](#footnote-6) For a user, these details may include demographic information such as age, occupation, gender, region, etc. Information about participating items (movies in the case being investigated), may include actors, duration, rating, genre, etc. The content-based methods use details about the users and items to build a model that predicts which items and users are likely to be matched. Content based models may be created based on user-centric models, where recommendations are made by how well items match a user profile, or based on item-centric models, where recommendations are made based on how well users match an items profile. A depiction of these two models is provided in Figure 2.

Compared to the other major category of recommendation system techniques, Collaborative Filtering, content based models are easier to implement due to the presence of immediately usable data and do not have the same “cold-start” challenges that collaborative filtering faces. The downsides of content based techniques include user reluctance by some users to provide relevant, demographic information that would support the models. It also may be the case that recommendations often feel static and not relevant to a particular user where sufficient details are not known.[[7]](#footnote-7) In these cases, an alternate method such as recommendation of generic, random, or popular items to a new user, or incorporation of a hybrid recommendation model, may be beneficial to the end user experience.

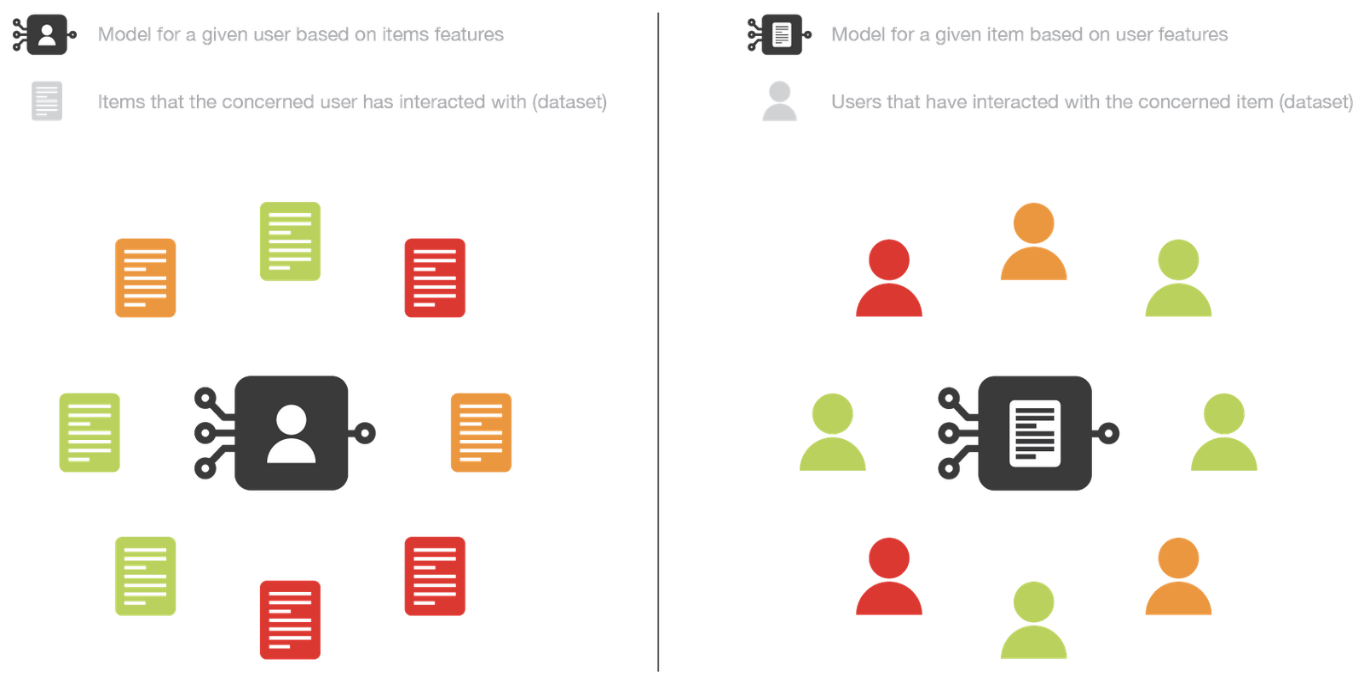


Figure 2 User & Item Focused Content Recommender Models (Rocca, 2019)

