**Building a Recommender System for Vacation Rentals in Canada Using Natural Language Processing**

**Project Report**

**Frankline Ononiwu**

Problem Statement

The project is an original end-to-end project originated by the author. Vacation rentals are popular for weekend vacations in Canada. Most of the booking for these rentals are done on booking sites and the consumers depend on these booking sites for recommendation of vacation rentals that suits their needs. The accuracy of these recommendation engines built into rental sites will determine if a consumer gets a desired vacation rental. In this case study, we use a popular booking site ‘tripadvisor.com’ to build a recommendation engine for vacation rentals in Canada. Dataset of customer reviews serves as the training dataset, and using Natural language Processing algorithms, these reviews are used to build a recommender engine. We applied Deep learning and classification modeling to improve the accuracy of our Recommender system.

Web Scraping

To obtain original data to be used for building the recommender system, I scraped the website [www.tripadvisor.ca](http://www.tripadvisor.ca) with the search keyword as ‘Vacation Rentals’ on the 10th of September 2019. The search yielded about 870 results. The results were then scraped from the website using selenium and python package. The contents scraped from the website for each rental included;

**Title, Price, Rating, Reviews, Review URL, Location**

The process of scraping is explained in the web scraping code attached to this GitHub page. The contents obtained from the web search was dependent on the date. However, we believe that this search represents to a higher degree the vacation rentals currently available in Canada. This informs the choice of using TripAdvisor as the booking website of choice for the data, since it has been highly rated as one of the best booking websites.

Moreover, the purpose of the whole study is to build a recommender system, which I believe the modelling can be applied to any dataset.

Data Preprocessing

Data obtained from web scraping was messy and had to be cleaned. Using the excel file extra columns was created to separate the city and province of the rentals. Then the bubble rating was converted to the Rating of the rentals from reviews. Further preprocessing was carried out in the pandas package of python.

**The Dataset:**

print ('Unique counts:’, df.nunique())

Unique counts: Title 846

Bubble\_Count 7

Rating 7

Review\_Count 70

Review\_counts 69

City 327

Province 12

Reviews 570

The title is the most with unique values and covers the number of rows of the dataset. This means that there are 846 different rentals retrieved from the scraping.

**Preprocessing methods:**

**Removal of extra columns not required for the analysis and modelling** – The bubble count and Review were removed from the dataset since they would have no impact on the modelling.

**Remove new line characters and numbers** – New line characters and numbers were removed from the Review column to work with only the texts.

**Handle missing values** – missing values were observed for the Rating, Review counts and the Reviews columns. The missing values for the Rating and Review count was filled with the mean of the columns. This was used to avoid skewing the data with a median value since most of the rating value are skewed. This also led to an additional category for the rating which is our target variable.

After that, the remaining missing values were removed but dropping all rows with a missing value to take care of the missing values in the Review column.

Exploratory Data Analysis

Exploratory data analysis was performed primarily with the pandas python package. The aim of the exploratory data analysis was to understand the dataset and to explore relationship between the variables before modelling. Observations in trends and relationships can be used to formulate hypothesis before modelling is carried out. Exploratory data analysis was performed in two stages. The first is the data wrangling which provided a summary of the dataset and its structure. The second is the actual analysis which included grouping analysis and visualization of variable relationships using the matplotlib python package.

Modeling

The modelling was performed with both traditional classification models and a simple deep neural network model. The purpose of this modeling was to predict and possibly improve the accuracy of the Recommender System.

**Classification Models**

The modeling was performed as a classification model since we are trying to determine recommendation for the rentals. The project is a multi-label classification problem with the classes obtained after encoding the ratings from the rental reviews. The encoding produced seven(7) different classes. (Five(5) classification algorithms were used for the modeling to determine which algorithm produces the best score and is a better fit for the modeling problem. Below are the algorithms used and their respective model score;

Logistic Regression **(0.6643356643356644)**

Random Forest Classifier **(0.6643356643356644)**

Adaboost Classifier **(0.6923076923076923)**

KNearest Neighbor Classifier **(0.6013986013986014)**

Support Vector Machine Classifier **(0.5874125874125874)**

The Adaboost model showed the higher accuracy probably because the boosting improved the accuracy of the model.

**Deep Learning Model**

In addition to the classification models, an attempt was made to apply deep learning model for the review classification modeling. A simple deep learning model was used with only three layers and a relu activation function. The modeling was done with keras on a Tensorflow backend. The model was compiled with a stochastic gradient.

A screenshot of a cell phone

Description automatically generated

The accuracy of the deep learning model increased with increasing the epochs and the batch size. The accuracy was not much different from the accuracy for the classification models. Likely due to the lack of much dataset.

A Recommender Engine

The recommender engine was built with Cosine Similarity model which is explained in the appendix section. Using the reviews from the customer term frequency was generated and applied to cosine similarity models for recommending vacation rentals based on customer reviews.

**Term Frequency Vectorization**

Term frequency vectorizer was used to vectorize the reviews and produce a term frequency matrix. Based on the review matrix, rentals with similar reviews can be matched with the similarity matrix.

**Cosine Similarity**

The model built with this allows us to match rentals based on the similarity of their reviews. The reviews are sorted based on scoring and top 3 most similar scores of reviews corresponding to similar rentals are displayed.

Conclusion

In this project, we attempted to build a recommender system from scratch to recommend vacation Rentals in Canada based on customer reviews. We chose a website for reviews of vacation rentals and scraped the data from the website to get the needed parameters for analysis. We were able to successfully clean the data and perform exploratory data analysis before applying some machine learning and deep learning algorithms to model the data. Finally, we applied a matrix factorization to build the recommender engine which was able to correctly predict and recommend the vacation rentals based on user ratings and reviews. We experienced the challenge of getting enough data for the project based on the fact we had limited vacation rentals in Canada. However, we wanted to ensure the integrity of our recommender engine.

Our exploratory data analysis revealed that different factors like location and type of rentals affects the ratings the rentals received this was done without putting the reviews into consideration but we believe that the ratings are a reflection of the reviews. Then we used different modelling to predict the ratings of the vacation rentals. Though the deep learning model gave a high model score as the epochs increased, it seems likely that the accuracy of the model was hampered by the small size of the data.

In order to improve some of our models, we could do a few things. we could have used

GridSearchCV instead to test more hyperparameters at the expense of computational time. We

could have used our random forest model to do some feature selection, and used the most

important features in our both our logistic regression model and boosting model. Finally, we

could have tried XGBoost, a different boosting algorithm which has been shown to typically

have higher performance than other boosting algorithms. Nonetheless, we have a somewhat

accurate model that does not overfit.

Appendix

Explanation of Support Cosine Similarity

Explanation of Random Forest Classifier

Works Cited