Homework 4: Analyzing Restaurant Reviews using Machine Learning and Naive Bayes Methods

Student: Patrick Walsh
Professor: Dr. Ami Gates
School: Syracuse University
Date: 2/10/2024

INTRODUCTION

Product reviews are valuable to businesses for several crucial reasons. For starters, they provide insight into consumer purchasing decisions and rationale, and shed light on consumer spending habits. Reviews not only provide insights into consumer choices but actively help to influence consumer choices. For example, positive reviews may help encourage consumers to make certain purchases while negative reviews may deter certain purchases. Product reviews also help to build trust and credibility for products and businesses by building consensus among customers as to their quality and reputation. Furthermore, product reviews provide a mechanism for consumer feedback and improvement where businesses can make changes to products and services based on the needs and wants of their customers. Additionally, product reviews serve as a form of consumer-to-consumer marketing, building awareness and brand loyalty based on direct consumer feedback.

With these benefits in mind, we turn to a dataset of restaurant reviews that can provide insights and benefits to businesses by using machine learning to evaluate two aspects of these reviews:

1. Determining whether reviews are real or fake, and
2. Categorizing reviews as positive or negative.

Each aspect provides different benefits to businesses. For example, determining whether reviews are real or fake allows businesses to focus on reviews that are legitimate and which reviews to ignore since they are not from real customers. Another advantage of separating real from fake reviews is that it allows businesses to filter out fake reviews so that they are not seen by customers, since an abundance of fake reviews on a product will erode consumer trust not only in the review process but potentially in the product itself.

In categorizing reviews as either positive or negative (known as sentiment analysis), businesses can gauge whether a product is likely to be purchased based on the influence of these reviews. A consensus of negative reviews may hurt brand reputation and loyalty and deter customers from shopping at that business altogether, while a consensus of positive reviews may have the opposite effect.

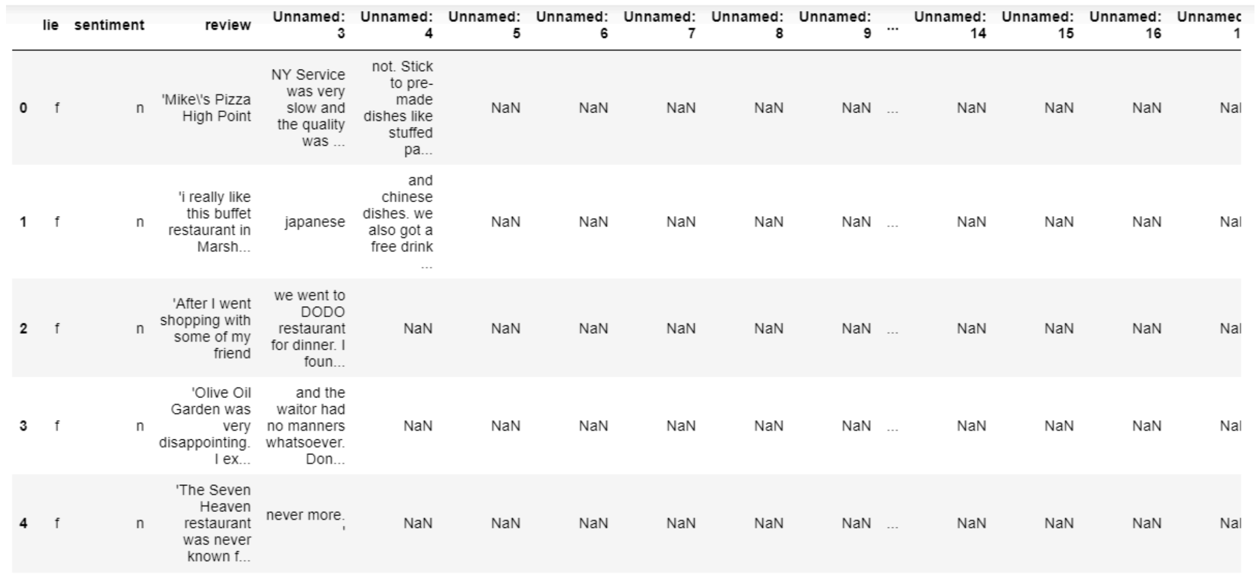
The purpose of this paper will be to use machine learning methods to analyze both of these aspects of product reviews for restaurants and analyze the business value that this analysis brings.

