Airbnb Crete Reviews Sentiment Analysis

Leonardo Cohen Ioannis Panagiotidis Xaralampos Peteinarelis

***Abstract*—The Airbnb Crete Reviews dataset consists of 264,398 reviews from the official site of Airbnb. The dataset contains reviews in different languages. In this project,we predicted the number of positive and negative reviews based on sentiments using different classification models, including Logistic Regression, Naïve Bayes, Decision Tree (Random Forest, Adaboost), SVM, and Deep Learning using keras library.**

***Index Terms*—Classification, Sentiment Analysis, Machine Learning, Deep Learning**

I. Preprocess

We translated some of the Greek reviews to English using the free version of DeepL Translation, end we dropped every review in different language from the dataframe. After that we used SentimentIntensityAnalyzer from nltk.sentiment.vader python package, to analyze the sentiment of the reviews. Then, we performed downsampling because the dataset was imbalanced so the final dataset has 7741 equally distributed entries. The Airbnb Crete Reviews contains html tags which do not serve any purpose for detecting sentiment, we also decided to remove punctuation whatsoever, even if this means that we get rid of emoticons (there are very few of them anyway), but it makes it easier for us to handle. Finally we lowercase everything. We also apply Porter stemming algorithm, which helps s replace every word with it's root, and so words like cats and cat, or running and run, become the same. This has been shown to improve classification accuracy in sentiment analysis tasks. Another step that helps improve classification performance is what we call negation handling, intuitively words that are preceded by a negation (i.e no, not, hardly, etc.) means the opposite, and thus we replace every patter of the form [negation] [word] by neg [word]. So for example, hardly good would be replaced by neg good.

II. Sentiment

The Reviews Dataset didn't have something to indicate sentiment so we applied nltk.sentiment.vader which is a pretrained model to produce sentiment on text. We wanted to have a binary classification problem, so we set a threshold to 0.5 and we mapped everything above 0.5 to 1(Positive) and everything below to 0(Negative)

III. Vectorization

We used Bag of Words method to vectorize our data specifically we used CountVectorizer from sklearn to perform it. Vectorization is a technique that is used to transform a given text into a vector on the basis of the frequency (count) of each word that occurs in the entire text. The vocabulary we used has length of 10000.

IV. Normalizing

We normalized(unit norm) the data because bag of words vectorize sentences by counting words so there are might be big differences between the dimensions of the vectors and that might affect our models.

V. Non-Linear Machine Learning Models

We used these models to predict Sentiment ,Logistic Regression, Naïve Bayes, Decision Tree (Random Forest, Adaboost), SVM.

A. Logistic Regression

In statistics, the (binary) logistic model (or logit model) is a statistical model that models the probability of one event (out of two alternatives) taking place by having the log-odds (the logarithm of the odds) for the event be a linear combination of one or more independent variables ("predictors") We used LogisticRegression from sklearn: Accuracy: train = 0.7680, test = 0.7624

B. Naive Bayes Classifier

In statistics, naive Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naive) independence assumptions between the features.

- **Gaussian NB** When dealing with continuous data, a typical assumption is that the continuous values associated with each class are distributed according to a normal (or Gaussian) distribution. For example, suppose the training data contains a continuous attribute, x. The data is first segmented by the class, and then the mean and variance of x is computed in each class [GaussianNB] Accuracy: train = 0.74546, test = 0.74090
- **MultinomialNB** With a multinomial event model, samples (feature vectors) represent the frequencies with which certain events have been generated by a p1, ..., pn is the probability that event i occurs (or K such multinomials in the multiclass case). A feature vector x =(x1,...,xn) is then a histogram, with xi counting the number of times event i was observed in a particular instance. [MultinomialNB] Accuracy: train = 0.68839, test = 0.69080
- **BernoulliNB** In the multivariate Bernoulli event model, features are independent Booleans (binary variables) describing inputs. Like the multinomial model, this model is popular for document classification tasks. [BernoulliNB] Accuracy: train = 0.7231, test = 0.7283

- **ComplementNB** Complement Naive Bayes is somewhat an adaptation of the standard Multinomial Naive Bayes algorithm. Multinomial Naive Bayes does not perform very well on imbalanced datasets. Complement Naive Bayes is particularly suited to work with imbalanced datasets. [ComplementNB] Accuracy: train = 0.6878, test = 0.6915

C. Random Forest

Random forests or random decision forests is an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. We used GridSearch for hyperparameter tuning Number of Estimators: [5,50,100] Max Depth of the trees : [2,10,20,None] We checked the train and the test accuracy of these combinations: number estimators = 50, max depth = 20

[RandomForestClassifier]Accuracy: train = 0.9978, test = 0.7894

number estimators = 100, max depth = 20

[RandomForestClassifier]Accuracy: train = 0.9980, test = 0.7890

number estimators = 100, max depth = None

[RandomForestClassifier] Accuracy: train = 0.9984, test = 0.7902

D. AdaBoost with Decision Trees

We used Adaboost to boost the performance of the decision tree classifier with max depth = 5

