Movie Recommendation System Using Opinion Mining and Analysis

Madhavi Patel * (1147984)(GDL) Email: mpatel79@lakeheadu.ca,
Vikas Patel (1141148)(GDL) Email: patelv175@lakeheadu.ca,
Tirth Patel (1143729)(GDL) Email: tpatel16@lakeheadu.ca,
Priyankkumar Patel (1150518)(GDK) Email: ppatel92@lakeheadu.ca

Department of Computer Science, Lakehead University, Thunder Bay, Ontario

Abstract

Cloud computing is a concept which provides on-demand availability of computer system resources, data storage and processing power, without the user having to manage them directly on a pay-per-use basis. This project is based on Amazon Personalize working on Amazon Web Service cloud using Natural Language Processing and Machine Learning for movie recommendation system. In this project, we learn Amazon Personalize which helps us to create real-time personalized movie recommendations in AWS. Amazon Personalize is a fully managed machine learning service that enables developers to build customized recommendation systems. Here we will build, train, and deploy Amazon Personalize recommendation model (a solution version) with the AWS Management Console or programmatically by using the AWS SDK for Python. We used the AWS SDK for Python (Boto3) to create, configure, and manage AWS services, such as Amazon Elastic Compute Cloud (Amazon EC2) and Amazon Simple Storage Service (Amazon S3). The SDK provides an object-oriented API as well as low-level access to AWS services. Hereby after implementation and development we have evaluated a solution and created a campaign to make recommendations for our users.

Index terms- Cloud computing, Amazon Web Services, Sentiment Analysis, Natural Language Processing, Machine Learning, Amazon Personalize, Amazon Simple Storage Service (Amazon S3).

1 Introduction

Everyone loves movies, regardless of age, gender, race, colour, or geographic location. We are all connected in some way through this amazing medium. However, the most interesting thing is how unique our choices and combinations are in terms of movie preferences. A recommendation system is a system that predicts or filters preferences based on user choices. Recommendation systems are used in various fields, including movies, music, news, books, research articles, search queries, social tags, and general

products. This filtering strategy is based on the data provided about the project. The algorithm recommends products that are similar to products that users have liked in the past. This similarity (usually cosine similarity) is calculated based on the data we have about the item and the user's past preferences.

There are several methods to employ systems on AWS. Amazon Personalize, offers a completely managed solution for this. Amazon Personalize is a service that focuses in building recommend system solutions using artificial intelligence and machine learning. It analyses the data automatically, selects features and algorithms, optimises the model depending on your data, and then deploys and hosts the model for real-time recommendation inference. To create a suitable sentimental analysis tool, we must first understand how to extract data from various online sites, and then use an appropriate algorithm to analyse the polarity (positive or negative) of the retrieved data. The data examined can be utilised to make changes or as a guide for others.

2 Problem Statement

Recommendation of new material to consumers is a top goal for digital entertainment. It is generally done based on past customer feedback. The most important need for a movie recommendation system is that it must give users with recommendations for films that are similar to their tastes. Our objective is to evaluate user feedback and make fresh movie recommendations based on the results. It is now possible to work on this solution in this era because of Machine Learning. Machine Learning algorithms makes it easy to analyze all past data by training and give predictions that are useful.

3 The proposed method

As collaborative filtering approaches give the most accurate predictions, most recommender systems employ them to forecast user requirements. Rather of utilising collaborative filtering approaches, many academics are now focusing on creating alternative methods to increase accuracy. Here we deployed a similar system on AWS cloud which uses Machine Learning algorithms to predict movie recommendations for users on the basis of their past reviews.

So in this project, we learn Amazon Personalize which helps us to create real-time personalized movie recommendations in AWS . Amazon Personalize is a fully managed machine learning service that enables developers to build customized recommendation systems. Amazon Personalize makes it easy for developers to build applications capable of delivering a wide range of personalization experiences, including specific product recommendations, personalized product re-ranking, and customized direct marketing. Amazon Personalize does not require machine learning experience. We can build, train, and deploy Amazon Personalize recommendation model (a solution version) with the AWS Management Console or programmatically by using the AWS SDK for Python.

Major steps are:

• Format input data and upload the data into an Amazon S3 bucket, or send real-time event data.
• Select a training recipe (algorithm) to use on the data.
• Train a model (called a solution version in Amazon Personalize) using the recipe.
• Deploy a campaign to make real-time recommendations or a batch processing job for batch recommendations.