## Collaborative Recommender Systems

In contrast to content-based methods, collaborative filtering generates recommendations based on interactions between users and items. These methods do not require any specific information about the user or the items, only their interactions – in the context of this project: user submitted movie ratings. At a very high level, a user would submit ratings of various movies in the catalog and the system would generate recommendations based on favorably rated items from users with similar rating patterns.[[8]](#footnote-8) These data are reflected in a two-dimensional matrix where user IDs form one axis, movie titles form the other axis, and the applicable ratings complete the matrix. This is graphically depicted in Figure 3.

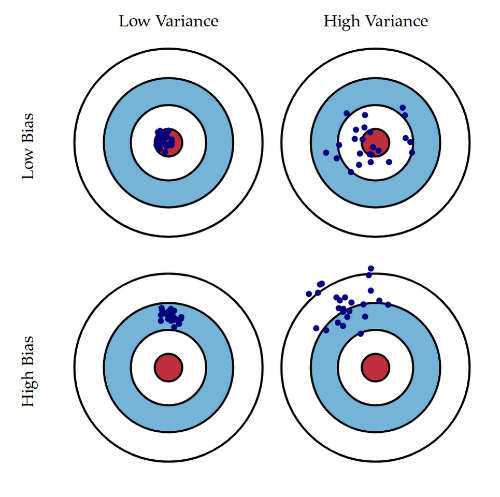
Relative to content filtering methods, collaborative recommendation techniques can provide better results in terms of diversity (dissimilarity between recommended items), serendipity (unexpectedly relevant recommendations), and novelty (recommendations that were previously not known by a user).[[9]](#footnote-9) Recommendation system design and evaluation considerations are discussed in more depth below.

Collaborative filtering methods can be further broken down into two primary sub-categories: Memory and Model based methods. Memory based approaches rely solely on past interactions between users and items (the user submitted movie ratings) and model-based filtering relies on machine learning to train a model to better predict applicable recommendations. Relative to each other, memory based collaborative systems tend to have a lower theoretical bias and higher variance than model based collaborative recommenders, as depicted in Figure 5 below.



Figure 3 Collaborative Recommender System (Rocca, 2019)

The user specific inputs based on their participation with the model can lead to better recommendation outcomes, but has some notable drawbacks. The first challenge that a collaborative model has to overcome is the “cold-start” problem. The cold-start problem is a reflection of a non-existent or sparse data set for a user until that user provides feedback on the relevant items participating in the recommendation system.[[10]](#footnote-10) As a user builds an item feedback profile, the collaborative model will be able to make increasingly better matches with similar users. Some common methods to overcome the “cold-start” challenges would be to recommend random or popular items to particular users or specific items to random users until a sufficient amount of interaction data is compiled for the algorithm to operate with reasonable accuracy.



**Collaborative – Model Based:**

Higher Bias, Lower Variance

**Collaborative – Memory Based:**

Lower Bias, Higher Variance

**Content Based:**

Highest Bias, Lowest Variance

Figure 4 Relative Bias and Variance in Recommender Models (Rocca, 2019)

Another challenge that collaborative models is that these models are more computationally extensive, more complex to implement and manage, and more expensive. With collaborative models, it is important to consider size of the intended data sets and select appropriate algorithms, such as factorization machines, to ensure that the process is economically sound.[[11]](#footnote-11)

## Hybrid Systems

As with any system, the means and methods chosen will have benefits and constraints. Some of the primary benefits and constraints relative to content and collaborative recommendation systems have been identified in prior sections. Many modern recommendation engines use hybrid approaches that balance the strengths and weaknesses of two or more techniques. A common hybrid system is to address the cold-start challenge of collaborative systems by implementing a content-based system until sufficient data is generated by a user to incorporate collaborative techniques.[[12]](#footnote-12) A graphical depiction of this type of hybrid system is provided in Figure 4.

