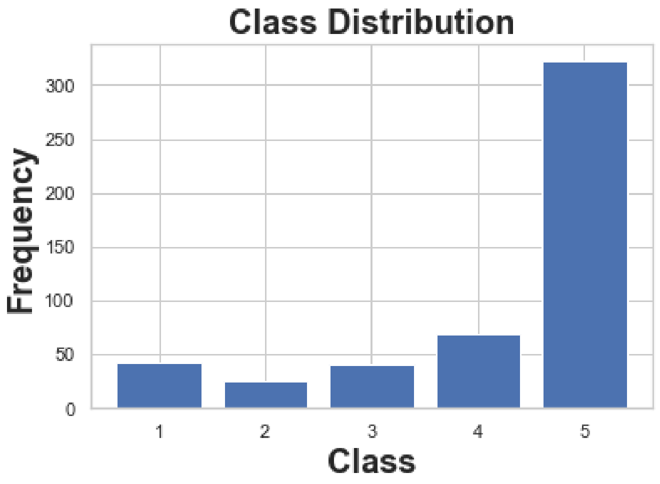
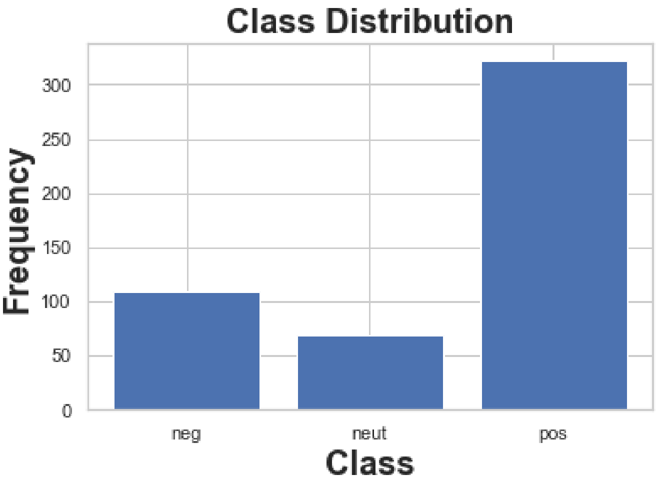
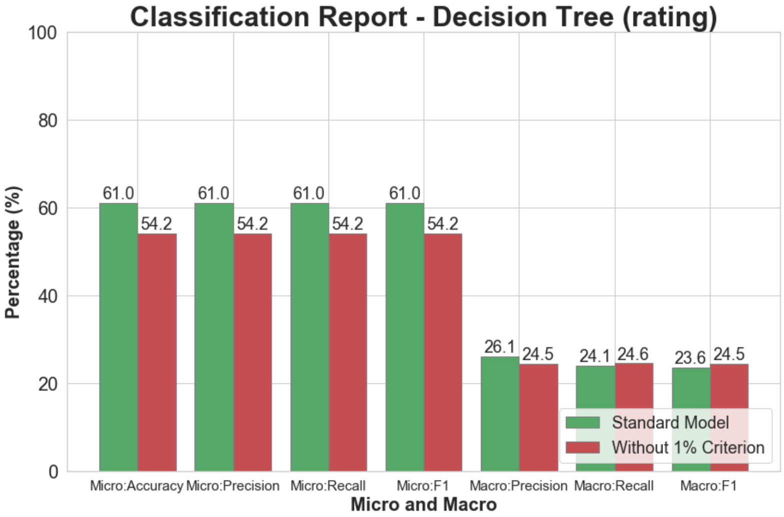
Dataset Class Distribution





Question 1

1i) Decision Tree – Ratings (standard vs without-1% stopping criterion)



Observations (Similarities & differences):

1 - The micro results for the standard model were all much higher.

2 - The macro results are much closer between the two models, with even slightly higher macro recall and F1 results.

3 - More deeply in macro, we can see that the recall and F1 is very slightly lower for the standard model but slightly higher in precision.

