YELP - Business closure prediction

12/02/2022

Aakash Sundaresan

Ankit Nitinkumar Bhawsar

Serah Varghese

Shravani Nimbolkar

Frank Lin

Overview

YELP is an American company that primarily focuses on publishing public-sourced reviews about restaurants, cafes and other businesses. We have a dataset which consists of data from YELP about various restaurants. Using this data we try to make meaningful inferences on whether a restaurant is open or not, given a certain set of attributes.

Goals

1. Preprocess the data for EDA
2. Perform EDA to extract meaningful insights and observe patterns in the data
3. Process the data in order to get clean uniform datasets
4. Run various Machine Learning models on the clean data
5. Make predictions on whether a restaurant will be open or not in the future

Specifications

Initially the dataset consists of 14 variables- 13 independent and 1 dependent variable. Below is the definition of each one of them:

- Business_id: A Unique string id given to every business
- Name: Name of the business
- Address: Address of the given business
- City: Name of the city where the business is located
- State: Name of the state where the business is located
- Postal_code: Postal code of where the business is located
- Latitude and Longitude: latitude and longitude of the business
- Stars: Average stars given by the public for that business
- Review_count: Number of reviews received by the business
- Attributes: Dictionary containing attributes of the business. For example, does the business accept appointments only, accepts credit cards etc
- Categories: Dictionary describing the categories of the business ie what the business fundamentally does. Example: doctor, shipping, restaurants etc
- Hours: Dictionary consisting of the working hours and days of the business
- Is_open: If the business is open : 1, or it is closed: 0

Milestones

I. Importing Dataset

We imported the dataset into our python notebook using pandas. The data had a shape of (150346,13). We also found that the data is not balanced and the classes had a 70-30 imbalance.

II. Text Preprocessing

A. Geolocation features (Longitude and Latitude)

We performed Kmeans clustering on the longitude and latitude features to observe different clusters. Using the elbow method we found the optimal value of the number of clusters to be 4. We then assigned different cluster labels to all the businesses depending upon their geolocations.

B. Text Features (Attributes and Categories)

We had 2 columns Attributes and Categories which have text in them. In order to run models with these columns, we processed them using NLTK. These columns were a list of dictionaries, so the first step we did was to make them a sequence of words. Then, we used CountVectorizer in order to convert the words into integers based on their frequency. We applied this on Attributes and Categories.

C. Datetime (Hours)

On the hour column which was a datetime feature, we split into 3 columns for each day - start, end and total hours using the datetime package. After finishing all the data processing work mentioned above, our data was clean and ready for modeling.

III. Exploratory Data Analysis (EDA)

In this part we plotted various graphs to observe patterns in the dataset and to answer a few questions like:

- What is the count of open or closed restaurants across different states?
- What is the relation between Star rating and Review counts?
- What is the count of businesses that are opened or closed based on star rating?
- What are the unique business categories and their counts?

IV. Model Building

A. Logistic Regression

Since all the preprocessing is done, we started running models on the processed data. The baseline model i.e. predicting all as 1, has an accuracy of 79. We then did a 70-30 split on the dataset and then ran a logistic regression model. We ran a logistic regression model and got accuracy, TPR, and FPR of 0.8204, 0.9626, and 0.7309 respectively.

B. Linear Discriminant Analysis

After running the logistic model, we also applied the linear discriminant analysis model. The accuracy, TPR, and FPR on the test set were 0.8268, 0.9585, and 0.6842, respectively, which indicates a slight increase in accuracy.

C. Decision Tree Classifier

The next machine learning model we ran was the decision tree classifier. Initially, we did cross validation on the training set and searched for the optimal value of cost complexity parameter (ccp_alpha). The best ccp_alpha was 0.00005. Then, we applied the decision tree classifier model with the given parameters to the testing set. The accuracy, TPR, and FPR on the test set were 0.8351, 0.9614, and 0.6548, respectively.

D. Random Forest

We ran the random forest model using the default parameters. The accuracy, TPR, and FPR on the test set were 0.8386, 0.9568, and 0.6196, respectively.

E. Boosting

Next, we ran boosting models like AdaBoost, XGBoost, and CatBoost.

- AdaBoost

We trained an AdaBoost classifier which gave an accuracy of 0.8209, TPR of 0.93 and FPR of 0.6107 which is more or less similar to the Random forest classifier.

- XGBoost

The advantage of XGBoost over the gradient boosting algorithm is that it performs parallel processing at the node level, making it more effective. So we trained the XGB classifier and achieved an accuracy of 0.832 but an increased FPR of 0.7259.