## Design Considerations and Performance Evaluation Criteria

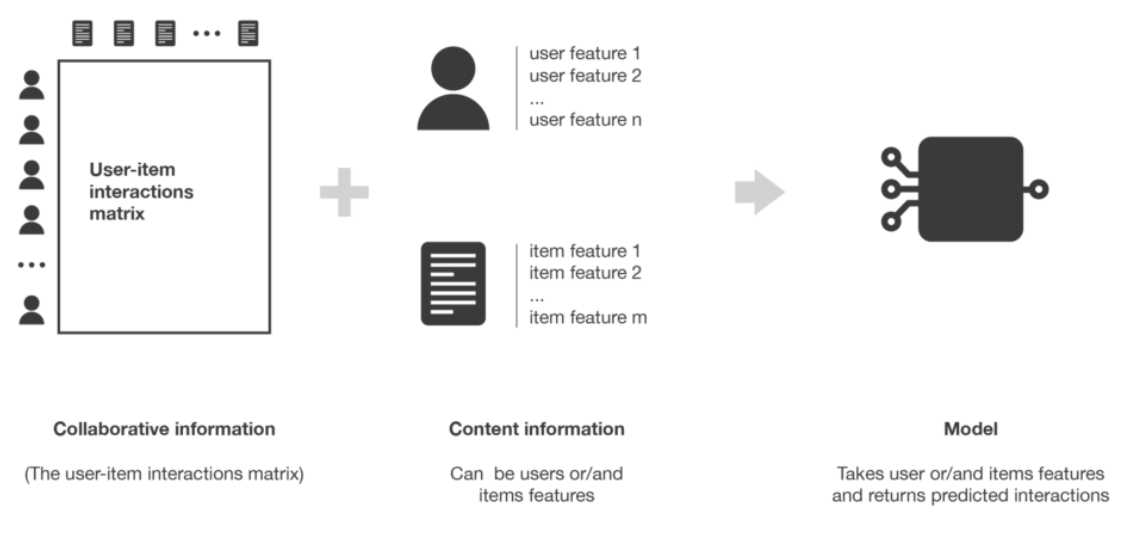


Figure 5 Content Based Recommendation Model (Rocca, 2019)

When establishing design criteria and requirements for a recommendation system, it is critical that the intent of the system is well understood. There is often a natural desire to focus minimizing bias and variance at the expense of the overall user experience. In Shani and Gunawardana’s paper, *Evaluating Recommendation Systems*, they point out that recommendation systems are used for more than precise predictions of items applicable to the user:

*“In many applications people use a recommendation system for more than an exact anticipation of their tastes. Users may also be interested in discovering new items, in rapidly exploring diverse items, in preserving their privacy, in the fast responses of the system, and many more properties of the interaction with the recommendation engine.”* [[13]](#footnote-13)

Shani and Gunawardana propose a an excellent guide for design considerations. Their evaluation criteria is defined below:[[14]](#footnote-14)

User Preference - The business operating the recommender system may want to place a higher importance on certain users. E.g. users who have interaction with a larger number, or wider variety of items may be prioritized over users who have only interacted with one item.

Prediction Accuracy – This is a measure, often qualitative, of how close the recommendations are to their intended purpose. In terms of bias and variance depicted in Figure 5, accuracy is the bullseye.

Confidence – Confidence reflects how often the recommendations will be within a reasonable level of accuracy. In terms of bias and variance depicted in Figure 5, confidence is similar to the variance and will generally improve as data used in the models becomes more robust.

Trust – this aspect reflects a qualitative measure of the users’ trust in the recommendations. It could be that the recommendations provided are not chosen, but the user recognizes that the system is providing reasonable recommendations. This may encourage users to accept recommendations for previously unknown items.

Novelty – Novelty is related to the system providing recommendations for items for which they were previously unaware. A simple way to improve novelty in a system is to exclude items that a user has previously interacted with (e.g. a movie for which the user has submitted a review).

