Sentiment Analysis on Restaurant Reviews for Yelp Data Set

Presentation by: Prianka Ball, Shubhneet Grover

Problem Statement

- Customers are increasingly providing reviews online. Potential customers would read online reviews before visiting any restaurants. Bad reviews can have huge impact on the business

- As the frequency of the reviews have increased, businesses have faced difficulty analyzing every review for their qualitative information. Analyzing reviews can also help them improve their business

Goal

- Analyze the key factors affecting restaurant reviews.
- The clean data would then be incorporated into different Machine Learning Classification models. The best model would be selected for deployment.
- Code was finalized and deployed using FLASK. Final model is able to understand if a review is positive or negative

Approach

- Exploring all data to find patterns.
- Classify review sentiments into positive and negative using star rating. Rating of 4 and 5 were labelled as positive, rating of 1,2,3 were labelled as negative.
- Explore text data separately and was cleaned using processes common in in NLP.
- Use Bag of Words Transformation to prepare data for ML model. We can also tested with Tf-IDF Transformation
- Trained the data using different ML Classification models. Our analysis showed that Naive Bayes model gave the best result .
- Sentiment analysis uses natural language processing (NLP) and machine learning algorithms, to automatically determine the emotional tone behind online conversations. We deployed the model using FLASK.

Methodology

- ➢ **STEP 1**: Downloaded business and review data from Kaggle YELP Review. Joined both datasets. Data was filtered to include reviews from restaurant that are open.
- ➢ **STEP 2**: Selected columns that are necessary for our analysis.
- ➢ **STEP 3**: Filtered data to include reviews written in english.
- ➢ **STEP 4**: Labelled reviews as positive and negative based on review rating.
- ➢ **STEP 5**: Explored text data to find differences between positive and negative reviews, common words(bi-gram, tri grams), common adjectives and nouns.

Methodology

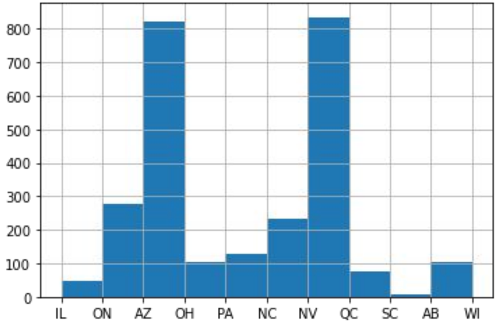
- **STEP 6**: Explored text data to find patterns in the data and identify necessary steps for cleaning data and standardizing the data. We standardizing the data by lowercasing, removing stop words, replacing contracted words .
- **STEP 7**: Tokenized each word and used lemmatization method.
- **STEP 8**: Use Bag of Words transformation to incorporate into Classification ML Models. We have also tested models with TF-IDF transformation
- **STEP 9**: Segment data into test and train
- **STEP 10**: Test with different models: Naive-Bayes, Random Forest, Logistic Regression. Select model with best output.

Tools Used

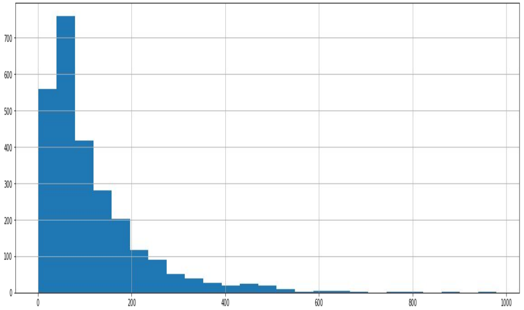
- Python packages used:
  - Nltk: Has all standard algorithms common in NLP
  - String: For string manipulation
  - Collections: Makes collection containers used to store collection of data
  - Langdetect: Detect languages
  - Sklearn: Machine Learning models
  - Matplotlib, seaborn, numpy, pandas
- Jupyter Notebook
- Flask for deployment
- Some css and html while deploying

Exploratory Data Analysis

The dataset had more positive reviews.



