Task 5: Cross-validation and test sets.

Step 1: Data split

Split your training and validation dataset into 10 equal folds in two different ways.

1. Random: Assign the files to folds randomly.
2. Stratified random: Assign the files to folds randomly but make sure that there is the same number of positive and negative reviews in each fold.

Step 2: Cross-validation

You can now perform cross-validation using each set of folds. You should evaluate the accuracy of the Naive Bayes classifier using 9 folds for training and the 10^{th} one for testing. Repeat this 10 times so that each fold is tested only once. The cross-validation score is the average accuracy across all the runs. How do the two methods of splitting the data compare?

Apart from accuracy it is useful to calculate the variance across the cross-validation runs:

$$var = \frac{1}{n}\sum_{i}^{n}(x_i - \mu)^2$$

where x_i is the score of the i^{th} run, there are n runs and μ is the average of the scores.

NB: to get an unbiased estimate of the true / population variance when calculating sample variance, we would normally divide by $n-1$; for now, however, we will just use n.

Step 3: Evaluation on held-out data

Remember when in Task 2 you used a training–development dataset? Now you can finally use the test set which we held out! Train your Naive Bayes classifier using the training set from before and evaluate it using the held-out test set.

Step 4: The Wayne Rooney effect.

Apart from the portion of the dataset we held out before, now you can also download a new set of reviews from 2016. The original reviews were collected before 2004. How does your system (trained on your original training set) perform on this new review set?

Step 5 (not part of tester)

Run your sentiment lexicon system on the held-out data (as Step 3) and also on the 2016 data. Is there a significant difference between this system and the Naive Bayes system?

Starred tick

The term “Wayne Rooney effect” is not standard but it’s been used by people in the UK sentiment business to refer to the situation when a word or phrase that was a good cue for sentiment in the original data turned out to indicate the opposite sentiment in later data. Are there any words which are showing such behaviour in the 2016 data? What other reasons are possible for changes in performance? To what extent do these explain your results?

Note that we are expecting you to look at the 2016 data in order to do this **error analysis**. However, in a real experiment, after looking at test data, you have to obtain additional data to evaluate any further experiments.