Serendipity – This is an evaluation based on how surprising the recommendations are. This is a challenging criterion in that it represents more than providing random results or recommending a movie based on similar casts. Serendipity is achieved if it is a surprising result that is novel and accurate.

Diversity – Diversity is a measure of variety in the recommendations that does not have a materially negative impact on prediction accuracy.

Utility – This is a measure of how useful a recommendation is to a user. One example suggested by Shani and Gunawardana is that a movie that is rated by a user with five-stars may be reasonably understood to be an excellent movie by that user (and other similar users). may be considered to be an excellent movie. Therefore, there is more utility in recommending the movies that have five-star ratings by relevant users.

Risk – this is a measure of how likely it is that the system will recommend something inappropriate.

Robustness – Robustness is a measure of the stability of the recommendation system when presented with bad data or fake information. E.g., correcting for a user intentionally submits poor ratings for all movies that cast a certain actor that declined to go on a date with that user in 2002. Those ratings are not valid to the system and may skew the recommendations.

Privacy – as previously mentioned, content-based models rely on data about the users and the items. It is important to protect the users’ privacy and to help them feel confident that any data they contribute will be handled appropriately.

Adaptivity – Adaptivity is the system’s ability to adapt to changes in the item catalog and/or user preferences over time.

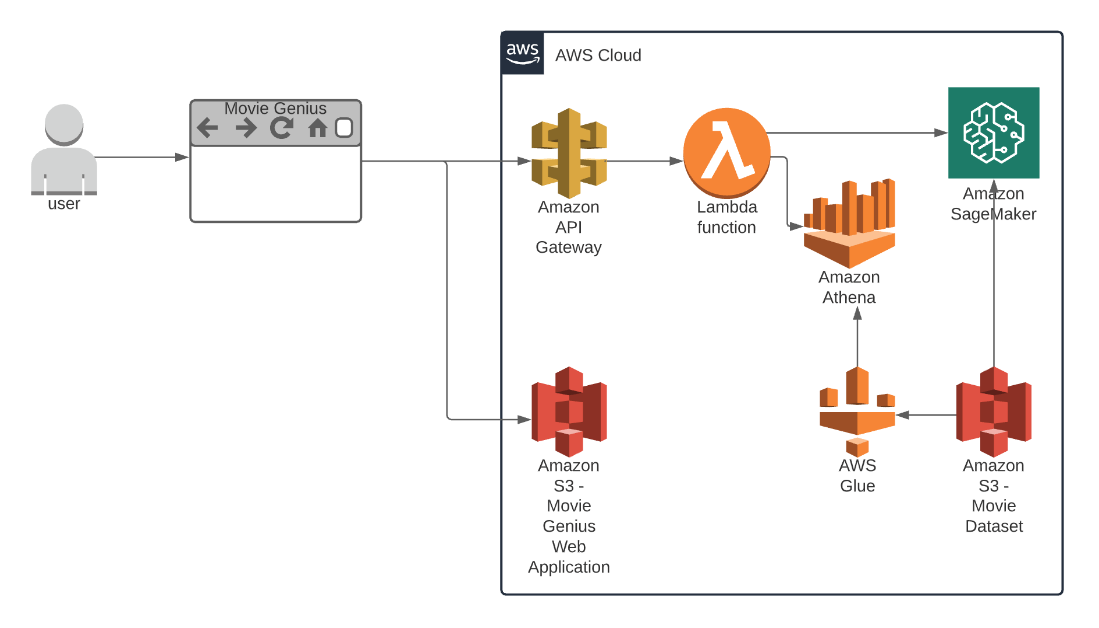
Scalability – This is a measure of the recommendation system’s ability to grow to support demand. This can be a challenge for complex collaborative systems in particular as they are more computationally intensive.

# Design

The movie recommendation system for this project has been developed completely on Amazon Web Services serverless platforms. Utilizing serverless Platforms as a Service (PAAS) have allowed the team to build a working movie recommendation system very quickly. The team was focused on the solution without needing to be concerned about managing servers and other infrastructure.