Explanation:

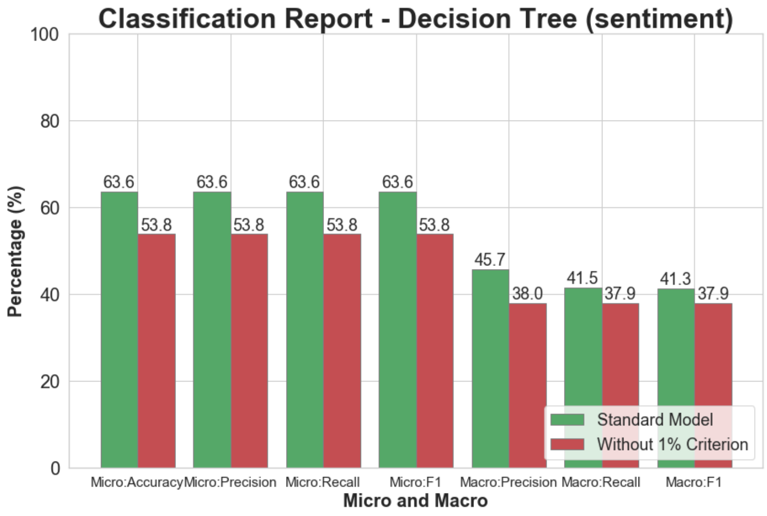
1 - The standard model generally performed better because of the 1% criterion. Having pruned the branches of the decision tree, it has stopped the model from overfitting, resulting in much higher micro results than the “without 1% criterion” as evident.

2 - I believe the fact that the macro results being very similar does not have so much to do with the model itself, but more to do with the unevenness class distribution for the rating system (we can see the discrepancy in the class distribution graph above). Having such an imbalanced class distribution would affect the reliability of macro-results since it considers all classes to be of ‘equal weight’.

3 - It makes sense for the standard model to have higher macro precision since the 1% criterion stops the model from overfitting, more likely resulting in more true positives. The fact that recall was higher for the modified model meant that there

were higher numbers of false negatives, which could possibly be because that the branches that were pruned by the 1% affected those which incorrectly predicted those false negative results.

1ii) Decision Tree - Sentiment (standard vs without-1% stopping criterion)



Observations (Similarities & differences):

1 – The micro results were all much higher in the standard model than the results of model without 1% criterion.

2 – The macro results were all higher in the standard model than that of the 1% criterion model.

3 - The difference between the models in terms of macro results aren't as large as that when comparing with micro results.

Explanation:

1 - The standard model generally performed better in the 3-class dataset because of the 1% criterion. This is expected having pruned the branches of the decision tree, since it would have stopped the model from overfitting, resulting in much higher micro results than the "without 1% criterion" as evident.

2 – This a somewhat expected result, as it was expected for the model with 1% criterion to stop overfitting, resulting better results for all classes, which further resulted in better macro results.

3 – The fact that the macro gap in results was not as large as the micro gap was likely due to the 1% criterion having more affect in certain classes than in others. Since macro assumes all classes to have equal weighting, which we know is not the case as we can see the imbalanced class distribution (as shown at the start of the report).

1iii) DT - Explain any differences in the results between scenarios 1 (Ratings) and 2 (Sentiment).

Observations (Similarities & differences):

1 – The overall micro results showed slight improvement in favour of the standard model and sentiment dataset.

2 – The overall macro results were found to be significantly much better for both models with the sentiment dataset.

Explanation:

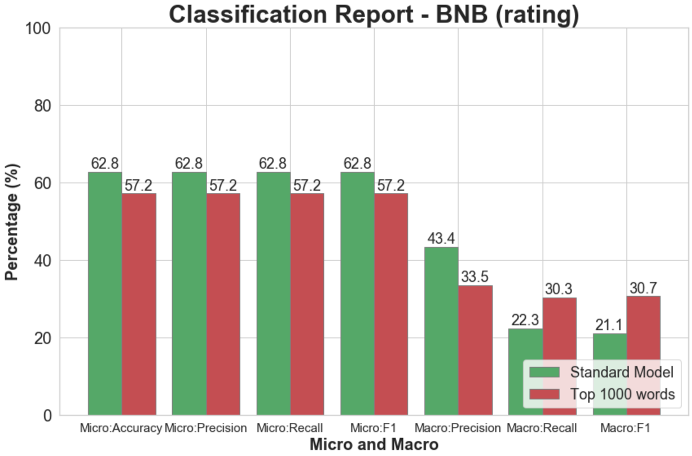
1 – It is expected that the sentiment dataset would perform better since we have reduced the number of classes from 5 to 3, which not only helped balance the classes a little bit (as shown in the class distribution graph), but also helped reduce the probability of chance that that model could incorrectly predict a class. Hence, it is expected that the sentiment dataset would outperform the ratings dataset. As for the 1% criterion filter, it has shown to have negative effect on the decision tree results. This may be because it could have removed those branches, which would have provided value/meaning.