[AdaBoostClassifier] Accuracy: train = 0.9984, test = 0.7405

E. Support Vector Machine Classifier

We standardized our data and fitted them in a svm model: [Support Vector Machine] Accuracy: train = 0.8771, test = 0.8027

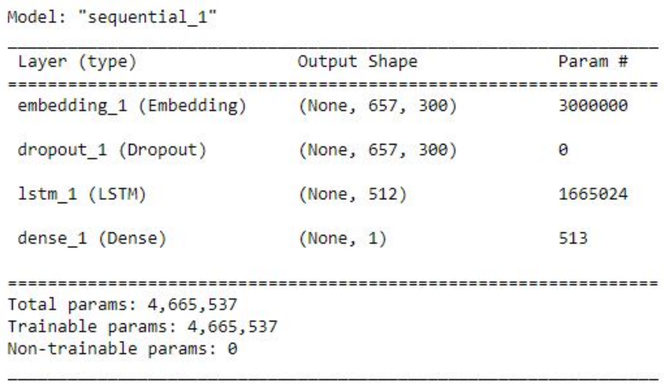
VI. Neural Networks

A. Preprocess Data

We followed a different method to preprocess our data in order to feed them in the Neural Network model we created. We used the tokenization method from keras with vocabulary updated with the reviews of the dataset. Then we padded everything with zeros and every input vector has length 657.

B. Model

Model with keras Tokenization indexes:



```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, 657, 300)          3000000

 dropout_1 (Dropout)         (None, 657, 300)          0

 lstm_1 (LSTM)               (None, 512)               1665024

 dense_1 (Dense)             (None, 1)                 513

=================================================================
Total params: 4,665,537
Trainable params: 4,665,537
Non-trainable params: 0
_________________________________________________________________
```

The model we used is the following: An embedding layer with vocabulary size = 10000 dimension of embeddings = 300, Dropout mechanism with probability of dropping out 0.2, a LSTM Layer with 512 units and 0.2 dropout and a dense layer with binary output, using sigmoid as an activation function.

C. Compilation and Evaluation

We compiled the model using as loss function binary crossentropy and optimizer adam v2.Adam with learning rate 0.001

We fitted our data using 32 as batch size. The training of the Bag Of Words model stopped at 5 epochs with

Train accuracy: 0.9720 val accuracy: 0.8902

Test accuracy: 0.89706

References

[1] Bishop, C.M. (2006). *PATTERN RECOGNITION AND MACHINE LEARNING*. M.Jordan(Ed.) & J.Kleinberg(Ed.) & B.Scholkop(Ed.) Singapore: Springer Science+Business Media, LLC

[2] Brownlee, J. (2019, August 7). *A gentle introduction to the bag-of-words model*. Machine Learning Mastery. Retrieved May 30, 2022, from https://machinelearningmastery.com/gentle-introduction-bag-words-model/

[3] Sawhney, P. (2021, September 20). *Introduction to stemming and lemmatization (NLP)*. Medium. Retrieved May 30, 2022, from https://medium.com/geekculture/introduction-to-stemming-and-lemmatization-nlp-3b7617d84e65

[4] Sklearn.feature extraction.text.CountVectorizer. scikit. (n.d.). Retrieved May 30, 2022, from https://scikit-learn.org/stable/modules/generated/sklearn.feature extraction.text.CountVectorizer.html

[5] Wikimedia Foundation. (2022, April 25). *Zipf's Law*. Wikipedia. Retrieved May 30, 2022, from https://en.wikipedia.org/wiki/Zipf's_law

[6] Rai, S. (2021, March 10). *Word cloud: A text visualization tool*. Medium. Retrieved May 30, 2022, from https://medium.com/analytics-vidhya/word-cloud-a-text-visualization-tool-fb7348fbf502

[7] Knight, A. (2021, April 26). *LSTM neural network: The basic concept*. Medium. Retrieved May 30, 2022, from https://towardsdatascience.com/lstm-neural-network-the-basic-concept-a9ba225616f7

[8] Tensorflow. TensorFlow. (n.d.). Retrieved May 30, 2022, from https://www.tensorflow.org/

[9] Scikit-learn. Scikit-learn. (n.d.). Retrieved May 30, 2022, from https://scikit-learn.org/stable/index.html

[10] "Get the Data." Inside Airbnb, http://insideairbnb.com/get-the-data/.

[11] NLTK, https://www.nltk.org/api/nltk.sentiment.vader.html.

[12] Deepl Translate API: Machine Translation Technology. DeepL Translate API — Machine Translation Technology. (n.d.). Retrieved June 7, 2022, from https://www.deepl.com/pro-api?cta=header-pro-api