

Machine Learning Engineer Nanodegree

Capstone Project

Srimaya Padhi
November 02nd, 2020

I. Definition

Project Overview

The service industry is highly volatile and is driven by customer satisfaction. Nowadays most of the service sector enterprises, such as hotels, restaurants, food delivery, etc., have an online presence to publicize themselves. Potential customers often look at the experiences and reviews of previous customers for guidance. Thus, it bodes well for businesses to analyze their reviews and derive insights into the likes and dislikes of their customer base.

In this project, the customer reviews for TripAdvisor have been analyzed. The aim of the project is to classify verbose, human written customer reviews into 5 ratings based on semantic analysis of the reviews. The dataset for this project is available on [Kaggle](#). The dataset is available under the Creative Commons license.

Problem Statement

The problem being solved is a classification problem of processing customer reviews from Trip Adviser and assigning it a numerical rating of 1 to 5.

This is an Natural Language Processing (NLP) sentiment analysis problem to understand user choices. This problem is a classification problem which can be solved using different classifiers (LGBM, Naïve Bayes, Random Forrest, Deep learning, etc.). The problem will be solved by using classification algorithms. It is a multi-class classification problem of human written text. The solution should be able to classify future incoming reviews into one of the five ratings with a certain level of accuracy.

Metrics

The solution of this problem can be expressed in terms of accuracy of results predicted by the trained model. Accuracy is defined in as the ratio of correctly classified reviews and the total no. of reviews. Another interesting metric to track will be precision, since we would want to know how many of the reviews binned into one rating belong in that rating. For a two-bin classification problem, accuracy and precision are defined as follows,

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{Total Predictions}}$$

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

II. Analysis

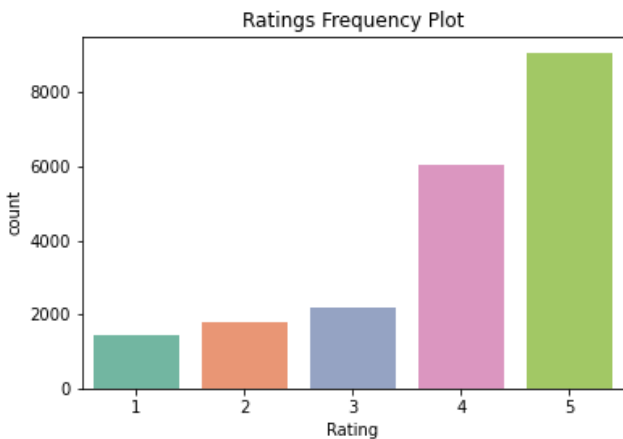
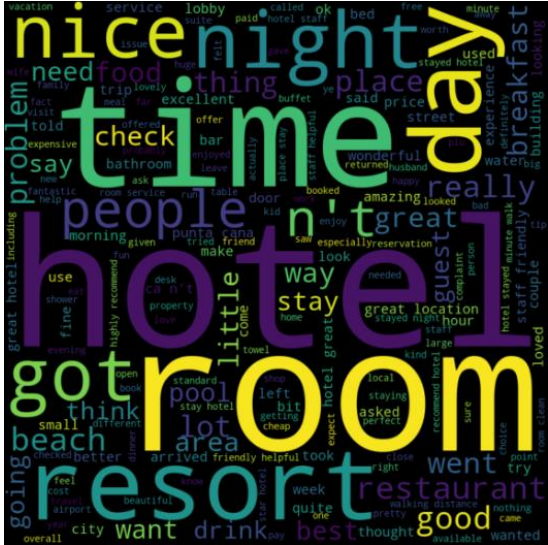
Data Exploration