ANALYSIS

Data Preprocessing

The dataset consists of labeled, semi-structured restaurant reviews located in a single CSV file. The first column contains labels of true or false for whether they are real reviews, and the second column contains labels of negative or positive for the review sentiment classification. The remaining columns contain the restaurant reviews text broken up into multiple columns across the CSV. A number of data preprocessing steps are necessary to clean up the CSV and also to create two separate dataframes from this initial CSV file. The first dataframe will be used to create a Lie Detection dataframe for the restaurant reviews, and the second dataframe will be used to create a sentiment classification dataframe.

Below is a snapshot of the CSV file before any preprocessing has been applied. As can be seen in the snapshot, the data is very messy and disorganized, containing many unnamed columns and null rows along with miscellaneous characters. The data preprocessing will clean this up.

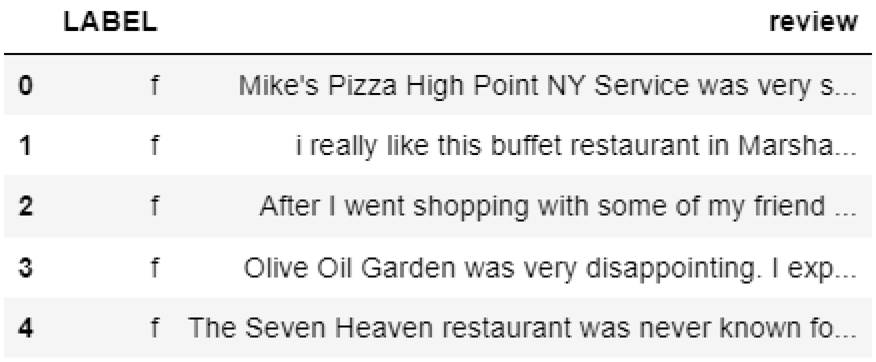


	lie	sentiment	review	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unnamed: 9	...	Unnamed: 14	Unnamed: 15	Unnamed: 16	Unnamec 1
0	f	n	'Mike\'s Pizza High Point	NY Service was very slow and the quality was ...	not. Stick to pre-made dishes like stuffed pa...	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	Nal
1	f	n	'i really like this buffet restaurant in Marsh...	japanese	and chinese dishes. we also got a free drink ...	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	Nal
2	f	n	'After I went shopping with some of my friend	we went to DODO restaurant for dinner. I foun...	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	Nal
3	f	n	'Olive Oil Garden was very disappointing. I ex...	and the waitor had no manners whatsoever. Don...	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	Nal
4	f	n	'The Seven Heaven restaurant was never known f...	never more.'	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	Nal

5 rows × 24 columns

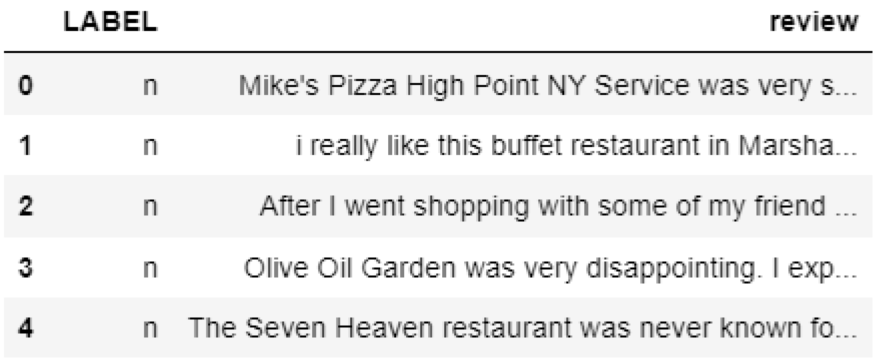
After cleaning up the dataset and splitting the file into two separate dataframes for lie detection and sentiment classification, the finalized dataframes look like this:

Lie Detection dataframe



	LABEL	review
0	f	Mike's Pizza High Point NY Service was very s...
1	f	i really like this buffet restaurant in Marsha...
2	f	After I went shopping with some of my friend ...
3	f	Olive Oil Garden was very disappointing. I exp...
4	f	The Seven Heaven restaurant was never known fo...

Sentiment Classification dataframe



	LABEL	review
0	n	Mike's Pizza High Point NY Service was very s...
1	n	i really like this buffet restaurant in Marsha...
2	n	After I went shopping with some of my friend ...
3	n	Olive Oil Garden was very disappointing. I exp...
4	n	The Seven Heaven restaurant was never known fo...

