* *Methods*: What did you do with the data, precisely?

1. Tokenize the data to form a feature vector.
2. Build a text classifier by using Logistic Regression and Naïve Bayes methods.

Logistic Regression Method:

1). In our project we are using a Logistic Regression model of the form f:X->Y where Y is boolean valued variable and X is a feature vector containing continuous variables.

2) Logistic Regression assumes a parametric form for the distribution P(Y|X), then directly estimates its parameters from the training data. The parametric model assumed by Logistic Regression in our case is shown below, as Y is boolean :

p(yi|xi)=1÷1+e−yixi⋅θ

Naïve Bayes Method:

* In our project we are using a Naïve Bayes rule to approximate a target function of the form f:X->Y where Y is boolean valued variable and X is a feature vector containing continuous variables
* The rule is stated below:
  1. P(y/x) = p(x/y)p(y)/ p(x)

As we have a binary classification label vector Y we can use above rule for the classification for an instance X.

1. In Logistic Regression based classification method assess classifier’s accuracy by performing cross validation technique on the training data set.
2. Perform experiments to analyze if the classifiers accuracy can be improved by using different tokenization techniques and parameters (min\_df, max\_df, ngrams\_range, binary..etc) to form efficient feature vectors .
3. Read testing data and predict the labels of all the documents using classifier.
4. Compute the accuracy of the classifier on the testing data and analyze top errors that the classifier has made while predicting labels of the documents in training set.

Experiments:

We are performing two experiments having different sizes of test documents. Few common functions used in Logistic Regression for both the data sets are listed below:

**Function name**:  **compare\_n\_folds ()**

On varying the setting of n\_folds parameter in the do\_expt function to be in [2,5,10,20], we are getting below graph with accuracy on y-axis and n\_folds on x-axis:

Graph: Refer figure xxx in folder yyy

Analysis: As the number of folds increases the classifier’s accuracy increases because for higher number of folds we will get a bigger training set.

**Function name**: compare\_binary **()**

By calling the do\_expt twice, once with binary=True, and once with binary=False we are getting below accuracies:

Accuracy: [0.85414556962025312, 0.82892405063291152]

Analysis: There is an update rule in the feature vector representation. For term frequency the variance in the update rule will be higher as compared to binary. Therefore we are observe a better accuracy while using a binary feature vector as compared to term frequency.

**Function name**: tokenizer\_expt **()**

An experiment to see how does the tokenizer affect results for four different tokenizers as below:

1- tokenize

2- tokenize\_with\_punct

3- tokenize\_with\_appostrophe

4- tokenize\_with\_negative\_words

Accuracy: [0.854145569, 0.8390189873, 0.83889240506, 0.8565189873]

Analysis: We can see that “tokenize\_with\_negative\_words “results in higher accuracy because negation plays a special role in sentiment expression.

**Function name**: min\_df\_expt **()**

This function varies the setting of min\_df parameter in the do\_expt function to be ints in the range (1,10) (inclusive), we are getting below graph with accuracy on y-axis and min\_df on x-axis:

Graph: Refer figure xxx in folder yyy

Analysis: We observe accuracy better at min\_df = 2 than min\_df = 1 because very rare terms are removed and the classifier doesn’t get trained on these terms. Hence it values only relevant terms.

**Function name**: max\_df\_expt **()**

This function varies the setting of max\_df parameter in the do\_expt function to be one of [.1, .2, .3, .4, .5, .6, .7, .8, .9, 1.] , we are getting below graph with accuracy on y-axis and max\_df on x-axis:

Graph: Refer figure xxx in folder yyy

Analysis: We observe best accuracy when max\_df = .2 because it helps in limiting the size of the vocabulary to sentiment expressing words like: love, miss instead of stop words (the, of, a, an.. etc)

**Function name**: n\_grams\_expt **()**

This function varies the setting of n\_grams parameter in the do\_expt function to be one of [(1,1), to (1,6 )] we are getting below graph with accuracy on y-axis and n\_grams\_range on x-axis:

Graph: Refer figure xxx in folder yyy

Analysis: We have observed that unigrams outperforms bigrams , trigrams because unigrams provides

a good coverage of the data.

**Experiment 1 :**

**Data Set** :

|  |  |  |
| --- | --- | --- |
| Labels/Category | Train document counts | Test document counts |
| pos/positive | 200 | 102 |
| neg/negative | 197 | 99 |

**Logistic Regression:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Tokenizer 1 | Tookenizer 2 | Tokenizer 3 | Tokenizer 4 |
| **Test data accuracy** | 0.8905 | 0.8706 | 0.8806 | 0.8856 |
| False Positive documents | 14 | 16 | 13 | 14 |
| False Negative documents | 8 | 10 | 11 | 9 |

**Naïve Bayes:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Tokenizer 1 | Tookenizer 2 | Tokenizer 3 | Tokenizer 4 |
| **Test data accuracy** | 0.88059 | 0.89054 | 0.86567 | 0.87064 |
| False Positive documents | 14 | 13 | 15 | 15 |
| False Negative documents | 10 | 9 | 12 | 11 |

**Experiment 2:**

**Data Set**:

|  |  |  |
| --- | --- | --- |
| Labels/Category | Train document counts | Test document counts |
| pos/positive | 200 | 193 |
| neg/negative | 197 | 199 |

**Logistic Regression :**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Tokenizer 1 | Tookenizer 2 | Tokenizer 3 | Tokenizer 4 |
| **Test data accuracy** | 0.8163 | 0.7934 | 0.8112 | 0.8189 |
| False Positive documents | 37 | 40 | 40 | 40 |
| False Negative documents | 35 | 41 | 34 | 31 |

**Naïve Bayes:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Tokenizer 1 | Tookenizer 2 | Tokenizer 3 | Tokenizer 4 |
| **Test data accuracy** | 0.82142 | 0.81887 | 0.80102 | 0.79846 |
| False Positive documents | 48 | 46 | 48 | 48 |
| False Negative documents | 22 | 25 | 30 | 31 |

**Analysis:**

* + 1. From the above experiments we observe that accuracies obtained by both the classification methods i.e. Logistic Regression and Naïve Bayes are nearly equal.
    2. Both methods failed to identify true labels of few documents for the test data:
    3. In both the cases classifier’s accuracy reduces as the size of testing data increases while keeping the same size of training data