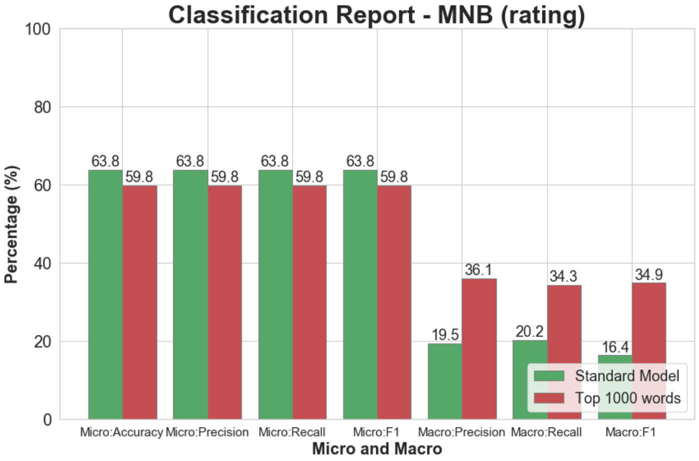
2 – Having 3 chosen classes as opposed to 5 classes would improve the balance in class distribution. Hence, the macro results unsurprisingly improved as compared to the rating dataset.

Question 2

2i) BNB, MNB – Ratings (Standard vs Top 1000 words)

BNB, MNB - - Ratings





Observations (Similarities & differences):

1 – The micro results of the standard model are higher than that of the top 1000 word model for BNB and MNB.

2 – Overall the macro results are much lower than that of the micro results for BNB and MNB.

3 – The BNB macro results are quite mixed. Much higher in precision for the standard model but much higher in recall and F1 for the top 1000 word model. Whilst, the MNB macro results shows that the top 1000 words model is much better than that of the standard model.

Overall – The results suggest that different classes had different 'reactions' to the top 1000 word filter for MNB and BNB. We can see this from the much higher macro results in top 1000 words but higher micro results in standard model. Here, I would be more inclined to lean with the micro results, since our classes are quite imbalanced.

Explanation:

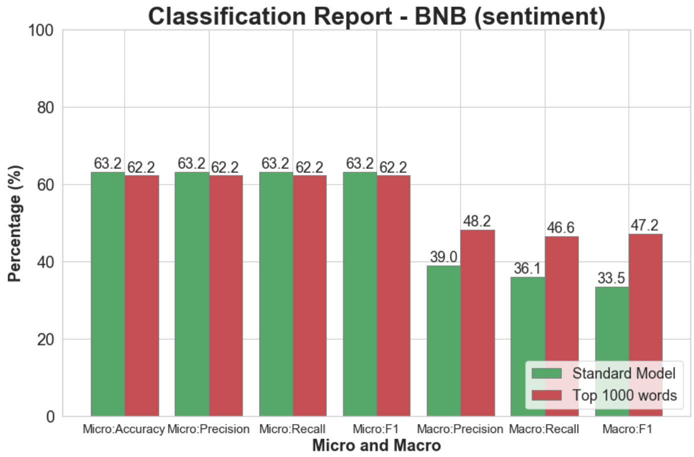
1 – Even words outside the top 1000 words were of much value to our prediction. Intuitively, it makes sense that a larger vocabulary of words would perform better.

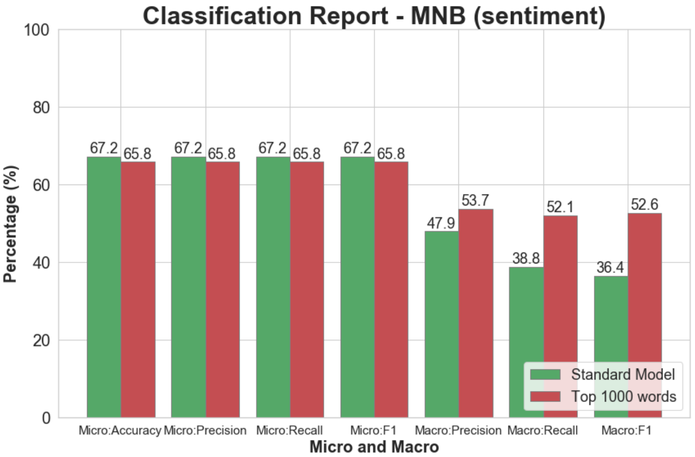
2 – As similarly mentioned before, it is most likely due to the imbalance in classes (as we can see in the class distribution graph).