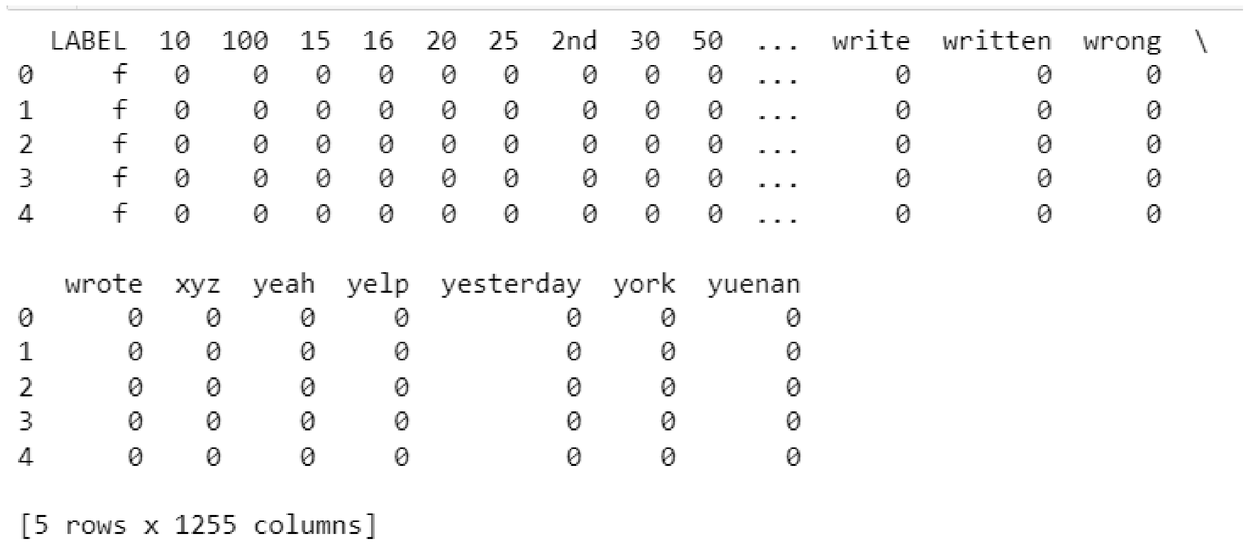
Once the two dataframes were created, the next step was tokenization and vectorization to prepare the data for training a Naive Bayes machine learning model. Details follow.

Models/Methods

Tokenization was carried out with SciKit-Learn's built-in tokenizer and vectorizer methods. Specifically, the CountVectorizer() and TfidfVectorizer() were used to create vectorized dataframes. The vectorizers were used to create the following 4 dataframes (df):

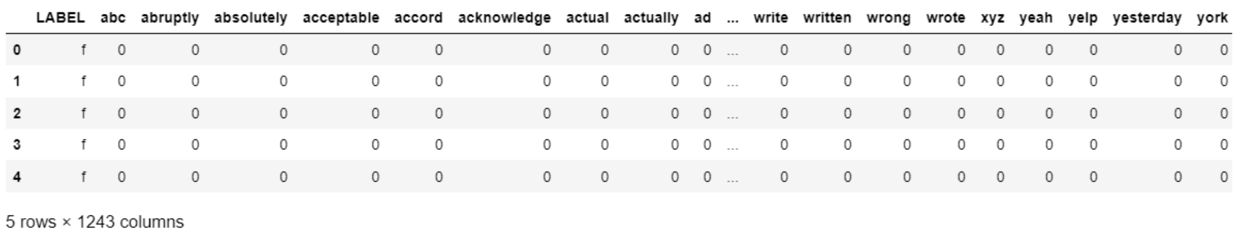
1. a vectorized df of the lie dataset using CountVectorizer,
2. a vectorized df of the lie dataset using TfidfVectorizer,
3. a vectorized df of the sentiment dataset using CountVectorizer, and
4. a vectorized df of the sentiment dataset using TfidfVectorizer.

The vectorizers created dataframes with 1,255 columns representing the vocabularies of the restaurant reviews where each column represents a unique word in the review minus stopwords. The snapshot below is an example of one such dataframe at this stage of preprocessing:



```
  LABEL  10  100  15  16  20  25  2nd  30  50  ...  write  written  wrong  \
0     f   0    0   0   0   0   0    0   0   0  ...      0        0      0
1     f   0    0   0   0   0   0    0   0   0  ...      0        0      0
2     f   0    0   0   0   0   0    0   0   0  ...      0        0      0
3     f   0    0   0   0   0   0    0   0   0  ...      0        0      0
4     f   0    0   0   0   0   0    0   0   0  ...      0        0      0

   wrote  xyz  yeah  yelp  yesterday  york  yuenan
0      0    0     0     0          0     0       0
1      0    0     0     0          0     0       0
2      0    0     0     0          0     0       0
3      0    0     0     0          0     0       0
4      0    0     0     0          0     0       0

[5 rows x 1255 columns]
```