The first few rows of data in the dataset are provided below as a sample,

Review	Rating
nice hotel expensive parking got good deal stay hotel anniversary, arrived late evening took advice previous reviews did valet parking, check quick easy, little disappointed non-existent view room room clean nice size, bed comfortable woke stiff neck high pillows, not soundproof like heard music room night morning loud bangs doors opening closing hear people talking hallway, maybe just noisy neighbors, aveda bath products nice, did not goldfish stay nice touch taken advantage staying longer, location great walking distance shopping, overall nice experience having pay 40 parking night,	4
ok nothing special charge diamond member hilton decided chain shot 20th anniversary seattle, start booked suite paid extra website description not, suite bedroom bathroom standard hotel room, took printed reservation desk showed said things like tv couch ect desk clerk told oh mixed suites description kimpton website sorry free breakfast, got kidding, embassy suits sitting room bathroom bedroom unlike kimpton calls suite, 5 day stay offer correct false advertising, send kimpton preferred guest website email asking failure provide suite advertised website reservation description furnished hard copy reservation printout website desk manager duty did not reply solution, send email trip guest survey did not follow email mail, guess tell concerned guest.the staff ranged indifferent not helpful, asked desk good breakfast spots neighborhood hood told no hotels, gee best breakfast spots seattle 1/2 block away convenient hotel does not know exist, arrived late night 11 pm inside run bellman busy chatting cell phone help bags.prior arrival emailed hotel inform 20th anniversary half really picky wanted make sure good, got nice email saying like deliver bottle champagne chocolate covered strawberries room arrival celebrate, told needed foam pillows, arrival no champagne strawberries no foam pillows great room view alley high rise building good not better housekeeping staff cleaner room property, impressed left morning shopping room got short trips 2 hours, beds comfortable.not good ac-heat control 4 x 4 inch screen bring green shine directly eyes night, light sensitive tape controls.this not 4 start hotel clean business hotel super high rates, better chain hotels seattle,	2
nice rooms not 4* experience hotel monaco seattle good hotel n't 4* level.positives large bathroom mediterranean suite comfortable bed pillowsattentive housekeeping staffnegatives ac unit malfunctioned stay desk disorganized, missed 3 separate wakeup calls, concierge busy hard touch, did n't provide guidance special requests.tv hard use ipod sound dock suite non functioning. decided book mediterranean suite 3 night weekend stay 1st choice rest party filled, comparison w spent 45 night larger square footage room great soaking tub whirlpool jets nice shower.before stay hotel arrange car service price 53 tip reasonable driver waiting arrival.checkin easy downside room picked 2 person jacuzzi tub no bath accessories salts bubble bath did n't stay, night got 12/1a checked voucher bottle champagne nice gesture fish waiting room, impression room huge open space felt room big, tv far away bed chore change channel, ipod dock broken disappointing.in morning way asked desk check thermostat said 65f 74 2 degrees warm try cover face night bright blue light kept, got room night no, 1st drop desk, called maintenance came look thermostat told play settings happy digital box wo n't work, asked wakeup 10am morning did n't happen, called later 6pm nap wakeup forgot, 10am wakeup morning yep forgotten.the bathroom facilities great room surprised room sold whirlpool bath tub n't bath amenities, great relax water jets going,	3

The dataset has 20491 rows with two columns: 'Review' and 'Rating'. The dataset has no null values. Hence, no additional cleanup or padding has been performed on the dataset. As seen above, the 'Review' column is an English text-based field. The 'Rating' column is a numerical field with values between 1 and 5. The 'Rating' column has an average value of ~3.95 which indicates that the dataset is more skewed towards higher ratings.

Exploratory Visualization

 <table border="1"><caption>Ratings Frequency Plot Data</caption><thead><tr><th>Rating</th><th>Count</th></tr></thead><tbody><tr><td>1</td><td>1500</td></tr><tr><td>2</td><td>1800</td></tr><tr><td>3</td><td>2200</td></tr><tr><td>4</td><td>6000</td></tr><tr><td>5</td><td>8800</td></tr></tbody></table>	Rating	Count	1	1500	2	1800	3	2200	4	6000	5	8800	
Rating	Count												
1	1500												
2	1800												
3	2200												
4	6000												
5	8800												
<p>The 'Rating' column has a higher frequency of 4 and 5 ratings.</p>	<p>Word cloud of most commonly occurring words in the 'Review' column.</p>												

Algorithms and Techniques

For all the algorithms and implementations in this project, the *random_state* parameter has been set 42 wherever applicable as it is the ultimate answer.

The algorithms used in this project are:

- Universal-sentence-encoder:

The universal sentence encoder encodes text into high dimensional vectors that can be used for text classification, semantic similarity, clustering and other natural

language tasks. It is an encoder trained with deep averaging network (DAN) encoder. This encoder is modeled towards inferring the meaning of word sequences instead of individual words. The input is variable length English text and the output is a 512-dimensional vector. It uses semantic similarity which measures the degree to which two pieces of text carrying the same meaning. A huge advantage of using this module is that it is not required to preprocess the data before applying the module.

Further details about this encoder can be found on [TensorFlowHub](#) and the paper [Universal Sentence Encoder](#) on arXiv.

