ANALYSING WATCH TIMES OF HOTSTAR USERS AND PREDICTING THE USER SENTIMENT

Omkar Khanvilkar
osk8557@rit.edu
RIT ID: 347001066

Pranav Nair
pn1922@rit.edu
RIT ID: 359006909

Sujan Dutta
sd2516@rit.edu
RIT ID: 782000242

Varun Tandon
vt9438@rit.edu
RIT ID: 572002978

1 DEVELOPMENT OF QUESTION/HYPOTHESIS

Over the past decade, we have seen an increase in viewership for streaming services across the globe. Sites like Netflix, Hulu, and Hotstar dominate the streaming market with the content that they provide. The common denominator across these sites is the userbase and the feedback they provide to get a better experience. Hotstar is a platform that has over 100 million user count, and more than 35,000 hours of content accessible over a variety of genres. As is the case with most major streaming services, using the sheer scale of users and content to create tailor-made recommendations and content for users generates a lot of value for Hotstar. Thus comes our problem statement. How does a service like Hotstar capture its users' sentiments? Regular questioning often breaks immersion and can cause a biased answer. To approach this problem, we look at an alternate method, deducing the user sentiment by analysing the user behaviour. The data the user generates is watch time in seconds, over multiple parameters. The users were also asked if they had a positive or a negative sentiment based on their overall experience. Our goal is to find the correlation between these parameters and the sentiment.

2 DATA RESEARCH

Upon searching for relevant data for the proposed problem, we found a dataset on HackerEarth [1] that comprises of two files: *train_df.json*, which consists of watch times and the sentiment for 200,000 users and *test_df.json*, which consists of the sentiments for 100,000 users.

The data itself consists of 7 unique columns: id: A unique identifier variable. titles: Titles of the show watched by the user along with the watch time. genres: Names of the genres watched by the user along with the watch time. cities: Names of the cities the user was present in along with the watch time. tod: The time of the day in hours the user began watching along with the watch time. dow: The day of the week the user began watching along with the watch time. segment: The sentiment of the user (as positive or negative.)

3 LITERATURE REVIEW

The topic of customer (user) classification and segmentation has gained a lot of research attention in the last two decades because of the emergence of multiple OTT platforms. It all started with the release of the famous Netflix dataset [2] that contains 100M movie ratings in 2006. Since then many researchers have used this dataset to build movie recommendation systems and customer segmentation models. However, our dataset is a much newer one and it comes from Hotstar, a different OTT platform. While doing literature survey we found that there exist many works that use techniques like matrix factorization, nearest neighbors approach [3], clustering approaches [4] to recommend movies (or products) and perform customer segmentation. In this work, however, we are treating the problem as a classification problem as the data contains a label denoting the segment (positive or negative) corresponding to a customer. Treating the problem as a classification task will allow us to use the popular tree-based algorithms like Random Forest and XGBoost.

4 ANALYSIS: STRATEGY

There were two main issues faced with this data set. Primarily, the data was purely categorical in nature, in the form of "value:watchtime", separated by commas. Preprocessing this kind of data tends to be pretty time consuming and heavy on the computational load. To resolve this, the preprocessing was done on the json files itself, that allowed the local systems to utilize the most of their computational power without wasting computational time. The preprocessing itself was to create a function that parsed through the row of a single user, separate the value and the watch time, and add the value as a column with the user as the row, and the watch time as the data for that cell. With each user parsed, common columns were merged together, and a sparse data frame was created, with every single value type as a new column. The second issue faced was the heavy class imbalance. The negative sentiments overshadowed the positive sentiments with a 92:8 ratio. With such a heavy imbalance, the models were sure to overfit and predict the incorrect class. The solution proposed for this issue was to oversample the positive class and undersample the negative class. The oversampling technique used here was SMOTE. After doing so, the ratio of the negative class to the positive class reduced to 5:2. The main features used for the final dataset were: cities gernes titles dow tod

Upon further analysis, the following assumptions were made based on EDA: The cities with users of positive sentiment did not follow the trend of the cities with users of negative sentiment. Furthermore, there were more cities with negative sentiment than positive. Cricket was the highest watched genre, but there was no other sport in the top ten most-watched genres. The most-watched hour of the day was 9pm, followed by 10pm The most-watched day of the week was Saturday, followed by Friday. This indicates that the users tend to watch on weekends rather than weekdays. This also corroborates with the time of the day analysis; Users watch on weekend nights, but not on Sunday nights.

