data-X | Homework 9: NLP

Louis Tilloy

1 Unigram setting

Original reviews
Results obtained with these two lines of code:

```
original_clean_reviews=review_cleaner(train['review'],lemmatize=False,stem=False)
train_predict_sentiment(cleaned_reviews=original_clean_reviews, y=train["sentiment"],ngram=1,
                        max_features=1000)
```

```
 The training accuracy is:  0.9999
 The validation accuracy is:  0.82

CONFUSION MATRIX:
         Predicted
          neg pos
 Actual
     neg   [2101  447]
     pos   [ 453 1999]

TOP TEN IMPORTANT FEATURES:
['worst', 'bad', 'great', 'waste', 'awful', 'terrible', 'excellent', 'worse', 'best', 'boring']
```

A. lemmatized reviews

```
lemmatize_clean_reviews=review_cleaner(train['review'],lemmatize=True,stem=False)
train_predict_sentiment(cleaned_reviews=lemmatize_clean_reviews, y=train["sentiment"],ngram=1,
                        max_features=1000)
```

```
 The training accuracy is:  0.99995
 The validation accuracy is:  0.8314

CONFUSION MATRIX:
         Predicted
          neg pos
 Actual
     neg   [2121  427]
     pos   [ 416 2036]

TOP TEN IMPORTANT FEATURES:
['bad', 'worst', 'great', 'waste', 'awful', 'excellent', 'best', 'terrible', 'boring', 'nothing']
```

B. stemmed reviews

```
stem_clean_reviews=review_cleaner(train['review'],lemmatize=False,stem=True)
train_predict_sentiment(cleaned_reviews=stem_clean_reviews, y=train["sentiment"],ngram=1,
                        max_features=1000)
```

```
 The training accuracy is:  1.0
 The validation accuracy is:  0.819

CONFUSION MATRIX:
         Predicted
          neg pos
```

```
 Actual
     neg   [2100  448]
     pos   [ 457 1995]

TOP TEN IMPORTANT FEATURES:
['bad', 'worst', 'wast', 'great', 'aw', 'love', 'excel', 'bore', 'terribl', 'best']
```

Analysis

We can see that there is no significant difference between the training on the original data, the lemmatized one and the stemmed one. The accuracy is 1% better for the lemmatized version but it might as well just be the randomness from the training of the random forest. The three options also have the same trade-off precision/recall.
Whatever the method used, the network manages to completely over-fit the training data. Therefore in this case, the lemmatization or the stemmization do not bring enough generality to the words.

2 Bigram setting

original reviews

```
original_clean_reviews=review_cleaner(train['review'],lemmatize=False,stem=False)
train_predict_sentiment(cleaned_reviews=original_clean_reviews, y=train["sentiment"],ngram=2,
                        max_features=1000)
```

```
 The training accuracy is:  0.99995
 The validation accuracy is:  0.821

CONFUSION MATRIX:
         Predicted
          neg pos
 Actual
     neg   [2113  435]
     pos   [ 460 1992]

TOP TEN IMPORTANT FEATURES:
['bad', 'worst', 'great', 'awful', 'waste', 'excellent', 'terrible', 'worse', 'boring', 'best']
```

A. lemmatized reviews

```
original_clean_reviews=review_cleaner(train['review'],lemmatize=True,stem=False)
train_predict_sentiment(cleaned_reviews=original_clean_reviews, y=train["sentiment"],ngram=2,
                        max_features=1000)
```

```
 The training accuracy is:  1.0
 The validation accuracy is:  0.8174

CONFUSION MATRIX:
         Predicted
          neg pos
 Actual
     neg   [2085  463]
     pos   [ 450 2002]

TOP TEN IMPORTANT FEATURES:
['bad', 'worst', 'great', 'waste', 'awful', 'excellent', 'terrible', 'worse', 'nothing', 'wonderful']
```