- **LGBMClassifier:**
This classifier has been used as it had the highest accuracy scores in the benchmark. Light GBM (LGBM) is a gradient boosting framework which is based on tree-based learning algorithm. LGBM grows the learning tree vertically (leaf wise) as compared to other tree algorithms which grow horizontally (level wise). It can handle a large size of data, takes lower memory to run, and has a faster runtime.
The default parameters for the classifier are listed [here](#).
- **AdaBoost Classifier:**
This classifier is an iterative ensemble method. It creates a strong classifier by combining multiple poor performing classifiers. It is usually used with decision trees with max depth of 1. It optimizes the results created by the weak learner by adding weak learners sequentially.
The AdaBoost classifier was trained with a decision tree of max_depth=1 and random_state = 42. Rest of the default parameters are listed [here](#).
- **LogisticRegression:**
Logistic regression is one of the most fundamental algorithms used in machine learning. It is named regression, but it is a probabilistic classification model. A threshold is defined between the different categories and then the probability is calculated of whether the item lies in one or the other category. The problem being solved is an ordinal logistic regression problem.
The default parameters for the classifier are listed [here](#).
- **GridSearchCV:**
GridSearchCV is a technique used for optimizing the hyperparameters of any machine learning algorithm using cross validation. The model is trained on the grid of all selected hyperparameters. The performance of the selected

hyperparameters is tested on a dedicated evaluation set that was not used during model training.

The default parameters used are 'accuracy' for grid search scorer and 5 fold cross validation. The rest of the default parameters are listed [here](#).

Benchmark

The benchmark for this project has been taken from [this Kaggle workbook](#). The benchmark is the accuracy scores for classifying the data using the following classifiers:

Classifier	Accuracy Score
LightGBM	59.48%
XGBM	58.06%
GaussianNB	50.74%
RandomForest	55.31%

The LightGBM classifier shows the best accuracy. The objective of this project would be to beat these accuracy scores by either optimizing one of the above-mentioned classifiers or using a different classifier.

III. Methodology

Data Preprocessing

The dataset used for this project was well defined. No additional cleanup or padding was performed on the data.

The dataset was divided into a training and testing set using the 'train_test_split' method. 10% of the dataset was earmarked to be used for testing and was never used for training. The training and testing 'Review' columns were converted into 512 dimension vectors using the universal sentence encoder. The final important input variables, their shape, and description are listed below,

Input Variable	Shape	Description
X_train	(18441, 512)	Training data used for training the classification algorithms.
X_test	(2050, 512)	Testing data used to generate predictions from trained model.
y_train	(18441,)	Training data used for training the classification algorithm.
y_test	(2050,)	Testing data to which predictions from the model are compared.

Implementation

The implementation of this project was done in three basic steps:

- Implement and optimize LGBMClassifier()
- Implement and optimize AdaBoostClassifier()
- Implement and optimize LogisticRegression()

The implementations of all the algorithms are quite straightforward and are available in the attached Notebook.

Refinement

The applied algorithms were optimized using the GridSearchCV method. The details for the individual algorithms are discussed below.

LGBMClassifier:

The parameters and values chosen for refinement are listed below. The decision on these parameters was made after consulting the "[Parameter Tuning](#)" section of the algorithm's documentation.

Parameter	Description	Values
num_leaves	This parameter controls the complexity of the tree.	30, 40, 50, 60, 70
min_child_samples	This parameter defines the min data which should be present in each child and helps avoid overfitting in an LGBM model.	20, 40, 80, 100
max_depth	This parameter denotes the maximum depth of the tree. Setting the value ≤ 0 implies there is no limit.	-1, 3, 5, 7

The initial accuracy with default parameter values was 59.66%.

The final accuracy after hyperparameter tuning was 59.76%. The parameters for the best estimator were : *num_leaves = 50, min_child_sample = 100, max_depth = -1*

AdaBoostClassifier:

The parameters and values chosen for refinement are listed below.

Parameter	Description	Values
n_estimators	The max no of estimators at which boosting is stopped.	20, 50, 100
learning_rate	This decides the contribution of each estimator	0.01, 0.05, 0.1, 0.3, 1
algorithm	Decides which boosting algorithm to used, real vs discrete	SAMME, SAMME.R

The initial accuracy with default parameter values was 53.95%.

The final accuracy after hyperparameter tuning was 55.12%. The parameters for the best estimator were : *n_estimators = 100, learning_rate = 0.3, algorithm = SAMME.R*

LogisticRegression

The parameters and values chosen for refinement are listed below.

Parameter	Description	Values
C	Inverse of regularization strength.	0.01, 0.1, 1, 10, 100
penalty	Norm used during penalization.	l1, l2, elasticnet, none
solver	Algorithm to use in the optimization problem	newton-cg, lbfgs, liblinear, sag, saga

The initial accuracy with default parameter values was 61.41%.

The final accuracy after hyperparameter tuning was 60.91%. The parameters for best estimator were : $C = 10$, $penalty = 'l2'$, $solver = 'lbfgs'$

IV. Results

Model Evaluation and Validation

The final model which was used for this project is the LogisticRegression model.

The model parameters for best accuracy of 61.41% are $C = 1$, $penalty = 'l2'$, $solver = 'lbfgs'$

Since these model parameters were obtained after a 5-fold cross validation, we can assume the model to be fairly consistent with respect to replication and sensitivity.