The movie recommendation system is comprised of a front-end user interface for requesting recommendations, a data access layer for mediating data requests, a machine learning model, a database management system and data storage. The figure and details below will highlight each of the AWS services used to deliver the system functionality.

Figure 6: Movie Recommendation System Cloud Architecture



## Front-End Web Application

The front-end web application is a single page web application comprised of flask, html, css and javascript. The web application will allow a user to choose a movie that they have seen or heard about that they would like to get recommendations for other movies that the user might also like. We chose to host the web application on Amazon S3. S3 serves as both the storage location and the web server for hosting the application. Figure 6 below shows the front end for user inputs.

Table

Description automatically generated

Figure 6 - Front End

## Data Accesss/API Layer

The web application will interact with an api (application programming interface) to make requests and receive the resulting recommendations from the recommendation engine. Amazon API Gateway has been used to define the public end point that the front-end application will interact with. When a GET request is made to this endpoint an Amazon Lambda function will be executed to handle the request processing logic. The lambda function’s job is to make a request to the ML recommendation model to retrieve recommendations and then query our movies dataset to embellish the data and convert to JSON to be returned back to the front-end application for display.

Machine Learning Engine

AWS SageMaker is a fully managed cloud machine learning service. With SageMaker, we have built up a model based on content-based recommendations as it recommends to the user about its favorite movie based on its selection. In our model, we applied cosine similarity that measures the similarity between two vectors of an inner product space. It takes the angle between two non-zero vectors and calculates the cosine of that angle, and this value is known as the similarity between two vectors.

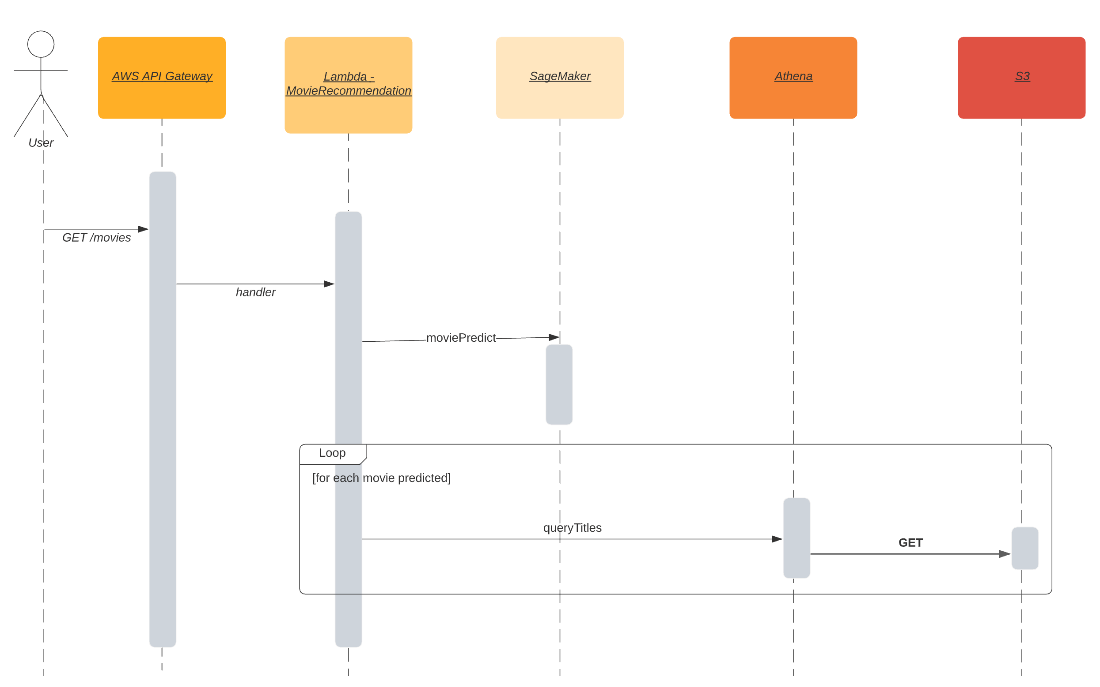
## Data Storage

The movies data is a set of comma separated files (csv) provided by Kaggle, a public resource for data scientists to publish and share data sets for building machine learning models. The data set has been kept in Amazon S3.