4 Implementation and development

4.1 Backgroung and setup

Cloud machine learning platform, enables developers to create, train and deploy machine learning models in the cloud. To train a solution for movie title suggestions, we utilised Amazon Personalize, and to prepare the data, we used the AWS SDK for Python. We developed a solution and a campaign, and then used Amazon Personalize to deploy the recommendation algorithm. Amazon Personalize makes suggestions using a machine learning model that has been trained on our data. The data used to train the model is kept in a dataset group of similar datasets. Each model is built using a recipe that includes an algorithm tailored to a certain use case.

We have created an Amazon SageMaker notebook instance and attach the appropriate policies required in this lab for our SageMaker role. Finally, we create the Jupyter notebook and save it as our project name. To execute Amazon Personalize jobs in SageMaker, we need to attach the appropriate IAM policy to this role. For this step, we attach the IAMFullAccess and AmazonPersonalizeFullAccess AWS managed policies to the SageMaker IAM role.

4.2 Download and prepare dataset

Our Amazon Personalize model is trained on the MovieLens Latest Small dataset that contains 100,000 ratings and 3,600 tag applications applied to 9,000 movies by 600 users. The MovieLens dataset is curated by GroupLens Research. Dataset stores historical and real-time data from interactions between users and items. In this part, we downloaded our dataset, inspected the dataset, then created the dataset group and schema for this project.

Database Description: Small: 100,000 ratings and 3,600 tag applications applied to 9,000 movies by 600 users.
Last updated 9/2018.
Features : userId, movieId, rating, timestamp, imdbid, tag, userid, titles, genres
README.html; ml-latest-small.zip (size: 1 MB)

Here we defined two variable in the dataset to filter out unliked movies and better simulate data gathered by a video-on-demand (VOD) platform. Also we created the dataset group, schema and intersections dataset that is used to train Amazon Personalize model.

4.3 Preprocessing of dataset

In this part we configure our Amazon S3 bucket and import the data into the program. Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry leading scalability, data availability, security, and performance.

Here, we configure the Amazon S3 bucket policy so that Amazon Personalize can read the content of our S3 bucket. To import the dataset into Amazon Personalize, we can create the import job that loads the data from Amazon S3 into Amazon Personalize to use for our model.

4.4 Create and evaluate a solution

Configuring the solution allows you to customize the solution parameters and recipe-specific hyperparameters so the model meets your specific business needs. In Amazon Personalize, training the model is referred to as creating a solution version. Amazon Personalize generates metrics that you can use to evaluate the performance of the model before you create a campaign and provide recommendations.

4.5 Create a campaign and get recommendations for users

A campaign is used to make recommendations for our users. In this part, we created a campaign by deploying our solution version. To deploy a solution version, we have created a campaign in the console or by calling the CreateCampaign API and then we will choose the version of the solution to use. We have created the campaign and wait for the campaign status to be active. Then, we use our campaign to get real-time movie title recommendations for users from Amazon Personalize

5 Conclusion

Amazon Personalize makes it simple for developers to create apps that provide a variety of personalisation experiences, such as customised product suggestions. So here using Amazon Personalize we have created a real-time personalized movie recommendation in AWS.Format input data and upload the data into an Amazon S3 bucket, or send real-time event data.We created the Amazon S3 bucket to stage your dataset, created the appropriate policies and roles for Amazon Personalize to access the data. We Selected a algorithm to use on the data. We evaluated the solution version for model performance and created a campaign by deploying our solution version. Finally, We

deploy a campaign to make real-time recommendations or a batch processing job for batch recommendations.

6 Contribution

All the team members have mutually worked on this project. The individual roles and contributions of the team members are described below.

- Vikas Patel (Research Expert and Database Analyst)
  Vikas worked on background and setup for the project. Taking reference from the past related papers and projects he contributed towards finalising the cloud service to be used with Aamazon personalize. Also contibuted towards getting the dataset to be worked on.

- Tirth Patel (Lead Programmer 1)
  Tirth worked with Vikas on getting the dataset ready to be worked on. He first worked on configuring of dataset and then preprocessing of dataset to make it ready to use for model.

- Madhavi Patel* (Team Leader) and Priyank Patel (Lead Programmer 2)
  Madhavi and Priyank both worked very closely for creating and evaluating an Amazon Personalize solution. They worked on creating a campaign to deploy the solution version and get recommendation of movies for users from their previous reviews.

