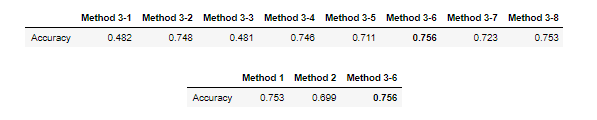
Sarah Ganci

INLS 613

11/26/2018

Homework 3

Note: All my work is in the python notebook that I have also attached in my submission. I deleted two rows of input that had special characters because they were causing issues when I tried to read in the file using pandas.read\_csv(), so the results and accuracy may be slightly skewed.

1. Methods:
   1. Method 1: Score by Majority
      1. Description:
         1. I created a function “majority\_wins” that takes in a row from the DataFrame generated from music.train.annotators.csv. This function counts the number of positive and negative ratings the row has and returns a “0” if the row has more negative than positive or “1” if the row has the same number or more positive than negative.
         2. I then created another function “majority\_wins\_convert”, which generates one label for every row in the DataFrame using the “majority\_wins” function.
      2. Motivation:
         1. The motivation for this method is simple: if the majority of annotators agree on a label, then that label is probably the correct label. This method does not account for inter-annotator agreement or random chance, but it should serve at least as a good baseline.
      3. Accuracy and Explanation:
         1. I tested the accuracy of this method by creating a TFIDF vector based on the method 1 labels and text and training a naïve bayes model on that vector and labels. I then tested the accuracy of that model using the provided test data. The resulting accuracy was: 0.753. I suspect that this set of labels worked well because of the reasons posed in the motivation—if the majority chooses a particular label, then that label should be the correct label. This particular method could be improved to reflect inter-annotator agreement and the effects of random chance. Additionally, this method could be improved to handle the cases where there is no winner—the current method labels ties as positive, which could skew the results.
   2. Method 2: Instance Weighing
      1. Description:
         1. I created a function “confidence” that returns a tuple containing the majority winning label along with the number by which that label won (i.e. if there were 6 “negative” annotations, the output would be [0, 4]).
         2. I created another function “confidence\_convert” that takes in the DataFrame of the annotators and calls “confidence” on each row. The function returns a tuple containing the weighted text data and y labels. One row of text and one label y is added to the output text and labels the number of times that the label won for that row (i.e.: if a row gets the confidence score of [0,4] a “0” would be added 4 times to the y labels part of the output and the text for that row would be added 4 times to the text part of the output.
      2. Motivation:
         1. The duplication of entries which have a higher confidence score should accomplish the goal of weighing the more certain entries (with higher annotator agreements) and eliminating entries with no winning label. Hypothetically, a model trained on more instances that have a higher level of certainty of being positive or negative will perform better.
      3. Accuracy and Explanation:
         1. I tested the accuracy of this method by creating a TFIDF vector based on the method 2 text output containing duplicate entries. I then trained a naïve bayes model on that vector and method 2 generated labels. I tested the accuracy of that model using the provided test data. The resulting accuracy was: 0.699. I suspect that this model did worse than the majority wins model of method 1 because the model overfit to the training data (i.e. adding the duplicates of entries up to 8 times could have overfit the model to the training data).
   3. Method 3: Top Annotators
      1. Description:
         1. I first explored the consistency of each annotator:
            1. I created a function “cross\_val” that computes the 10 cross validation score/
            2. I created a tfidf vector based on the text column of the annotators data
            3. I computed the cross-validation score using a naïve bayes model, the same tfidf vector, and each annotator’s set of labels.
            4. I sorted the annotators by their cross-validation scores.
         2. I modified my confidence and majority wins functions to take in an array of annotators, such that the label is computed based on the labels only from that set of passed in annotators.
         3. I computed the cross-validation scores for the top 1-8 annotators.
         4. The best score came from selecting the lone top annotator (annotator 8).
      2. Motivation:
         1. Hypothetically, the combination of the individual annotators whose labels achieve high accuracy should (when combined) generate a model that can achieve a high accuracy. Eliminating inconsistent annotators whose labels generate low accuracy should improve the model.
      3. Accuracy and Explanation:
         1. Using the labels from top annotator (annotator 8) generated a model with very low accuracy—0.482 (yikes). This could have happened because the model could be over fit to the training data. Maybe annotator 8’s labels just happened to work really well on the training data. Out of curiosity (and for comparison purposes), I also created label sets for the other top n annotators. I found that using the labels from the top 6 annotators to generate the one true label allowed for the highest accuracy score over all—0.756. Perhaps some noise was reduced by eliminating the lowest two annotators (4 and 6).
2. Discussion: taking into consideration everything you tried and whether or not it worked, provide a discussion of your overall results. Did you notice any trends? Do you have any ideas for why these trends occurred? What did you learn?
   1. Resulting Accuracies: The best accuracy came from the model trained by the labels generated using Method 3 with the top 6 annotators. The model trained by Method 1’s labels had a very similar accuracy score. The worst accuracy came from the model trained on the “best” annotator’s labels. The ideas of weighing instances by their confidence and selecting the best annotators were appealing because they posed solutions to the issues of close votes/random chance and eliminated poorly trained annotators with bad judgement. However, both methods (method 2 and most of method 3) generated models that were overfit to the training data. The best two models (models trained based on labels generated from Methods 1 and 3-6) have enough opinions to generate potentially correct labels—both of these methods avoid overfit by not relying on just one annotator’s judgements.
   2. Trends: I didn’t notice any trends in the contents data specifically, but I did notice the aforementioned trend of overfitting. As I have suggested already, the models generated from method 2 and most of method 3’s trials overfit to the training dataset. Method 2 potentially led to overfitting because too many duplicates of the training dataset were added. Method 3 (especially the trials with fewer annotators) led to the generation of an over fit model because an incorrect assumption was made—while annotator 8’s labels had a high cross validation score on the training data, that does not mean that the labels were correct or that the model trained on those labels will correctly generate labels for all data (or the test data). Additionally, we cannot assume (as I did) that combining the labels of the top scoring annotators will lead to a high scoring set of labels—we cannot assume that these high scoring annotators agree or that their annotations are correct to begin with.
   3. What I Learned: From this assignment, I learned more about how to use ML tools (particularly python, jupyter notebook, pandas, and sklearn). I also learned how to apply the principles we have learned in class (inter-annotator agreement, training and testing models, cross validation, etc.) to a real-life situation (determining the true labels for a dataset).