In the document below we explain the features that were used and tested in order to produce our model. We briefly touch on the different classification models that were used and the results we obtained from the different models. In the last section we include references to the resources that were used:

1. System Description
   (a) Pre-Processing
   In preprocessing the sentences we accounted for negation by tagging each word having a negation with a _NEG.
   (b) Feature List
      i. NetPositive/Negative/Neutral: This feature consists of the count of Positive, Negative and Neutral Words (using the sentiment lexicon)

      ii. NetPositive/Negative/NeutralNormalized: This feature is similar to the one described earlier but is normalized by dividing by the Sentence Length.

      iii. NetSentimentNormalizedAvg: This feature contains the average of the count of Positive Words and Number of Negative Words. Here we multiplied the total number of negative words by -1. This provides us with an approximation of how many negative or positive words there are on a Net basis.

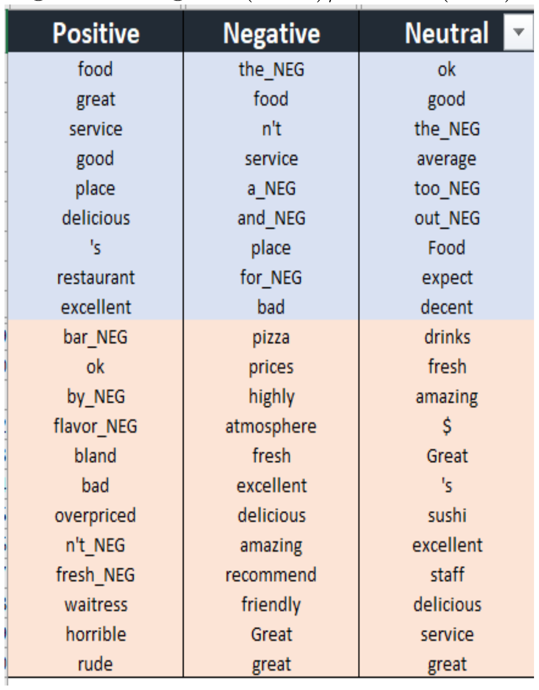
      iv. NetPositive/Negative/ NeutralLikelehood: This feature is based on the naive bayes formula: $P(c|x) = \frac{P(x|c)P(c)}{P(x)}$
      Whereby x represents a word and c a class. In computing this feature we sum up all the probabilities computed for each sentence based on the words in each sentences and produced a feature corresponding to each class.

      v. Z Score Positive/Negative/ Neutral: The Z-Score for each word is calculated as described in Olena Kummer's paper "Feature Selection in Sentiment Analysis":

      $Zscore(f) = \frac{a-n'\cdot P(f)}{\sqrt{n'\cdot P(f)\cdot(1-P(f))}}$

      Where a represents total number for a given word belonging in class f (positive/negative or neutral).
      n' represents the total occurences for that same word in all classes.
      $P(f)$ represents the probability of seeing that word in the corpus.
      In figure 1 Top Z-Score shows the the Top Z-Score for the words belonging in the corpus.
      After calculating the Z-Scores for all the words we loop on the words in a sentence and sum the Z-Scores for each word in order to produce three features.

Figure 1: Highest(Blue)/Lowest(Red) Z Scores



Positive	Negative	Neutral
food	the_NEG	ok
great	food	good
service	n't	the_NEG
good	service	average
place	a_NEG	too_NEG
delicious	and_NEG	out_NEG
's	place	Food
restaurant	for_NEG	expect
excellent	bad	decent
bar_NEG	pizza	drinks
ok	prices	fresh
by_NEG	highly	amazing
flavor_NEG	atmosphere	$
bland	fresh	Great
bad	excellent	's
overpriced	delicious	sushi
n't_NEG	amazing	excellent
fresh_NEG	recommend	staff
waitress	friendly	delicious
horrible	Great	service
rude	great	great

vi. Net Positive/Negative/Neutral Distance: We created a distance table that incorporates the inverse of the distance of each word in a sentence from the target word. Therefore, farther words from the target word would have a lower inverse distance. The Positive/Negative/Neutral Distance consists of the score defined in i. weighted by the distance distance to the target word.