7 Results

By using the Linear Regression model we got around 85% Accuracy. This is mainly because we just used 2 classes for the output i.e. Positive and Negative. If we add more classes to our model, training the model will become difficult so the accuracy may decrease. This discussion is continued later in the Future work section.

```
# Build a map to convert a movie id to the movie title
movies = pd.read_csv(dataset_dir + '/movies.csv', usecols=[0,1])
movies['movieId'] = movies['movieId'].astype(str)
movie_map = dict(movies.values)

# Getting a random user:
user_id, item_id = interactions_df[['USER_ID', 'ITEM_ID']].sample().values[0]

get_recommendations_response = personalize_runtime.get_recommendations(
    campaignArn = campaign_arn,
    userId = str(user_id),
)
# Update DF rendering
pd.set_option('display.max_rows', 30)

print("Recommendations for user: ", user_id)

item_list = get_recommendations_response['itemList']

recommendation_list = []

for item in item_list:
    title = movie_map[item['itemId']]
    recommendation_list.append(title)

recommendations_df = pd.DataFrame(recommendation_list, columns = ['OriginalRecs'])
recommendations_df.head()
```

Figure 1. Result code

Through out the project we use only and only aws services to gain more experience in cloud programming. In result we have used more than 80 $ credits. Our project traing cost is 4 $ per day. We used S3 bucket to store our databases and train result which reduce the cost and help us to decrees to 0.5 $/perday$.

The picture provided below is a snapshot of the UI we created. The link in the URL can be used by anyone (Only when the service is up and running) to check and run the model.

To exemplify the accuracy of our model, we have chosen two comments of customers from the Dataset, and ran it through our model, and accurate results were produced.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 371 entries, 0 to 370
Data columns (total 1 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Movies_name  371 non-null    object
dtypes: object(1)
memory usage: 3.0+ KB
None
```

	Movies_name
0	Toy Story (1995)
1	Heat (1995)
2	Casino (1995)
3	Twelve Monkeys (a.k.a. 12 Monkeys) (1995)
4	Seven (a.k.a. Se7en) (1995)

Figure 2. Watched Movies for user 339

Recommendations for user: 339

	OriginalRecs
0	27 Dresses (2008)
1	Get Smart (2008)
2	Kagemusha (1980)
3	AVP: Alien vs. Predator (2004)
4	A-Team, The (2010)

Figure 3. Result Recommendation for user 339

We also tested the model using some other example inputs which were not present in the dataset, and as per the accuracy mentioned above, we got very accurate results. The only flaw here is that the model cannot predict neutral comments or reviews provided by the customers because it is not trained for a neutral sentiment class.

8 Future Work and Discussion

We are planning to work on the UI and create a more interactive interface using the Flask framework. Due to time constraints, we have just worked on two output classes (Positive and Negative) to our model, but in future we are planning to add a third output class "Neutral", because a customer can also have neutral opinion. Model accuracy can be improved by using extensive deep neural networks architectures with contextual embedding.

In current scenario our project use 4$ per day. So, we will try to reduce cost by using less AWS services.

We will also try to Implement UI which can help user to chose more movies option dynamically and send their recommendation to their email by AWS SNS service.

References

[1] Hössjer, O., Rousseeuw, P.J. and Croux, C. (1996), Asymptotics of an estimator of a robust spread functional. *Statistica Sinica*, 6(2), 375–388.

[2] Therneau, T., and Grambsch, P.M. (2000) *Modeling Survival Data: Extending the Cox Model*. New York: Springer-Verlag.

[3] https://aws.amazon.com/getting-started/hands-on/

[4] Jin, R., and L. Si. 2004. "A Bayesian Approach Toward Active Learning for Collaborative Filtering." In Proceedings of the 20th Conference on Uncertainty inArtificial Intelligence, 278–285. Arlington, Virginia, United States: AUAI Press.

[5] Su, X., and T. M. Khoshgoftaar. 2009. "A Survey of Collaborative Filtering Techniques." Advances inArtificial Intelligence, article 421425, 2009:1–19, London, UK: Hindawi Publishing Corporation.

[6] Yu, K., A. Schwaighoferi, and V. Tresp. 2002. "Collaborative Ensemble Learning: Combining Collaborative and Content Based Information Filtering via Hierarchical Bayes." In Proceedings of Nineteenth Conference on Uncertainty inArtificial Intelligence, 616–623. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

[7] Cold-Start Recommendation Using Bi-Clustering and Fusion for Large-Scale Social Recommender Systems Authors Daqiang ZhangChing-Hsien HsuMin ChenQuan ChenNaixue XiongJaime Lloret