Justification

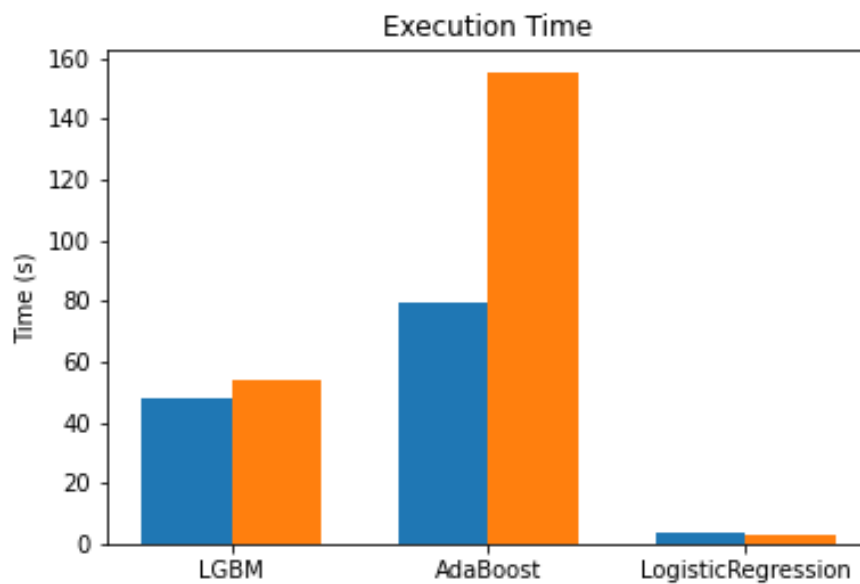
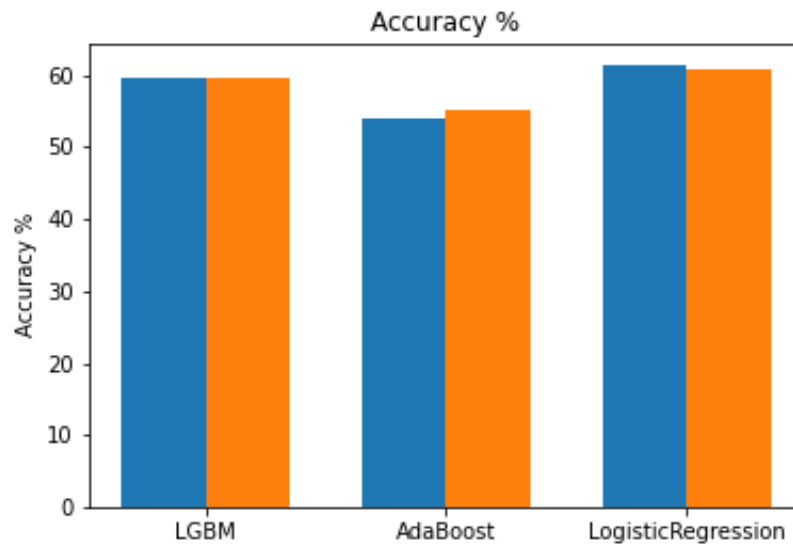
The model developed in this project provides an accuracy of 61.41% on the test dataset.

This is 3.2% better than the best accuracy score in the benchmark. Still, LogisticRegression is a better solution to this problem when the execution times are considered (see Conclusions).

The model in this project is a slight improvement and cannot be the final solution to all future data for this dataset.

V. Conclusion

Free-Form Visualization



As seen above, the accuracy of all the models considered in this project are similar. But, the difference in execution times should make LogisticRegression as the favored algorithm. This will become especially critical during cross validation with large number of parameters or if the base training set grows in size.

Reflection

This project is a classification problem in the domain of NLP. The combination of NLP and classification is a widely explored problem.

The wide variety of classification algorithms available make it difficult as there is no defined best which can directly be used out of the box. With the different available algorithms and possible combinations of hyperparameter tuning, it is near impossible to evaluate all possible combinations.

This project was my first exposure to universal-sentence-encoder and it was interesting to learn about an English to float vector conversion technique which takes semantic similarity into account.

The final solution to the project is according to my expectations but truthfully I had hoped to achieve a higher accuracy score. I believe the approach applied in this project is the correct one as we should focus on the relevant metrics and try to improve them iteratively.

Improvement

The accuracy achieved in this project can certainly be improved upon.

The algorithms implemented in this project can be further improved by increasing the scope of the parameters which were tuned. It will require a higher training time, but it might be worth it if we can increase the accuracy.

I researched a few implementations for NLP centered around CNN and LSTM (Long Short Term Memory) embedding of input data. Such methods might be explored to achieve better accuracy.