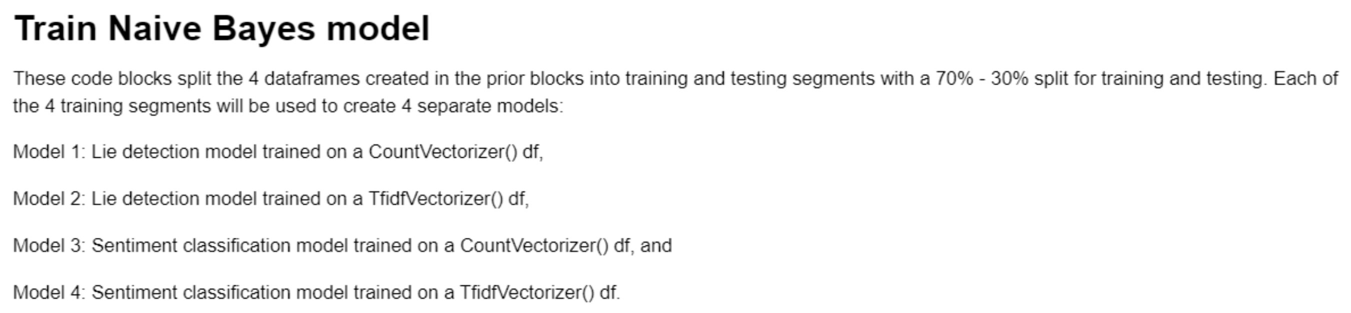
As can be seen in the screenshot, the dataframe includes numeric values such as 10, 100, and 2nd, which do not necessarily add value to the data. Therefore, these numeric columns will be removed, leaving only the columns in the vocabulary that contain alphabetic values:



	LABEL	abc	abruptly	absolutely	acceptable	accord	acknowledge	actual	actually	ad	...	write	written	wrong	wrote	xyz	yeah	yelp	yesterday	york
0	f	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
1	f	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
2	f	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
3	f	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0
4	f	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0

5 rows × 1243 columns

Once the 4 dataframes are created, it is time to train and build the models using the Naive Bayes method. To achieve this, the 4 dataframes are split into training and testing sets, with 70% of the data going to training and 30% going to testing. These data segments will be used to create 4 separate machine learning models. The screenshot below gives more details:



Train Naive Bayes model

These code blocks split the 4 dataframes created in the prior blocks into training and testing segments with a 70% - 30% split for training and testing. Each of the 4 training segments will be used to create 4 separate models:

Model 1: Lie detection model trained on a CountVectorizer() df,

Model 2: Lie detection model trained on a TfidfVectorizer() df,

Model 3: Sentiment classification model trained on a CountVectorizer() df, and

Model 4: Sentiment classification model trained on a TfidfVectorizer() df.

The multinomial Naive Bayes model will be used to create these 4 separate models, and here these 4 models are instantiated in SciKit-Learn:

```python
from sklearn.naive_bayes import MultinomialNB
import numpy as np

MyModel1 = MultinomialNB()
MyModel2 = MultinomialNB()
MyModel3 = MultinomialNB()
MyModel4 = MultinomialNB()
```

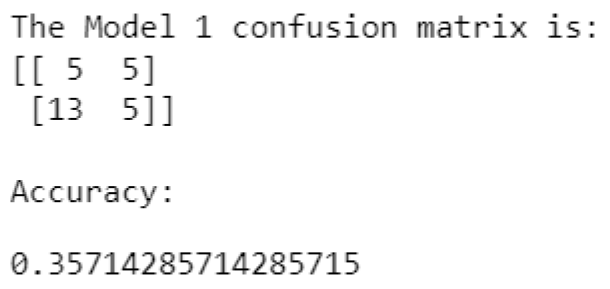
The models are trained using the .fit() method where the training set is used to train each model respectively on the vectorized dataframes.

The next section covers the results of the training and evaluation of these models.

