**PROJECT: Predicting if a customer's review is good or bad (Polarity)**

**APPROACH OF THE PROJECT**

1. Data importing, Text cleaning, Text analysis

* The program has been written in Python language and is written Jupyter Notebook.
* At first the necessary packages required for the program is imported:

NumPy, Pandas, matplotlib, scikit-learn, nltk, text blob, regular expression, seaborn

* The train and test dataset are loaded
* Basic checks for the dataset are done like checking shape, null values, target unique values, value counts by visualization
* Text cleaning: the review text in the datasets needs various types text cleaning processes. The text cleaning functions used here are
* ReviewText.lower() - make text lowercase
* re.sub('\[.\*?\] - remove text in square brackets
* re.sub('https?://\S+|www\.\S+ -remove links
* re.sub('<.\*?>+ )
* re.sub('[%s]' % re.escape(string.punctuation) -remove punctuations
* re.sub('\n')
* re.sub('\w\*\d\w\*) - remove words containing numbers
* These text cleaning regular expression functions are kept inside a defined function def clean\_text(ReviewText) (clean\_text() function applies a first round of text cleaning techniques)
* After cleaning of texts, a function text\_preprocessing is used. (-the function text\_preprocessing() then takes in the processed text from the clean\_text() function and applies techniques like tokenization and stop word removal.)
* Analyzing text statistics: data insights
* Text length analysis: created a new feature text\_len
* Word frequency analysis: created a new feature text\_word\_count
* By bar graphs and a scatter graph visualized the frequency and distribution of text length and text word count in bad and good polarity.
* Use of TextBlob to get the Sentiment Polarity
* To check text whether the correctly classified in the polarity
* To check whether it gives the accurate result for most positive and most negative review from the text
* To check text cleaning is accurate or further cleaning is required.
* checking of the most negative review with a sentiment polarity of –1
* checking of the most positive review with a sentiment polarity of 1
* The sentiment polarity shows the data cleaning and target variable has been correctly distributed
* As we can see the validity through the most negative and most positive review with a sentiment polarity of 1 and -1 has been achieved
* Thus, after text cleaning and checks a final dataframe has been formed for the model building: train\_final.csv

2. Model Building

* Load the train cleaned dataframe
* Use of lemmatizing – WordNetLemmatizer : to analyse the words as a single item. A function has been defined as own\_analyser .
* Declaring X and y variables
* Using train test split.
* CountVectorizer : to build a bag of words ‘BOW’ of known words with the own\_analyser function.
* TfidfVectorizer : Transforms text to feature i.e ‘BOW’ to vectors that can be used as input to estimator for the algorithm.
* A pipeline feature from sklearn library is used to put the CountVectorizer and TfidfVectorizer and different classifier algorithms together. Pipeline feature is good to use for feature selection
* Classifier Algorithms. used:
* **Naïve Bayes – multinomialNB**
* **Naïve Bayes- bernoulliNB**
* **Linear SVC**
* **Logistic Regression**
* **Random Forest Classifier**
* A classification report has been shown so as to see the various model accuracies and precision and recall scores.
* FROM THE ABOVE MODELS WE CAN GET THE HIGHEST ACCURACY FROM THE LINEAR SVC ALGORITHM
* Applied the Linear SVC model for the test dataset to get the predicted response of reviews.
* Finally created csv file containing UserID and predicted Response in a csv file for test data.

Author

Karunya Haloi