**IDS 566 Advanced Text Analytics Business -**

**Final Project Report**

**Group – 7**

**Team Members:**

**Palak Gupta 678860489**

**Meenakshi Janardanan 671917004**

**Ashwyn Muthukumaraswami 675870750**

**Gughanraj Selvaraju 662075168**

**Aruna Venkatasubramaniam 678860489**

**Introduction:**

The 20 Newsgroup dataset is a collection of 20,000 Newsgroup documents which are partitioned evenly across 20 different groups which corresponds to different topic. Some topics are very closely related and other topics are not.

The goal of the project is to train different classifier and test the outcomes trying to achieve as high accuracy. We used Naïve Bayes, Logistic Regression, and Support Vector Machines. We are trying to achieve high accuracy by using these methods.

**Business Relevance:**

These 20,00 messages that are categorized based on topic are automatically classified into text using natural language processing unlike manual classification where human categorizes text and classifies it accordingly.

**Data Description:**

The data is collection of newgroup documents. The data has split into train and test with 11314 and 7532 documents respectively.

The list of the 20 newsgroups and distribution in the train data:

We can see that all the classes are distributed almost evenly. There is no need to over sample or under sample in this case.

**Data Cleaning:**

**Words with Numbers and Numbers alone:**

**Challenge**: Examining the words revealed the presence of words with numbers or just numbers which we felt would be sub-optimal to use in our Bag of Words approach

**Solution**: Removed the numbers from the words using regular Expression

**Data Transformation:**

**Stemming and Lemmatization**:

Stemming and lemmatization is one of the pre-dominant pre-processing techniques in text analysis. We performed stemming with snowball stemmer and Lemmatization using WordNet Lemmatizer

**Challenge:** Adding the stemming and lemmatization as the part of the model building pipeline increased the runtime of models exponentially

**Solution:** Performed the stemming and lemma once as a preprocessing step and added the transformed text as the part of the original training data

**Vectorization:**

Vectorization enables us to convert data in a text format into Bag of Words. It performs tokenization, stop words removal, lower case conversion and de-duplication of words. We have used two types of vectorizer in this exercise:

1. **Count Vectorizer**: Count Vectorization involves counting the number of occurrences each word appears in a document
2. **TF-IDF Vectorizer**: Convert a collection of raw documents to a matrix of TF-IDF features. TF-IDF is an abbreviation for Term Frequency-Inverse Document Frequency and is a very common algorithm to transform text into a meaningful representation of numbers.
   1. Term Frequency = Total number of documents words is present /total words in doc
   2. Inverse Document Frequency = log (total number of documents/number of documents containing word w)

**Challenge:** Exclusion of most common words that occur in significant percent of documents

**Solution**: Ran a cross validation approach to fix on excluding the words that appear in less than 1% of the documents

**Models:**

Naïve Bayes, Logistic Regression with Lasso & Ridge Regularization, and SVM are the chosen models for this exercise. We built a pipeline using sci-kit learn library to efficiently run the different variations of the model

Cleaning +Stemming + lemma

Training the model

Reading the data

Evaluation of the model

**Challenge**: Finding the optimal hyperparameters for the each of the models

**Solution:** Grid Search based Cross validation was employed to find the best parameters for each of the models

**Results:**

For this exercise accuracy alone is used as the metric to find the best model. Precision and recall are typically used to find the performance of the class of interest over the other classes. Since we have 20 classes which all are assumed to have equal important using precision and recall will not result in any valuable insights.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Preprocess | Parameters | Training Accuracy | Testing Accuracy |
| Naïve Bayes | With Stemming | No Tuning | 95% | 81.90% |
| Without Stemming | clf\_\_alpha': 0.001, 'tfidf\_\_use\_idf': True, 'vect\_\_ngram\_range': (1, 2) | 99.94% | 85.09% |
| With Stemming | clf\_\_alpha': 0.001, 'tfidf\_\_use\_idf': True, 'vect\_\_ngram\_range': (1, 2) | 100% | 83% |
| **Logistic Regression with regularization** | **Without Stemming** | **CVectorizer\_\_max\_df': 0.1, 'CVectorizer\_\_ngram\_range': (1, 2), 'SGDC\_\_alpha': 1e-05, 'SGDC\_\_penalty': 'l2'** | **99.95%** | **85.66%** |
|
|
|
| With Stemming | CVectorizer\_\_max\_df': 0.1,'CVectorizer\_\_ngram\_range': (1, 2),'SGDC\_\_alpha': 1e-05,'SGDC\_\_penalty': 'l2' | 99.94% | 85.17% |
|
|
|
| SVM | With Stemming | No Tuning | 96.14% | 81.59% |
| Without Stemming | clf-svm\_\_alpha': 0.0001, 'tfidf\_\_use\_idf': True, 'vect\_\_ngram\_range': (1, 2) | 99.73% | 85.55% |
| With Stemming | clf-svm\_\_alpha': 0.0001, 'tfidf\_\_use\_idf': True, 'vect\_\_ngram\_range': (1, 2) | 99.55% | 82.83% |

**Findings & Insights:**

**Logistic Regression** **model without stemming and L2 Regularization** is the best model with test data accuracy of 85.66% barely edging the model with stemming.

Logistic Regression marginally outperforms SVM and Naïve Bayes because it doesn’t have any underlying assumptions. SVM assumes the data to be linearly separated and we can project the data into higher dimension using kernels to validate whether the transformation outperforms the current best model. Here however we did not transform data into a higher vector space since that would reduce the interpretability of model.