3 – The fact that the macro precision for BNB did much better in standard model, where the top 1000 words model did much better for the MNB is very interesting. The nature of BNB and MNB suggests that BNB would probably do better with features that are binary distributed, whilst MNB would perform better with features that are multinomial distributed. I think in this case, MNB would be better since our dataset deals with the frequency of word, which is multinomial distributed. On top of this, the macro recall and F1 agree, which could be because the model performs better in classes with less frequency according to the class distribution graph. It is also potentially worth mentioning that the top-1000 word model affects certain words like '0z' or '1/2', so it is hard to say which class this may affect from first glance without examination.

2ii) BNB, MNB – Sentiment (Standard vs Top 1000 words)

BNB, MNB - - Sentiment





Observations (Similarities & differences):

1 – The micro results of the standard model are slightly higher than that of the top 1000 word model for BNB and MNB.

2 – The macro results show the top 1000 words to perform much better for both BNB and MNB.

Overall – The micro results suggest standard model is better whilst the macro results suggest that the top 1000 words model is better.

Explanation:

Overall – I think the reason why macro results say top 1000 model is better whilst the micro results say standard model is better, is because of the imbalance in classes between ‘negative’, ‘neutral’, and ‘positive’. The top-1000 word model affects certain words like ‘0z’ or ‘1/2’, so it is hard to say which class this may affect. However, it seems as though the words above the top 1000 words still had impact on the dataset. As you can see in the class distribution graph for sentiment, there is about 120 negative, 75 neutral, and 350 positives. Because of this, I would be more inclined to lean with the micro results; standard model performs better.

2iii) Explain any differences in the results between scenarios 1 (Ratings) and 2 (Sentiment)

Observations (Similarities & differences):

Overall – Micro/macro results are both generally much higher for the sentiment dataset.

MNB, BNB – The MNB performed better than the BNB model in all macro/micro results for both sentiment and rating dataset.

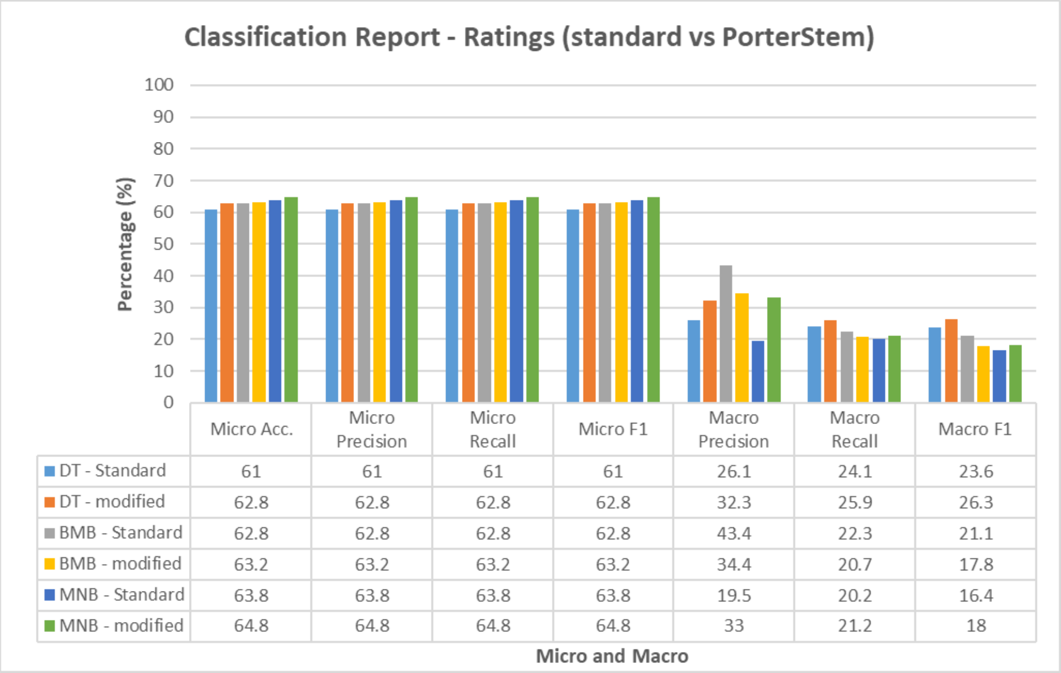
Explanation:

Overall - This would align with our predictions, since merging classes, with slightly less imbalance would generally result in better predictions. This is why we see slightly better results in sentiment than in ratings.