vii. Z Positive/Negative/Neutral Distance: These features are similar to those defined in v. but weighted by the distance.

viii. Naive Positive/Negative/Neutral: This feature uses the Naive based classifier on a sentence by multiplying all the probabilities defined under iv. in a given sentence. Under this feature, rather than taking the argmax to decide on the probability of a sentence having a positive, negative or neutral sentiment, we calculate the Baye's score for each class and divide by the total Baye's score among the all the classes. This gives us an idea about how confident is our model of a sentence to belong in a given class.

ix. Exclamation Mark Count: This feature is simply the count of occurences of exclamation marks in a given sentence.

x. Question Mark Count: This feature is simply the count of occurences of question marks in a given sentence.

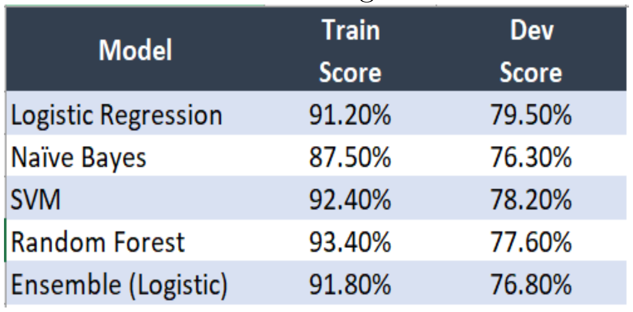
xi. Upper Count: This feature is simply the count of upper case words in a given sentence. Note that an upper case word is defined as having more than 1 letter in upper case.

xii. Final Score Adj/Adv: This feature find the adjective/adverb related to the target word (using POS tagging) in a given sentence and assigns the adjective/adverb a score of +1 if it's positive and -1 if it's negative. The feature will in the end be the sum of these scores. The score is normalized by the number of adjective/adverbs in a given sentence.

xiii. TF-IDF and Vectorizer: Note that we also tried the TF-IDF and Vectorization of all the words as features, but opted to not include them in the code because the sparcity of the matrix was negatively affecting the classifier.
Please note that most of the features noted above appear twice in the dataframe feature table because they are computed based on the complete sentence and the partioned sentence. Partionining of a sentences is done using the following words: "but","yet","however","nonetheless","." In partioning, these trigger words are used to extract the part of the sentence related to the target word.

(c) Models Employed
In the final model we used logistic regression as it gave the best score on the dev set. The below table summarizes the total accuracy scores we obtained on the dev set for each of the models used.

Figure 2: Score



Model	Train Score	Dev Score
Logistic Regression	91.20%	79.50%
Naïve Bayes	87.50%	76.30%
SVM	92.40%	78.20%
Random Forest	93.40%	77.60%
Ensemble (Logistic)	91.80%	76.80%

In the ensemble model, we included the predictions from all the models as features as well as the aspect-category as a dummy variable in the feature matrix. Then we used logistic regression and obtained a Dev Set accuracy of 76.8%.
Please note that we used Grid Search in most of the models in order to best calibrate the hyperparamaters of each classifier.
Also note that our scores are slightly different from those obtained in the assignment scoring due to use of a different random state.

(a) Resources Used:
For Lexicon, we compiled the lexicons from several websites:

i. https://www.cs.uic.edu/ liub/FBS/sentiment-analysis.html

ii. https://github.com/stepthom/lexicon-sentiment-analysis

References

[1] Olena Kummer and Jacques Savoy. Selection in Sentiment Analysis CORIA 2012

[2] Raisa Varghese and Jayasree M. Aspect Based Sentiment Analysis using Support Vector Machine Classifier 2013 International Conference on Advances in Computing

[3] Oscar Araque and Ganggao Zhu. Mining the Opinionated Web: Classification and Detection of Aspect Contexts for Aspect Based Sentiment Analysis. Univeridad Politecnica de Madrid