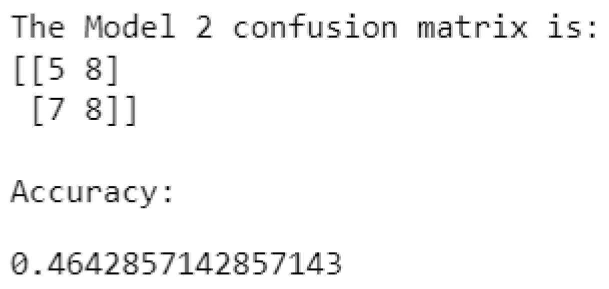
RESULTS

Below are the documented results of the models after training and evaluation. The results include a confusion matrix showing the predicted results vs. the actual labels, along with an overall accuracy metric for each model. Note the discrepancy in accuracy between the Lie Detection models (models 1 and 2) vs. the Sentiment classification models (models 3 and 4). In the world of text mining, determining a fake review from a real review tends to be much more difficult than classifying a review as positive or negative. This is because the linguistic cues for spotting fake reviews are more subtle and nuanced and require a deeper understanding of context compared to categorizing reviews as either positive or negative. Positive and negative reviews are often characterized by the specific vocabulary that falls into either positive words such as ‘good’, ‘great’, ‘tasty’, ‘satisfying’, etc., and negative words such as ‘bad’, ‘awful’, ‘gross’, ‘dirty’, etc. In the feature importance section of this paper, we will examine the most important words that the model used to make determinations of either fake/real or negative/positive.

Lie Detection results

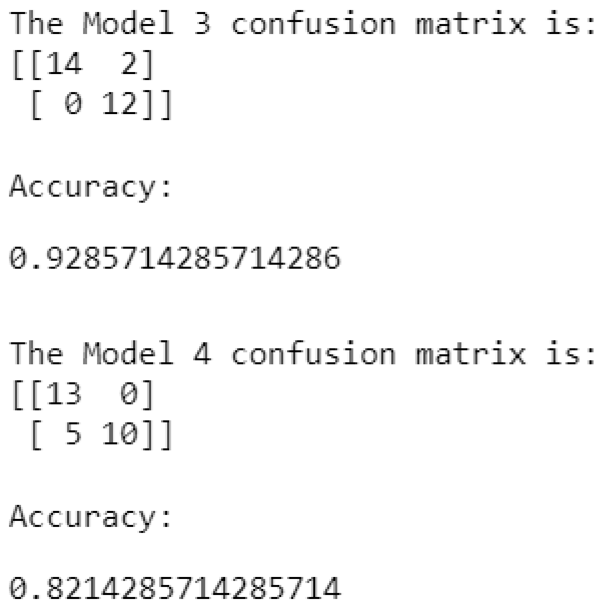


```
The Model 1 confusion matrix is:
[[ 5  5]
 [13  5]]

Accuracy:

0.35714285714285715
```



```
The Model 2 confusion matrix is:
[[5 8]
 [7 8]]

Accuracy:

0.4642857142857143
```

Sentiment Classification results



```
The Model 3 confusion matrix is:
[[14  2]
 [ 0 12]]

Accuracy:

0.9285714285714286

The Model 4 confusion matrix is:
[[13  0]
 [ 5 10]]

Accuracy:

0.8214285714285714
```