B. stemmed reviews

```
original_clean_reviews=review_cleaner(train['review'],lemmatize=False,stem=True)
train_predict_sentiment(cleaned_reviews=original_clean_reviews, y=train["sentiment"],ngram=2,
                        max_features=1000)
```

```
 The training accuracy is:  1.0
 The validation accuracy is:  0.817

CONFUSION MATRIX:
         Predicted
          neg pos
 Actual
     neg   [2091  457]
     pos   [ 458 1994]

TOP TEN IMPORTANT FEATURES:
['worst', 'bad', 'great', 'wast', 'aw', 'excel', 'bore', 'stupid', 'wast time', 'love']
```

Analysis

We can see that there is no significant difference between the training on the original data, the lemmatized one and the stemmed one. The three options also have the same trade-off precision/recall. As for the Unigram setting, the network manages to completely over-fit the training data whatever the method used. Therefore in this case, the lemmatization or the stemmization do not bring enough generality to the words.

3 Unigram and lemmatization setting

With 10 features

```
original_clean_reviews=review_cleaner(train['review'],lemmatize=True,stem=False)
train_predict_sentiment(cleaned_reviews=original_clean_reviews, y=train["sentiment"],ngram=1,
                        max_features=10)
```

```
 The training accuracy is:  0.87145
 The validation accuracy is:  0.559

CONFUSION MATRIX:
         Predicted
          neg pos
 Actual
     neg   [1404 1144]
     pos   [1061 1391]

TOP TEN IMPORTANT FEATURES:
['film', 'movie', 'one', 'good', 'character', 'time', 'like', 'get', 'story', 'even']
```

With 100 features

```
original_clean_reviews=review_cleaner(train['review'],lemmatize=True,stem=False)
train_predict_sentiment(cleaned_reviews=original_clean_reviews, y=train["sentiment"],ngram=1,
                        max_features=100)
```

```
 The training accuracy is:  0.9999
 The validation accuracy is:  0.7212

CONFUSION MATRIX:
         Predicted
          neg pos
 Actual
     neg   [1864  684]
     pos   [ 710 1742]

TOP TEN IMPORTANT FEATURES:
['bad', 'great', 'movie', 'film', 'one', 'best', 'even', 'like', 'love', 'nothing']
```

With 1000 features

```
original_clean_reviews=review_cleaner(train['review'],lemmatize=True,stem=False)
train_predict_sentiment(cleaned_reviews=original_clean_reviews, y=train["sentiment"],ngram=1,
                        max_features=1000)
```

```
 The training accuracy is:  0.99995
 The validation accuracy is:  0.8192

CONFUSION MATRIX:
         Predicted
          neg pos
 Actual
     neg   [2104  444]
     pos   [ 460 1992]

TOP TEN IMPORTANT FEATURES:
['bad', 'worst', 'great', 'waste', 'awful', 'excellent', 'worse', 'boring', 'best', 'nothing']
```

With 5000 features

```
original_clean_reviews=review_cleaner(train['review'],lemmatize=True,stem=False)
train_predict_sentiment(cleaned_reviews=original_clean_reviews, y=train["sentiment"],ngram=1,
                        max_features=5000)
```

```
 The training accuracy is:  1.0
 The validation accuracy is:  0.839

CONFUSION MATRIX:
         Predicted
          neg pos
 Actual
     neg   [2144  404]
     pos   [ 401 2051]

TOP TEN IMPORTANT FEATURES:
['bad', 'worst', 'great', 'waste', 'awful', 'excellent', 'best', 'nothing', 'worse', 'wonderful']
```

Analysis

We can see that increasing the number of features increases the accuracy. At some point (above 10 features, below 100) it makes the model over-fit because the accuracy on the training set becomes 1. However it manages to still increase the accuracy on the testing set which is pretty much luck since we are still minimizing the same loss with more features and that usually reduces the accuracy of the testing set because the model has more opportunity to exploit a set of features well-fitted for the training data.