5 ANALYSIS: CODE

After preprocessing, the final number of columns ended up being 1448. To reduce this number, feature engineering was performed on the dataset. Each feature was approached in order with the following results.

Genres: All sports (except for cricket) were combined into a single column, with the watch times being summed. A *genre_count* feature was added to keep track of the total genres watched by the user.

Cities: Only the top 15 cities based on watch time were added to the final dataset.

dow: Two new columns were added - *total_watch_time*, to keep track of the user's total watch time of the week, and *dow_count*, to keep track of the number of days the user has watched for.

tod: Only the top 5 hours based on watch time were added to the final dataset. An additional column for the total hours watched by the user.

titles: Only the top 24 shows was considered except the cricket matches. An additional column for the total number of titles watched by the user.

The final data frame consisted of 78 unique features excluding the target after the engineering. The over and under sampling was done to this data frame. Next, the four models were tested on this data frame. Each model was trained without any tuning and then validated on a validation data set. Then the model underwent hyperparameter tuning and was validated on a validation data set. The final results are shown in the following table.

Model	Default Hyperparameters	Tuned Hyperparameters
Logistic Regression	0.7802	0.78029
Decision Tree	0.6204	0.6682
Random Forest	0.8068	0.8056
XGBoost	**0.80**	**0.8119**

Table 1: AUC ROC score comparison

Thus, XGBoost was the model chosen to predict on our test data as it had the highest accuracy amongst the four. The next step was to deploy the model, using a python library, "streamlit". This is an open-source framework used for building web apps for machine learning and data science models. The UI of the web app allows input of the user number, and can predict the sentiment of the user.

6 WORK PLANNING AND ORGANIZATION

The entire workload was split equally amongst the four team members. Each part of the project was compartmentalized and broken into four equal parts, i.e., the preprocessing, EDA, and model engineering was tackled by all four members.

7 IMPROVING TEAMWORK AND COLLABORATION

In a collaborative project like this, it is very important to follow certain steps to maintain productivity. We decided to meet biweekly on zoom to update individual progress and discuss the future direction of work. We also used to exchange interesting ideas and results over slack. To work in parallel, we need to make sure that everyone has the access to the same version of the data. To ensure data consistency, we first preprocessed the data on a single machine and then distributed the copies among us. This project involved a large amount of data so it was not feasible to train our models every time. To overcome this issue we decided to save the models for later use and this made us much more productive.

8 INDIVIDUAL CONTRIBUTION

As mentioned above, the entire work was divided equally among the contributors. In the exploratory data analysis section, I took care of the city feature. First, I found the cities with the most watch times. Then I created visualizations to reveal how the distribution of city counts varies with respect to the sentiment of the users.

While performing feature engineering, I dealt with the genres column. This was significantly different than the other features because in the original data we had 35 different genres but some of them had very few occurrences. To reduce the sparseness, I decided to combine all the sports (Hockey, Tennis, F1, etc.) into one genre. But I kept Cricket as a separate genre because it is the most popular genre on Hotstar. I also came up with a new feature called *genre_count*. It is nothing but the number of genres that a user has watched in their entire time on the platform. After doing feature engineering, I had 23 features extracted from the genre column.

As described earlier, the dataset was heavily imbalanced so we opted for some specialized sampling methods. I employed SMOTE (Synthetic Minority Over-sampling Technique) to generate new samples of the minority class (positive segment) and randomly undersampling the majority class (negative segment).

In the model building portion, I worked on the XGBoost model. At first, I trained an XGBoost model with default parameters to obtain a baseline. Then I performed hyperparameter tuning with GridSearchCV to obtain a better model. This model produced the best ROC AUC score on validation data hence we decided to make this model our final model.

References

[1] "Disney+ hotstar customer segmentation data." https://www.hackerearth.com/problem/machine-learning/predict-the-segment-hotstar/. Accessed: 2021-12-09.

[2] J. Bennett, S. Lanning, *et al.*, "The netflix prize," in *Proceedings of KDD cup and workshop*, vol. 2007, p. 35, New York, NY, USA., 2007.

[3] G. Takács, I. Pilászy, B. Németh, and D. Tikk, "Matrix factorization and neighbor based algorithms for the netflix prize problem," in *Proceedings of the 2008 ACM conference on Recommender systems*, pp. 267–274, 2008.

[4] P. Q. Brito, C. Soares, S. Almeida, A. Monte, and M. Byvoet, "Customer segmentation in a large database of an online customized fashion business," *Robotics and Computer-Integrated Manufacturing*, vol. 36, pp. 93–100, 2015.