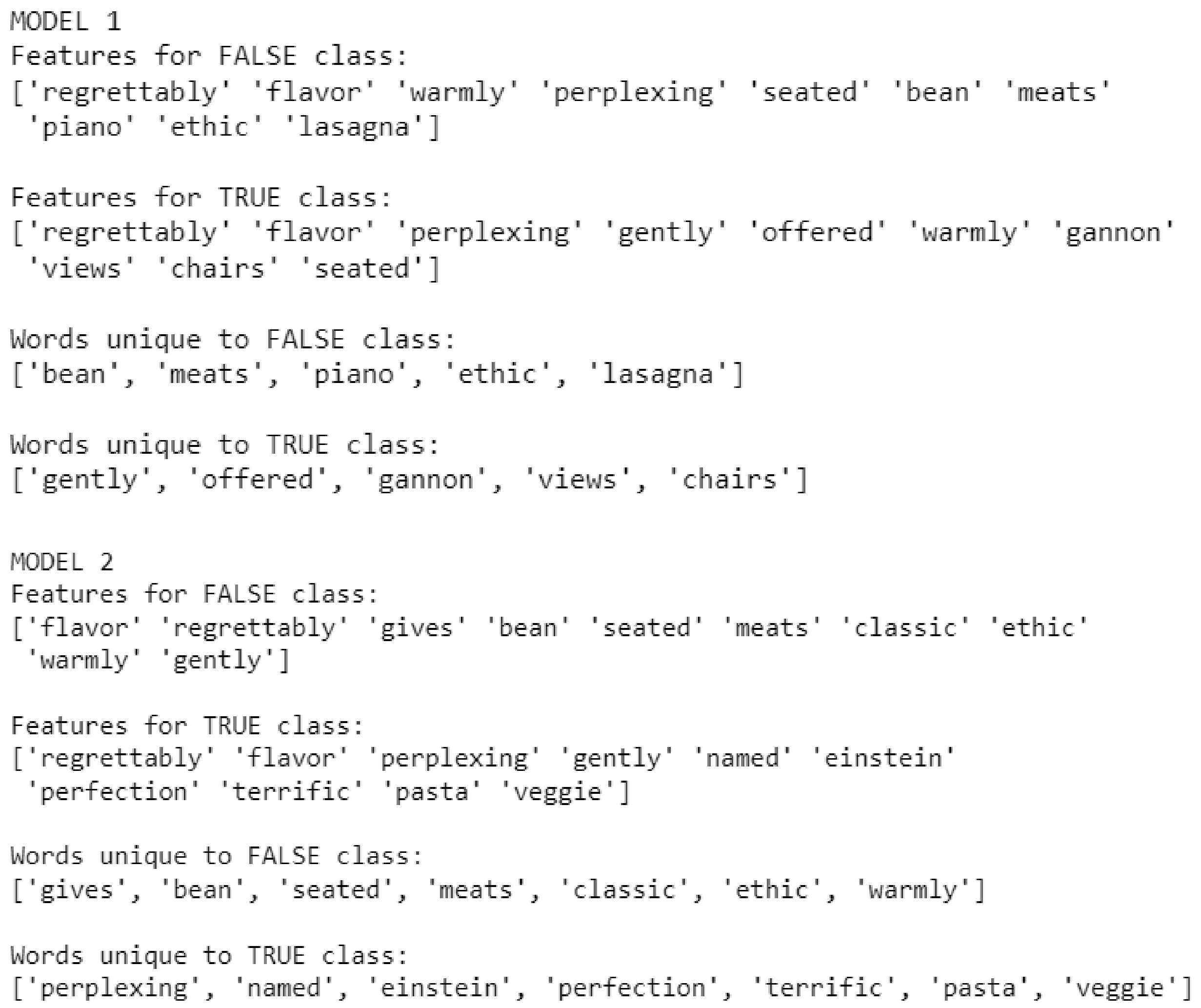
As the results show, the sentiment classification models performed much more strongly than the lie detection models, which is to be expected. In the world of machine learning, an accuracy of 50% is essentially no better than random guessing, so with lie detection results at 35% for the CountVectorizer() model (model 1) and 46% for the TfidfVectorizer() model (model 2), this shows that using the current methods and models does not give reliable results for separating fake reviews from real reviews.

It may be the case that the dataset is simply not large enough to yield meaningful results in the area of lie detection and that a larger dataset may lead to improved accuracy of the models. It may also be that additional feature selection is necessary in the form of additional data preprocessing and removing columns that do not bring value. Other methods could also be employed, such as n-gram analysis of bigrams or trigrams to capture additional context around the words. For example, if a review says that a dish is ‘not good’, a vectorized dataframe would split the words ‘not’ and ‘good’ and treat them as two separate words, with ‘not’ potentially being a negative word and ‘good’ potentially being a positive word. A bigram, on the other hand, may treat ‘not good’ as a single vectorized value, taking the ‘not’ portion into context and categorizing the bigram as a negative value overall.

Feature Importance

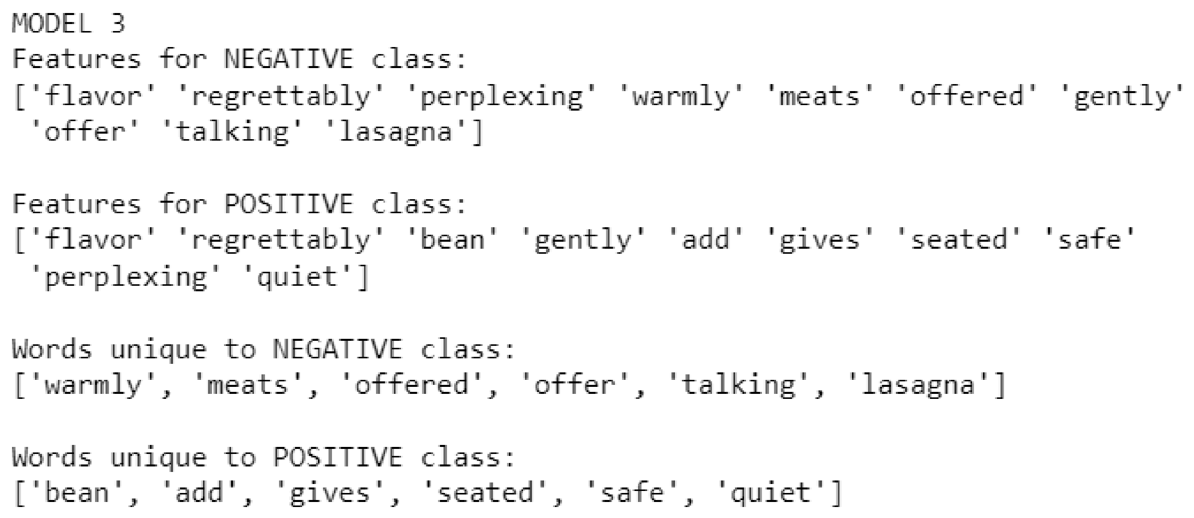
Additional insight into the models' results can be obtained by examining feature importance, whereby the most important words or features are analyzed to help determine why the models made the choices they did.