- CatBoost

  The CatBoost classifier was then trained using a learning rate of 0.1, while the other hyperparameters were left at their default values. After training the Catboost classifier for 1000 iterations we got an improved accuracy of 0.8475 but an increased FPR of 0.7187 over the random forest.

CatBoost CV

Since CatBoost gave promising results we tried 10 fold cross validation on CatBoost with 5000 iterations, loss function as Logloss, learning rate of 0.1 and l2_leaf_reg of 5. The best iteration had a Logloss of 0.3653. Later, we averaged the output of all the 10 models to get the final predictions (y_pred). We achieved an improved accuracy of 0.8480, TPR of 0.9561 and reduced FPR of 0.5712.

F. Semi-supervised

Semi-supervised learning refers to algorithms that attempt to make use of both labeled and unlabeled training data. First we used logistic regression and CatBoost model to fit on the already labeled train data. Then we created a list of -1 valued (unlabeled) for each row in the unlabeled portion of the test data which is y_test. Next, we appended the test data to the train data. We then used the SelfTrainingClassifier with logistic regression and the CatBoost classifier passed as the parameter base_classifier. We observed no signs of improvement with an accuracy of 0.82 and 0.84 with semi semi-supervised logistic model and CatBoost classifier, respectively.

G. Neural Networks

The last model we ran was the Artificial Neural network. We standardized the data and did all the necessary steps in order to run an ANN model. We used 4 layers - 1 input layer, 1 output layer and 2 hidden layers. Input layer had 40 nodes while the hidden layer had 20 nodes. We used sigmoid activation functions for the hidden layers and ReLU for the output layer. Finally, we evaluated based on accuracy, used stochastic gradient descent as our optimizer and ran 7 epochs. We got an accuracy of 0.8410 on our test set.

V. Final Results

Model	Accuracy	TPR	FPR
Logistic	0.8204	0.9626	0.7309
LDA	0.8268	0.9585	0.6842
CART	0.8351	0.9614	0.6548
Random Forest	0.8386	0.9568	0.6196
AdaBoost	0.8209	0.9322	0.6107
XGBoost	0.8326	0.9767	0.7259
CatBoost	0.8475	0.9640	0.7187
CatBoost CV	**0.8480**	**0.9561**	**0.5712**
Semi Supervised Logistic regression	0.8202	0.9592	0.7187
Neural Network	0.8410	-	-

VI. Model Improvements

Using the reviews dataset that consists of customer reviews , we can perform NLP on it to get the sentiment of the user and try to enhance the performance of the model better.

Due to the large size of this dataset and processing limitations, we have demonstrated it using a subset of the large. Here, we are taking 300,000 reviews instead of the entire dataset which when combined with the business dataset results in around ~12000 different businesses.

After merging and reducing the size of the dataset, we then performed preprocessing and word embedding on the reviews to get a cleaned dataset.This was followed by K-means clustering to help get an unsupervised learning closeness score and sentiment score which was then used to get the prediction score.

This prediction score can take many class values, but, for this project, we went ahead with two classes - positive sentiment(1) and negative sentiment(0).

For every business, the average of all the prediction scores were taken and stored in a new column corresponding to its Business ID.

After adding this new column, we can see an increase in the performance of CatBoost CV.

The following results were achieved for 300,000 reviews and ~12000 unique businesses:

Model: CatBoost CV	Without NLP	WITH NLP
Accuracy	0.8161	0.8222

This can be extended to the entire reviews dataset to get better results using NLP techniques as demonstrated above.

Conclusion

Since the relationship between the independent and dependent variable is not linear we observe that the performance of CART, Random Forest, Boosting and Neural Networks are much better in terms of accuracy. This is because these models are able to identify complex relationships in the data. The data has a lot of positive classifications i.e. it is imbalanced and hence the True Positive Rate is very high for all the models. This implies that any given model has the tendency to classify data as positive and most of the errors are negative classification which are classified as positives i.e. false positives. Given the fact that the dataset is imbalanced, we found that the CatBoost CV classifier could successfully classify the labels from both the classes and struck the right balance between TPR and FPR, yielding the highest accuracy score of 0.8480 among all the models.

Appendix. Links of Coding Part

Link for the Non-NLP part: 242 Final Project

Links for the NLP part:

- Processing and Embedding
- K Means
- Predictions
- Results: Without NLP, With NLP