Dataset included reviews from 11 different regions



Most reviews were around 100 words.

Exploratory Data Analysis: Word Cloud

Word Cloud with 4 and 5 star rating reviews

Word Cloud with 1, 2, 3 star rating reviews

Word clouds were helpful to give rough idea about the type of words included in the dataset. But they were not that helpful to understand the sentiments or difference in words in positive/negative reviews. Words like food, place , good, restaurant appeared in both positive and negative reviews

Exploratory Data Analysis: Parts of Speech Tagging

Negative Reviews

Adjectives

```
[('good', 285), ('other', 114), ('great', 91), ('nice', 84), ('bad', 69), ('first', 68), ('few', 57), ('little', 57),
('small', 55), ('much', 49), ('busy', 47), ('last', 44), ('many', 41), ('same', 39), ('special', 37), ('decent', 37),
('fresh', 34), ('sure', 33), ('big', 33), ('friendly', 31), ('different', 31), ('ok', 31), ('next', 31), ('deliciou
s', 30), ('terrible', 30), ('hot', 29), ('high', 29), ('tasty', 29), ('whole', 27), ('dry', 27)]
```

Nouns

```
[('food', 422), ('place', 241), ('service', 237), ('time', 202), ('order', 164), ('restaurant', 121), ('menu', 87),
('table', 84), ('server', 73), ('sushi', 70), ('experience', 67), ('chicken', 66), ('sauce', 60), ('meat', 59), ('mea
l', 59), ('pizza', 59), ('rice', 58), ('nothing', 57), ('way', 56), ('drink', 55), ('dinner', 53), ('location', 52),
('staff', 52), ('salad', 52), ('i', 50), ('night', 49), ('water', 49), ('lunch', 48), ('waitress', 48), ('something',
47)]
```

Positive Reviews

Adjectives

```
[('good', 259), ('great', 220), ('fresh', 77), ('delicious', 73), ('little', 71), ('friendly', 70), ('nice', 68), ('o
ther', 53), ('few', 53), ('first', 48), ('amazing', 47), ('favorite', 43), ('awesome', 41), ('excellent', 36), ('smal
l', 35), ('many', 34), ('next', 34), ('much', 34), ('new', 33), ('super', 33), ('sure', 32), ('different', 31), ('bi
g', 31), ('tasty', 30), ('Good', 28), ('clean', 26), ('only', 26), ('hot', 26), ('last', 24), ('perfect', 23)]
```

Nouns

```
[('place', 307), ('food', 262), ('service', 131), ('time', 121), ('order', 76), ('chicken', 76), ('menu', 74), ('rest
aurant', 73), ('staff', 68), ('dinner', 58), ('sushi', 58), ('night', 56), ('pizza', 53), ('lunch', 51), ('salad', 5
1), ('sauce', 48), ('meal', 48), ('bar', 46), ('rice', 45), ('day', 44), ('experience', 44), ('area', 44), ('way', 4
3), ('side', 41), ('spot', 37), ('soup', 37), ('price', 37), ('roll', 37), ('cream', 37), ('meat', 36)]
```

Finding most common nouns and adjectives for positive and negative reviews did not give a lot of insight into the data types

Exploratory Data Analysis: N gram Analysis

N-grams are contiguous sequences of n-items in a sentence.

In Natural Language Processing, N-grams as strings of words, where n stands for an amount of words that you are looking for.

The following types of N-grams are usually distinguished:

- **Unigram:** An N-gram with simply one string inside.
- **2 gram or Bigram:** Typically a combination of two strings or words.
- **3 gram or Trigram:** An N-gram containing up to three elements that are processed together.

