Data Science Project Report

1. Principal Investigators

Matthew DiDomizio (didomizm1@newpaltz.edu)

Maranda Dominguez (domingum6@newpaltz.edu)

1.1 Individual Contribution Breakdown (list the percentage)

Task	Matthew	Maranda	Total
Introduction	50%	50%	100%
Background	50%	50%	100%
Implementation	50%	50%	100%
Experiment Results and Discussion	50%	50%	100%
Conclusion	50%	50%	100%
Other contribution and explain	N/A	N/A	100%

2. Title of Project

Anime Recommendation System

3. Mentoring

Professor Min Chen, Department of Computer Science, SUNY-New Paltz

chenm@newpaltz.edu

4. Introduction

4.1 Project Motivation

The motivation for this project was our common interest in anime. With the completion of this project, we sought to obtain anime recommendations of our own based on anime series we previously watched.

4.2 Aims and Objectives

In this project experience, we aim to create a recommendation system which is able to accurately give anime series suggestions to new users based upon their similarity to existing users within the system. In order to achieve this goal, we will implement a K-Means clustering algorithm in Python and process a large dataset using MapReduce and Hadoop. Once the system is built and the correlation between different users is established, we intend to test the system to ensure both its proper functionality as well as its consistent precision in predicting which series a user would likely enjoy.

5. Background/History of the Study

A recommender system is a construct that seeks to predict user preferences. These systems ultimately help users discover products and content they might not have otherwise found on their own. Recommender systems are trained to understand the preferences, previous decisions, and characteristics of people and products using data gathered about their interactions. This can be done using various techniques, including collaborative filtering and content-based filtering. A content-based filtering system uses the attributes or features of an item. This type of recommender system is based on similarity of item and user features; given information about a user and items they have interacted with, it can predict future interactions. With content-based filtering, it is highly probable that a user will get recommended items with the same attributes or features as their past preferences. On the other hand, a collaborative filtering system uses similarity of user preference behavior based on past interactions in order to predict future interactions. The concept of collaborative filtering is that if users have made similar decisions in the past, like an anime choice, it will recommend a similar user that same anime choice.

Machine learning algorithms are used to build such recommendation systems based on users' previous behavior or interests. Machine learning algorithms learn from users' behaviors and categorize them into groups which contain users of similar behaviors. Various types of machine learning algorithms have been used, such as clustering algorithms, deep learning models, classification algorithms, and matrix factorization. Though, the type of algorithm to be used fully depends on whether the available data is labeled or not. If the data is labeled, one must use a supervised machine learning algorithm such as naïve Bayesian classifier. Conversely, if it is unlabeled, an unsupervised machine learning algorithm like K-means clustering must be used. These machine learning algorithms can be used in conjunction with the Hadoop MapReduce function in order to build a fully functioning recommender system.

6. Approach and Implementation

The K-Means clustering algorithm was put to use when working towards an implementation of this recommendation system. This unsupervised machine learning algorithm was adapted for use with MapReduce so that it could be attributed to usage with Hadoop. This would allow for large sets of data to be processed in a distributed system, thus taking advantage of the processing power of multiple computers to solve the problem at hand.

The first step of the overall process after deciding upon the topic and objective of this project was to source the data. A sizeable and well-structured dataset was curated from Kaggle for use with this system, and using modules such as Pandas, the data was explored, cleaned, and transformed. Once the data was condensed into a proper form for clustering and building user profiles, the number of clusters, K, was determined and initial random centroids of these clusters were picked. The data in this form was visualized with heatmaps to account for its multidimensionality.

At this juncture, with the prerequisites for the MapReduce algorithm to be run completed, the next step was to construct the algorithm itself. Both the map and reduce step were programmed in Python, and following with the sequence of steps required for K-Means clustering, such steps were separated into either the map or reduce portion. In particular, the map step involved taking an input of initial centroids and fitting them into Python data structures. The centroids were then used alongside the full dataset to calculate the Euclidian distance between each plotted datapoint and each centroid. Datapoints were assigned to the cluster whose centroid was nearest, and each completed cluster was ultimately sent to the reducer for further processing.