## Query Services

In addition to the information retrieved from the Machine Learning model, the application uses Amazon ATHENA to retrieve some additional information about movies that have been recommended to provide the user with as much information as possible to display on the list of recommendations in the front-end web application.

Figure 8: Application request flow from browser to back-end storage



# Conclusion

The design and development of the architecture of this project went very well. Having a solid foundation on the concepts and tools available from AWS as a result of the course content and instruction allowed us to deploy all the intended modules in a serverless architecture. As with many IT projects, the real challenge was integration of the individual models. The communications between the cloud environment and the user interface were challenging and require additional development to improve reliability.

The prototype deployed for this project represents the first step in delivering a robust and capable recommender system. While this content-based system is capable of providing recommendations based on movie characteristics, future projects in this area should work to build a hybrid model that leverages content-based approaches for a particular user until that user has submitted enough movie ratings to facilitate a transition to collaborative filtering.

# References

amazon.com. (2021, 2 18). *Neural Topic Model (NTM) Algorithim*. Retrieved from AWS Documentation: https://docs.aws.amazon.com/sagemaker/latest/dg/ntm.html

Barnes, B. (2019, 11 18). *The Streaming Era Has Finally Arrived. Everything is About to Change.* Retrieved from The New York Times: https://www.nytimes.com/2019/11/18/business/media/streaming-hollywood-revolution.html

Bhowmick, A., Prasad, U., & Kottur, S. (2014). *Move Recommendation based on Collaborative Topic Modeling.* Pittsburgh: Carnegie Mellon University.

Cook, S. (2021, 1 22). *The Complete List of Streaming Services in 2021 – 200+ Services*. Retrieved from Flixed: https://flixed.io/complete-list-streaming-services-2021/

Grand View Research. (2021). *Video Streaming Market Size & Share Report, 2021-2028.* San Francisco: Grand View Research.

Moore, K. (2020, 3 31). *How Long Would It Take To Watch All Of Netflix?* Retrieved from What's On Netflix: https://www.whats-on-netflix.com/news/how-long-would-it-take-to-watch-all-of-netflix/

Rabowsky, B., & Morrison, L. (2020, 11 17). *What's new in recommender systems*. Retrieved from AWS Media Blog: https://aws.amazon.com/blogs/media/whats-new-in-recommender-systems/

Rocca, B. (2019, 6 2). *Introductoin to recommender systems*. Retrieved from Towards Data Science: https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada

Shani, G., & Gunawardana, A. (2011). *Evaluating Recommendation Systems.* Redmond: Microsoft.

1. (Barnes, 2019) [↑](#footnote-ref-1)
2. (Cook, 2021) [↑](#footnote-ref-2)
3. (Barnes, 2019) [↑](#footnote-ref-3)
4. (Grand View Research, 2021) [↑](#footnote-ref-4)
5. (Moore, 2020) [↑](#footnote-ref-5)
6. (Rabowsky & Morrison, 2020) [↑](#footnote-ref-6)
7. (Rabowsky & Morrison, 2020) [↑](#footnote-ref-7)
8. (Rocca, 2019) [↑](#footnote-ref-8)
9. (Rabowsky & Morrison, 2020) [↑](#footnote-ref-9)
10. (Rabowsky & Morrison, 2020) [↑](#footnote-ref-10)
11. (Rabowsky & Morrison, 2020) [↑](#footnote-ref-11)
12. (Rocca, 2019) [↑](#footnote-ref-12)
13. (Shani & Gunawardana, 2011) [↑](#footnote-ref-13)
14. (Shani & Gunawardana, 2011) [↑](#footnote-ref-14)