Dan Burke

Dburke01@syr.edu

Homework 6

MB Naïve Bayes

05/27/2023

|  |  |
| --- | --- |
| Homework 6 – MB Naïve Bayes & Bernoulli Naïve Bayes | |
| **Introduction** | Sentiment analysis has a multitude of economic applications with text data both on the side of the consumer and that of the producer. This report attempts to address one application of sentiment analysis and lie detection within the scope of restaurant reviews.  This report intends to apply the Naïve Bayes (Multinomial and Bernoulli) approach to restaurant review text data sourced from course materials. This data possesses lie, sentiment, review and features.  Naïve Bayes is a probabilistic machine learning process utilizing the Bayes’ theorem. This process assumes that all features (columns) of the data set are conditionally independent. Within the scope of this report the features refer to induvial words; all individual words contained within the dataset. |
| **Analysis** | data preparation The restaurant review data has been provided by Professor Gates. This data’s initial form was within a CSV (“Comma Separated File”) file. This format presented a single complex issue preventing reading into a data frame via the pandas python library; multiple (often unknown amounts) of commas contained within the review column.    Figure Data Importation and Cleaning  This issue was solved by:   * Reading the CSV file as a text file from its local directory * Iterating line by line   + Splitting the string by comma values   + Iterating in a nested manner replacing all comma characters which appear after the first two comma characters: starting at the index of 2 and continuing through to the end of all comma values. * Concatenating the new review string with the previous segments split by the spit method call (index 0 and 1) * Writing this new string to a clean CSV file * Downloading (the effort utilized a virtualized environment) this new cleaned CSV file * Reading the cleaned data CSV to a new data frame   **Data Types**  The Multinominal (MB) Naïve Bayes and Bernoulli Naïve Bayes approach requires all data to be numerical, this requires two steps. First to check the cleaned data frames column types as well as count vectorizing (establishing word occurrence counts).  Two columns required a change to a categorical data type; “lie” and “sentiment”. Once these data types have been properly configured and verified the CountVectorizer method of the SKLearn Python library was utilized.    Figure Mutation to Categorical Data Type  **Lie and Sentiment Data Frame Creation & Label Extraction**  MB Naïve Bayes and Bernoulli Naïve Bayes models only allows for one “label” or “classifier”. This required the effort to create two data frame (though not necessary labels could be extracted and applied to models separately while using the same text data). Both “lie” and “sentiment” were stored within a “labels” variable.  With these separate data frames (lie and sentiment) it was now necessary to count vectorize the text data, in order to format it properly for generation of a MB Naïve Bayes and a Bernoulli Naïve Bayes models.    Figure Lie and Sentiment Data Frame Creation  **Additional Data Cleaning – Feature Requirements**  This effort though like Assignment 4, differs in that there are enhanced requirements in the form of feature (word) constraints; as well as the use of “TfidfVectorizer”. The enhanced feature requirements are as follows:   * Remove all words/features with a character count of two (2) or less. * Remove all words/features that are or contain numerical characters. * Remove all words/features that are larger that thirteen (13) characters.   To fulfill these enhanced feature constraints, a custom method was created to parse a count vectorized (either CountVectorizer() or TfidfVectorizer() output) and return a list of all features to remove. The decision was made to store these removed values so that further inspection may be applied to the removed values.    Figure Feature Enhanced Cleaning Method  This custom method is intended to be used after a corpus or textual dataset has been count vectorized and transformed into a data frame. This approach it intended so that the user may remove these features yet retain the counts of each feature should they wish to independently evaluate removed word occurrences.    Figure Custom Method Usage  **Fitting the Model - MB Naïve Bayes**  Once the text data intended for use with a model has been count vectorized via “TfidfVectorizer()” it was then necessary to create the “test, train, split” of the data. This is separating the data into a training (model generation) and test (prediction) segments. A 80% training and 20% testing split was utilized. The same split data was applied to MB Naïve Bayes Models as well as the Bernoulli Naïve Bayes Models.    Figure Sentiment Vectorization via TfidfVectorizer    Figure Lie Detection Vectorization via TfidfVectorizer    Figure Test Train Split MB Naïve Bayes & Bernoulli Naïve Bayes - Lie and Sentiment  Once the test, train, split has been created it was then required to fit the data to the model. X referring to the textual data, y referring to the label (classifier).  **Sentiment – MB Naïve Bayes**    **Lie Detection – MB Naïve Bayes**    **Sentiment– Bernoulli Naïve Bayes**    **Lie Detection – Bernoulli Naïve Bayes** |
| **results** | technical results **Sentiment Analysis – Naïve Bayes**      **Lie Detection – Naïve Bayes**      **Sentiment – Bernoulli Naïve Bayes**      **Lie Detection – Bernoulli Naïve Bayes** |
| **conclusions** | Initially, within assignment 4, the Multinomial Naïve Bayes approached produced results of acceptable accuracy ~78.9% within sentiment analysis and undesirable accuracy ~57.8% for lie detection. However, with the enhanced feature requirements that called for additional data cleaning, this effort has been able to increase sentiment analysis accuracy to an impressive ~94%, while lie detection degraded to ~47%. This brings to light how different levels of “cleanliness” of the data must be used depending on the analysis goal. With these new results, this effort theorizes that lie detection benefits from the data being in it’s most raw form, while sentiment analysis does not.  This new MB Naïve Bayes Sentiment model possesses an accuracy which may be considered for use within a production environment (pending additional testing and validation) while lie detection model does not.  The Bernoulli implementation with respect to lie detection model generation outperformed MB Naïve Bayes with an accuracy of ~73%. The Bernoulli approach also performed remarkably well with sentiment analysis producing an accuracy of ~84%. This is a are increase in accuracy which with additional cleaning of the data (expansion of stop words) may produce even more impressive results.  . |