The reduce step was made to take in the clusters through standard input and calculate the mean of each coordinate within all plotted points of a cluster. This yielded the overall average point of a cluster, which would then be used as its new centroid. These new centroids were then fed back into the mapper for multiple iterations, and the process was completed once the centroids would no longer change upon the completion of an iteration.

With the data fully processed and the true clusters found, test input user data was used to verify the precision of predictions by the system. A hypothetical user’s ratings for series and thus genres were used to associate them with a certain cluster, and the satisfactory nature of the final results was examined in detail.

7. Experiment Results and Discussion

After completing the recommendation system, results of the experiment were determined through a testing stage. This was carried out by setting up hypothetical scenarios in which a new user would have a single series or a minimal set of series which they had rated. These ratings could be high, low, or a combination of both, and in all cases the test user ratings were used to create a profile of the user. This profile would be compared to the existing plotted user profiles in the final clusters and the new user would be placed into one these clusters by finding which of the Euclidian distances between their plotted point and each of the centroids was the smallest.

Upon finding the proper cluster for a new user, a random user within that cluster would be used as a match, and series that they rated highly would be prioritized. A user would not be picked for comparison if they did not generally rate any series high. These highly rated series which the new user had not seen before would serve as the final recommendation. The true validity of recommendations is greatly subjective, thus testing with more real users would be ideal, however it can be said that our own personal recommendations were curated well.

Another method for determining the series to be recommended was considered in which the most commonly watched and highly rated series within an entire cluster would be selected, and while this would perhaps avoid the issue of outlier series being recommended when using a single random user, this method runs the risk of poor performance with little gain; there is potential for much redundancy and over-complexity in trying to find the “perfect” series to recommend. True perfection cannot be attained in this experiment because of the opinionated nature of recommendations, and furthermore, there is already a high likelihood that the most commonly watched and highly rated series within a cluster would also be picked at random, thus there is not a need to verify this point through extensive data processing.

8. Conclusion

Ultimately, it was found that the creation of the proposed Anime Recommendation System was successful. New users who feed their preferences into the system will likely find that series which are recommended to them are those that they will enjoy. Throughout this project, a greater understanding of data analysis, data munging, machine learning, distributed processing, as well as data science as a whole was gained. In the future, this system can be expanded upon and further enhanced for wider and even more precise usage.

9. References

Asad, Syed Muhammad. “AI Movies Recommendation System with Clustering Based K-Means Algorithm.” *Medium*, Medium, 19 Aug. 2020, asdkazmi.medium.com/ai-movies-recommendation-system-with-clustering-based-k-means-algorithm-f04467e02fcd.

Dey, Victor. “Collaborative Filtering vs Content-Based Filtering for Recommender Systems.” *Analytics India Magazine*, 25 Aug. 2021, analyticsindiamag.com/collaborative-filtering-vs-content-based-filtering-for-recommender-systems/.

Dwivedi, Rohit. “What Are Recommendation Systems in Machine Learning? | Analytics Steps.” *Analytics Steps*, 16 Apr. 2020, www.analyticssteps.com/blogs/what-are-recommendation-systems-machine-learning.

Grover, Prince. “Various Implementations of Collaborative Filtering.” *Towards Data Science,* Towards Data Science, 28 Dec. 2017, towardsdatascience.com/various-implementations-of-collaborative-filtering-100385c6dfe0.

Rocca, Baptiste. “Introduction to Recommender Systems.” *Medium*, Towards Data Science, 2 June 2019, towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada.

Roy, Abhijit. “Introduction to Recommender Systems- 1: Content-Based Filtering and Collaborative Filtering.” *Medium*, 31 July 2020, towardsdatascience.com/introduction-to-recommender-systems-1-971bd274f421.