Part 1: Learning from additional information only with random forest

The first method I tried is to use provided additional information only and learn with random forest.

Here I ignored labels (the first column) for two reasons:

1. I personally don't want to relate hygiene issue with a particular cuisine.
2. There are two ways of dealing with this feature. One is to flatten it as binary vectors, which makes the dimension of features huge. The other is to use word embedding to reduce dimension. But I will use word embedding processing for reviews, so I decide not to repeat for the implementation here.

The final random forest configs are:

1. Max_depth = 20
2. N_estimators= 10

The final F1 score is 0.5331

Third party library used:

1. Pandas
2. Scikit-learn

Part 2: Learning from reviews only with CNN

The second method I tried is to use reviews only, to train a neural network.

To construct input of CNN, I used following method:

1. Use training and testing documents together to train a word2vec model.
2. Use reviews to construct a matrix:
    a. Each line is a collection of reviews divided by blank (" "), separate as sentences.
    b. For each sentence, look up each word in each sentence for its corresponding vector, and construct a matrix for this sentence
    c. Use opencv to resize this matrix as a picture to a fixed size.
    d. Average all sentences for particular restaurant, and use this matrix as input of cnn
3. Construct CNN:
    a. Conv1: sequential
        i. A Conv2d layer: (in_channel: 1, out_channel: 16, kernel_size:5, padding:2)
        ii. Activation layer: ReLU
        iii. MaxPooling layer: (kernel_size:2)
    b. Conv2: Sequential
        i. A Conv2d layer: (in_channel: 16, out_channel: 32, kernel_size:5, padding:2)

ii. Activation layer: ReLU
iii. MaxPooling layer: (kernel_size:2)

c. Out: Linear(input: size of last layer, output: 2)

There was another method of constructing feature vector, which is to average all words in a line of review. I tried in a train-validation, but accuracy is low, because there are sometimes more than 10 thousands words in each line, and averaging them in one-dimensional vector makes them not very distinguishable, and it eliminated the "word order" information. So I decide to use 2d matrix as input.

I tried different vector length when modeling word2vec model, and the final result of NN output are:

Vector size	Validation	Test
10	0.51	0.56
20	0.46	0.55
50	0.53	0.38
100	0.47	0.31

So the final F1 score I got after this part is 0.55

Part3: Combination of reviews and additional information

In the end, I tried to combine reviews in part2 and additional information in part1.

I used CNN to generate a 10-dimensional output for each line of review, and combine this vector together with additional information used in part 1 as larger input features for random forest.

The configs of CNN and random forest are same as previous sections except that output layer of CNN is a 10-dimensional vector.

Vector size	F1 score
10	0.48
20	0.60
50	0.59
100	0.58

So the best score I achieved in this competition is 0.60

Bunni	0.6009	*0.5892*	0.603	*0.5886*	0.5977	*0.59*	2019-07-28 | 06:06:15	26