Exploratory Data Analysis: N gram Analysis +ve Review

Word	Frequency
food	1177
good	1089
great	1068
place	990
service	658
like	512
time	442
delicious	423
best	418
really	417
love	408
amazing	385
restaurant	372
chicken	349
definitely	333
try	328
menu	328
nice	327
got	324
ordered	320

bigram	frequency
ice cream	82
highly recommend	81
really good	78
great food	75
great service	75
food great	74
food good	70
service great	62
great place	56
love place	54
good food	52
happy hour	44
customer service	43
staff friendly	42
pretty good	40
food service	39
recommend place	39
good service	37
service good	36
las vegas	36

trigram	frequency
great food great	19
food great service	18
highly recommend place	15
love love love	11
french onion soup	10
food good service	10
corned beef hash	9
definitely recommend place	7
great food service	7
overall good experience	7
service great food	7
sweet potato fries	7
great customer service	7
service excellent food	6
overall great experience	6
good food good	6
good food great	6
staff super friendly	6
seated right away	6
excellent customer service	6

- Bi-grams and trigrams gives the most context into the text data we are working with.
- Most positive reviews seems to be around good food and service

Exploratory Data Analysis: N gram Analysis -ve Review

Word	Frequency
food	786
good	509
place	421
service	413
like	366
time	319
ordered	267
order	264
really	231
got	228
came	221
restaurant	215
chicken	189
great	184
went	173
table	167
minutes	159
better	154
people	147
nice	147

bigram	frequency
food good	49
pretty good	45
customer service	32
10 minutes	30
ice cream	28
tasted like	26
long time	24
fried rice	24
feel like	23
good food	23
food service	22
fast food	22
15 minutes	21
chips salsa	21
20 minutes	21
really good	21
dim sum	18
food pretty	17
food ok	17
good service	17

trigram	frequency
food pretty good	13
hand pulled noodles	6
really wanted like	6
waited 10 minutes	6
wanted like place	6
yes yes yes	5
hot sour soup	5
waste time money	4
overall food good	4
pretty good food	4
ice cream sandwich	4
waited long time	4
savory stuffed dumplings	4
wait 10 minutes	4
waited 20 minutes	4
20 minute wait	4
french onion soup	4
food good quality	4
food good service	4
really looking forward	4

- Similar to positive reviews, bi-grams and tri-grams also give more context to negative reviews.
- Most customer seems to leave negative reviews because of late service.

Exploratory Data Analysis:

Stop Words, Punctuation, Shortened words, Foreign words

```
[('.', 16573),
 ('the', 11255),
 (',', 9430),
 ('and', 9236),
 ('I', 7373),
 ('a', 6495),
 ('to', 5631),
 ('was', 5385),
 ('of', 3776),
 ('it', 3554),
 ('is', 3543),
 ('!', 3476),
 ('for', 3111),
 ('The', 2801),
 ('in', 2641),
 ('with', 2204),
 ('that', 2112),
 ('but', 2049),
 ('you', 1898),
 ("n't", 1814),
 ('food', 1789),
 ('on', 1767),
 ('had', 1638),
 ('we', 1624),
 ("'s", 1623),
 ('my', 1613),
 ('were', 1585),
 ('this', 1552),
 ('have', 1526),
 ('good', 1477),
 ('not', 1410),
 ('place', 1396),
 ('are', 1353),
 ('they', 1280),
```

```
('so', 1256),
('at', 1229),
(')', 1189),
('be', 1132),
('We', 1097),
('(', 1064),
('as', 1006),
('great', 963),
('here', 958),
('very', 944),
('service', 907),
('like', 854),
('there', 853),
('It', 847),
('our', 844),
('just', 801),
('out', 779),
('one', 761),
('time', 759),
('all', 755),
('back', 746),
('get', 744),
('would', 724),
('their', 696),
('me', 675),
('if', 669),
('go', 646),
('do', 643),
('...', 640),
('did', 639),
('which', 625),
('or', 617),
('from', 612),
('really', 606),
('up', 583),
```