In the screenshot below, the top 20 features for model 1 are shown, split into the features for the false class and true class (for fake reviews and real reviews, respectively). As can be seen, there is some overlap between the two classes, such as the word 'flavor' showing up in both classes. Therefore, the unique words for each class are also shown to help narrow down the most important word features for each class:

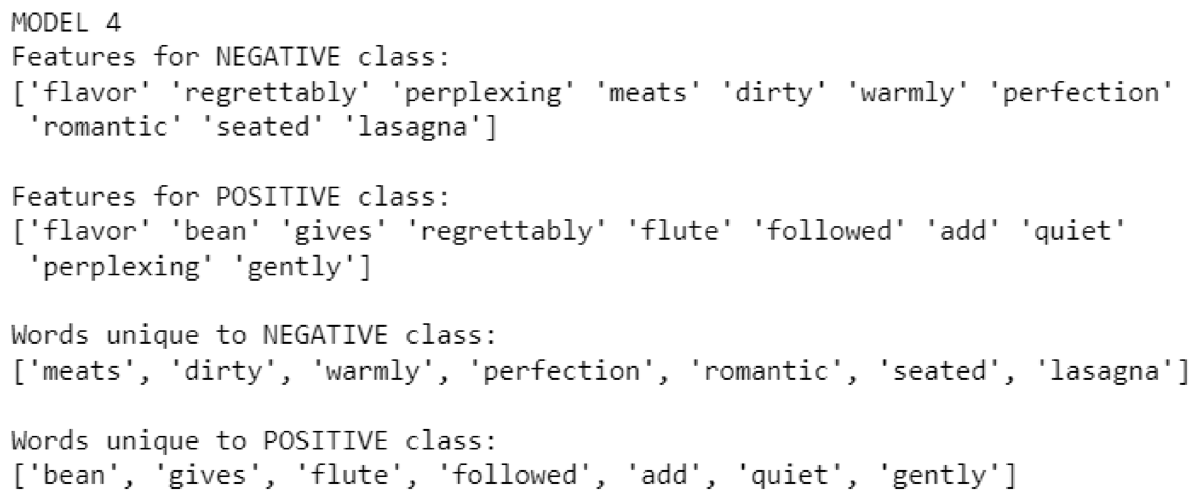


```
MODEL 1
Features for FALSE class:
['regrettably' 'flavor' 'warmly' 'perplexing' 'seated' 'bean' 'meats'
 'piano' 'ethic' 'lasagna']

Features for TRUE class:
['regrettably' 'flavor' 'perplexing' 'gently' 'offered' 'warmly' 'gannon'
 'views' 'chairs' 'seated']

Words unique to FALSE class:
['bean', 'meats', 'piano', 'ethic', 'lasagna']

Words unique to TRUE class:
['gently', 'offered', 'gannon', 'views', 'chairs']


MODEL 2
Features for FALSE class:
['flavor' 'regrettably' 'gives' 'bean' 'seated' 'meats' 'classic' 'ethic'
 'warmly' 'gently']

Features for TRUE class:
['regrettably' 'flavor' 'perplexing' 'gently' 'named' 'einstein'
 'perfection' 'terrific' 'pasta' 'veggie']

Words unique to FALSE class:
['gives', 'bean', 'seated', 'meats', 'classic', 'ethic', 'warmly']

Words unique to TRUE class:
['perplexing', 'named', 'einstein', 'perfection', 'terrific', 'pasta', 'veggie']
```

The same feature importance analysis is then run on the sentiment classification models, with unique words narrowed down for additional insight:



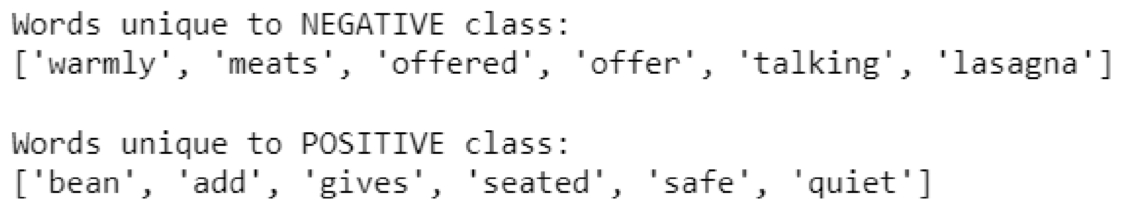
```
MODEL 3
Features for NEGATIVE class:
['flavor' 'regrettably' 'perplexing' 'warmly' 'meats' 'offered' 'gently'
 'offer' 'talking' 'lasagna']

Features for POSITIVE class:
['flavor' 'regrettably' 'bean' 'gently' 'add' 'gives' 'seated' 'safe'
 'perplexing' 'quiet']

Words unique to NEGATIVE class:
['warmly', 'meats', 'offered', 'offer', 'talking', 'lasagna']

Words unique to POSITIVE class:
['bean', 'add', 'gives', 'seated', 'safe', 'quiet']
```



```
MODEL 4
Features for NEGATIVE class:
['flavor' 'regrettably' 'perplexing' 'meats' 'dirty' 'warmly' 'perfection'
 'romantic' 'seated' 'lasagna']

Features for POSITIVE class:
['flavor' 'bean' 'gives' 'regrettably' 'flute' 'followed' 'add' 'quiet'
 'perplexing' 'gently']

Words unique to NEGATIVE class:
['meats', 'dirty', 'warmly', 'perfection', 'romantic', 'seated', 'lasagna']

Words unique to POSITIVE class:
['bean', 'gives', 'flute', 'followed', 'add', 'quiet', 'gently']
```

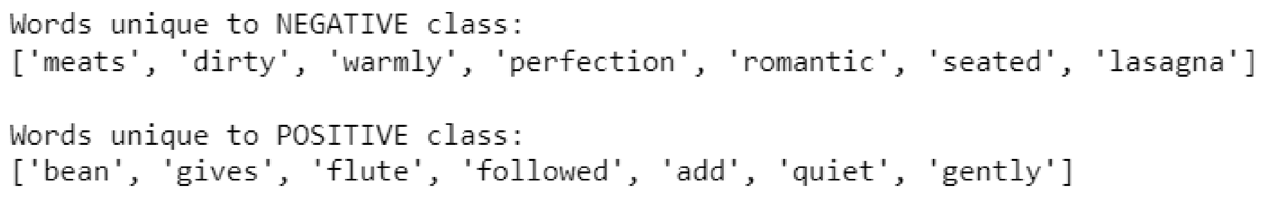
CONCLUSIONS

Starting with categorizing positive from negative reviews, machine learning provides a fairly strong method for this process, achieving a commendable accuracy of around 80%-90%. Furthermore, the analysis revealed that certain words were common to negative reviews and other words were common to positive reviews. For the CountVectorizer() model, these words were the most indicative of negative and positive reviews:



```
Words unique to NEGATIVE class:
['warmly', 'meats', 'offered', 'offer', 'talking', 'lasagna']

Words unique to POSITIVE class:
['bean', 'add', 'gives', 'seated', 'safe', 'quiet']
```

For the TfidfVectorizer() model, these words were the most indicative of negative and positive reviews:



```
Words unique to NEGATIVE class:
['meats', 'dirty', 'warmly', 'perfection', 'romantic', 'seated', 'lasagna']

Words unique to POSITIVE class:
['bean', 'gives', 'flute', 'followed', 'add', 'quiet', 'gently']
```

This suggests that reviews that contain these words may be indicative of either positive or negative reviews, which can be helpful to businesses in categorizing reviews according to their sentiment.

The results for determining fake reviews from real reviews in this exercise were inconclusive due to the low accuracy of the machine learning models. Therefore, the key takeaway is that fake reviews are difficult to determine from real reviews, especially given a relatively small dataset to work with. To extract business value from this particular aspect of the exercise, more sophisticated machine learning methods may be necessary, or at the very least, a larger dataset may be needed to train and build a more robust model. At 50% accuracy, a machine learning model is essentially no better than a random guess, so it brings little business value to addressing the problem.