- After tokenizing words, we found high frequency among words(eg. the, and, of) which did not add value to our analysis. These are called stop words.
- There were also high number of punctuations and shortened words.
- Our analysis also found review written in 5 different languages.

```
language
de          2
en       2621
es          2
fr          5
no          1
zh-cn       1
Name: text, dtype: int64
```

Cleaning Data/Pre-Processing steps

Text data is highly unstructured and requires a lot of preprocessing to gain any insight or feed into any models. After exploring the data, we decided to pre-process the data in the following ways:

- **Language** : Removing review that are not english
- **Removing stopwords**: Stopwords are the words in any language which does not add much meaning to a sentence.
- **Removed Punctuation**
- **Lowercasing the words**
- **Replacing shortened words**

Feature Engineering

- **Word Tokenization:** Tokenization is a way of separating a piece of text into smaller units called tokens.
- **Lemmatization:** Method of removing the suffix of the word and bringing it to a base word.
- **Bag of Words Transformation:** The bag-of-words model is a simplifying representation used in natural language processing and information retrieval (IR). In this model, a text (such as a sentence or a document) is represented as the bag (multiset) of its words, disregarding grammar and even word order but keeping multiplicity.
- **TF-Idf Transformation:** TF-IDF (term frequency-inverse document frequency) works by increasing proportionally to the number of times a word appears in a document, but is offset by the number of documents that contain the word. So, words that are common in every document, such as this, what, and if, rank low even though they may appear many times, since they don't mean much to that document in particular.

Feature Engineering: Lemmatization

Negative reviews

	Word	Frequency
0	food	793
1	order	605
2	good	518
3	place	497
4	time	424
5	servic	422
6	like	401
7	restaur	257
8	tabl	241
9	tri	239
10	wait	232
11	realli	231
12	got	228
13	came	221
14	come	220
15	ask	217
16	eat	206
17	want	191
18	chicken	190
19	drink	190

Positive reviews

	Word	Frequency
0	food	1189
1	place	1128
2	good	1109
3	great	1069
4	servic	669
5	order	661
6	like	590
7	time	578
8	love	572
9	tri	548
10	restaur	445
11	delici	433
12	best	418
13	realli	417
14	come	411
15	amaz	395
16	chicken	349
17	nice	340
18	menu	340
19	definit	339

- Stemming words led word endings to be removed.
- It did not add value to the overall final model.

Techniques and Algorithms

Naïve baes - It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors.

Random forest - Random forest is a type of supervised machine learning algorithm based on ensemble learning. Ensemble learning is a type of learning where you join different types of algorithms or same algorithm multiple times to form a more powerful prediction model.

Logistic regression - Logistic regression is a class of regression where the independent variable is used to predict the dependent variable

Output: Bag of Words Classifier

Naive Bayes Classifier

```
[[511  29]
 [ 99 148]]
              precision    recall  f1-score   support

           0       0.84      0.95      0.89       540
           1       0.84      0.60      0.70       247

    accuracy                           0.84       787
   macro avg       0.84      0.77      0.79       787
weighted avg       0.84      0.84      0.83       787

0.8373570520965693
```

Random Forest

```
[[521  19]
 [128 119]]
              precision    recall  f1-score   support

           0       0.80      0.96      0.88       540
           1       0.86      0.48      0.62       247

    accuracy                           0.81       787
   macro avg       0.83      0.72      0.75       787
weighted avg       0.82      0.81      0.80       787

0.8132147395171537
```

Logistic Regression

```
[[540    0]
 [247    0]]
              precision    recall  f1-score   support

           0       0.69      1.00      0.81       540
           1       0.00      0.00      0.00       247

    accuracy                           0.69       787
   macro avg       0.34      0.50      0.41       787
weighted avg       0.47      0.69      0.56       787

0.6861499364675985
```

- Naive Bayes Classifier had the highest accuracy score among all model.

Output: Tf-Idf Classifier

Naive Bayes Classifier

```
[[476  64]
 [ 85 162]]
              precision    recall  f1-score   support

           0       0.85      0.88      0.86       540
           1       0.72      0.66      0.68       247

    accuracy                           0.81       787
   macro avg       0.78      0.77      0.77       787
weighted avg       0.81      0.81      0.81       787

0.8106734434561627
```

Random Forest

```
[[521  19]
 [128 119]]
              precision    recall  f1-score   support

           0       0.80      0.96      0.88       540
           1       0.86      0.48      0.62       247

    accuracy                           0.81       787
   macro avg       0.83      0.72      0.75       787
weighted avg       0.82      0.81      0.80       787

0.8132147395171537
```

Logistic Regression

```
[[540   0]
 [247   0]]
              precision    recall  f1-score   support

           0       0.69      1.00      0.81       540
           1       0.00      0.00      0.00       247

    accuracy                           0.69       787
   macro avg       0.34      0.50      0.41       787
weighted avg       0.47      0.69      0.56       787

0.6861499364675985
```

When we used TF-IDF Classifier, accuracy score from Random Forest model and Logistic Regression stayed the same.

Accuracy score of Naive Bayes Model decreased.

Output: Using n-gram & Naive Bayes Model

2-gram BOW

```
[[504  36]
 [139 108]]
              precision    recall  f1-score   support

           0       0.78      0.93      0.85       540
           1       0.75      0.44      0.55       247

    accuracy                           0.78       787
   macro avg       0.77      0.69      0.70       787
weighted avg       0.77      0.78      0.76       787

0.7776365946632783
```

3-gram BOW

```
[[505  35]
 [202  45]]
              precision    recall  f1-score   support

           0       0.71      0.94      0.81       540
           1       0.56      0.18      0.28       247

    accuracy                           0.70       787
   macro avg       0.64      0.56      0.54       787
weighted avg       0.67      0.70      0.64       787

0.6988564167725541
```

2-gram Tf-Idf

```
[[463  77]
 [104 143]]
              precision    recall  f1-score   support

           0       0.82      0.86      0.84       540
           1       0.65      0.58      0.61       247

    accuracy                           0.77       787
   macro avg       0.73      0.72      0.72       787
weighted avg       0.76      0.77      0.77       787

0.770012706480305
```

3-gram Tf-Idf

```
[[474  66]
 [184  63]]
              precision    recall  f1-score   support

           0       0.72      0.88      0.79       540
           1       0.49      0.26      0.34       247

    accuracy                           0.68       787
   macro avg       0.60      0.57      0.56       787
weighted avg       0.65      0.68      0.65       787

0.6823379923761118
```

Using bi-gram and tri-gram lowered the accuracy score of the model. Using tf-idf transformation lowered the score further.

Output: Using stemming, 1-gram & Naive Bayes

BOW

```
[[510  30]
 [100 147]]
              precision    recall  f1-score   support

           0       0.84      0.94      0.89       540
           1       0.83      0.60      0.69       247

    accuracy                           0.83       787
   macro avg       0.83      0.77      0.79       787
weighted avg       0.83      0.83      0.83       787

0.8348157560355781
```

Tf-Idf

```
[[469  71]
 [ 81 166]]
              precision    recall  f1-score   support

           0       0.85      0.87      0.86       540
           1       0.70      0.67      0.69       247

    accuracy                           0.81       787
   macro avg       0.78      0.77      0.77       787
weighted avg       0.80      0.81      0.81       787

0.806861499364676
```

Stemming did not improve the model so we decided to leave it out in the final iteration.

Final Model Selection

The best model happens when we replace contractions with long form, convert words to lowercase, remove unwanted characters, remove stop words and use Bag of Words transformation.

Using 1-gram and Naive Bayes model gives the highest accuracy score. The model also used used alpha = 0.9 and fit_priori = FALSE

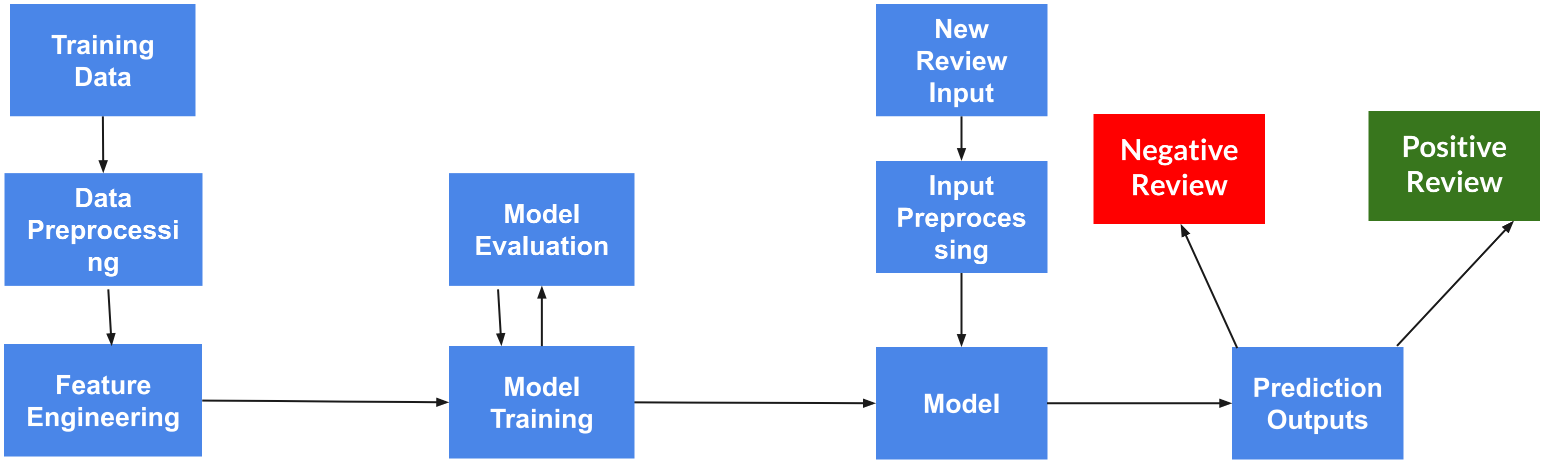
- Using tf-idf transformation lowers the accuracy score.
- Using more than 1-gram lowers accuracy score.
- Using logistic regression or random forest lowers the score too.
- Increasing the training data have shown to improve model performance

```
[[502  38]
 [ 85 162]]
              precision    recall  f1-score   support

           0       0.86      0.93      0.89       540
           1       0.81      0.66      0.72       247

    accuracy                           0.84       787
   macro avg       0.83      0.79      0.81       787
weighted avg       0.84      0.84      0.84       787

0.843710292249047
```

Conclusion and Future Scope

- Compare Naive Bayes Classification model against models like VADER
- Use deep learning algorithms for sentiment analysis and compare results.
- Current model does not understand negated words(eg. not good)
- Increase data size and explore how the model precision changes.
- Train model on equal number of negative and positive reviews

Model Deployment: Flow Diagram



Model Deployment: Techniques

- Build a machine learning model for Sentiment Detection of Restaurant reviews, then created an API for the model using Flask. Flask is a Python micro framework to build web application.
- Used predictive capabilities through HTTP requests.
- Our system has the following files:

```
Yelp_academic_dataset_business.json
yelp_academic_dataset_review.json
app.py
templates/
        home.html
        result.html
static/
        style.css
```

- **templates:** directory where Flask looks for static HTML files to render in web browser
- **static:** Formats the HTML files further
- **app.py :** Contains all the main code that will be executed by Python to run the Flask web app
- Go to the directory and run the app by typing *python app.py* on the terminal.

Model Demo

Developed simple web page with a form that fields to let you write your restaurant review. The message is submitted and it goes to a new page that shows if it is a negative or positive review