MNB, BNB – MNB uses a multinomial distribution for each of the features, whilst BNB uses binary. In this context of rating classification from text, MNB should generally out-perform BNB.

Question 3

3i) Ratings – Standard vs Porter-Stemming



	Micro Acc.	Micro Precision	Micro Recall	Micro F1	Macro Precision	Macro Recall	Macro F1
🟦 DT - Standard	61	61	61	61	26.1	24.1	23.6
🟧 DT - modified	62.8	62.8	62.8	62.8	32.3	25.9	26.3
◼ BMB - Standard	62.8	62.8	62.8	62.8	43.4	22.3	21.1
🟨 BMB - modified	63.2	63.2	63.2	63.2	34.4	20.7	17.8
🟦 MNB - Standard	63.8	63.8	63.8	63.8	19.5	20.2	16.4
🟩 MNB - modified	64.8	64.8	64.8	64.8	33	21.2	18

Observations (Similarities & differences):

Overall – The Porter Stemmer had the following results on the three models:

DT: Improved the micro, improved the macro

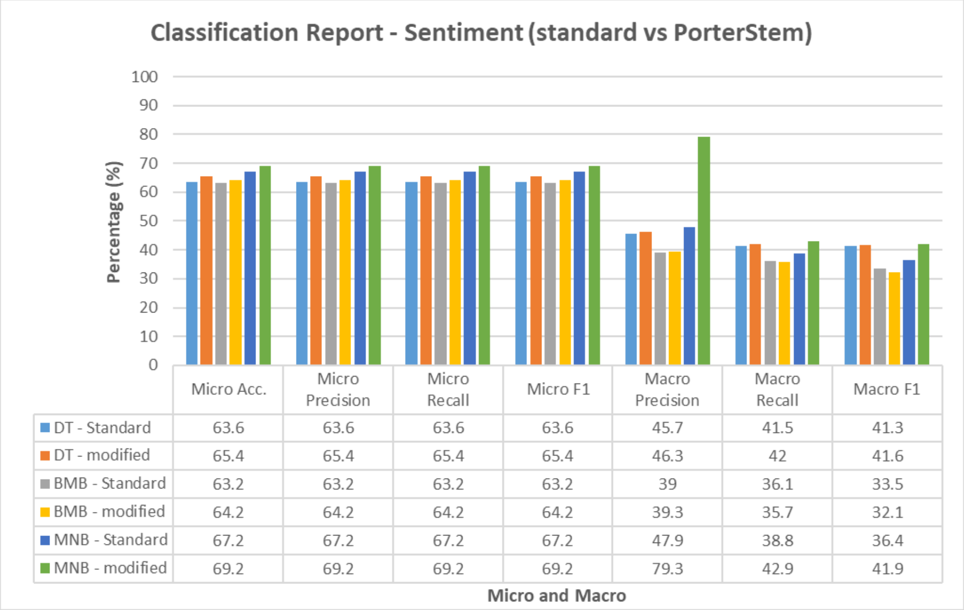
BMB: Improved the micro, worsened the macro

MNB: Improved micro, improved macro

Explanation:

Overall – The Porter-stemming filter would reduce commoner morphological and inflexional endings from words. Judging from the word bank in count vectoriser, a lot of the (would have been) less useful words have been merged together to become something more useful. For example, we can see in the words where “wanting” became “want”. Because of this, all of the models, DT, MNB and BMB have performed better. In short, there are less less-valuable terms and more more-valuable terms. This is consistent for all classes as we can see that it has performed better in both the macro and micro results.

3ii) Sentiment - Standard vs Porter-Stemming



	Micro Acc.	Micro Precision	Micro Recall	Micro F1	Macro Precision	Macro Recall	Macro F1
🟦 DT - Standard	63.6	63.6	63.6	63.6	45.7	41.5	41.3
🟧 DT - modified	65.4	65.4	65.4	65.4	46.3	42	41.6
▨ BMB - Standard	63.2	63.2	63.2	63.2	39	36.1	33.5
🟨 BMB - modified	64.2	64.2	64.2	64.2	39.3	35.7	32.1
🟦 MNB - Standard	67.2	67.2	67.2	67.2	47.9	38.8	36.4
🟩 MNB - modified	69.2	69.2	69.2	69.2	79.3	42.9	41.9

Observations (Similarities & differences):

Overall – The Porter Stemmer had the following results on the three models:

DT: Improved the micro, improved the macro

BMB: Improved the micro, about the same macro

MNB: Improved micro, improved macro

Explanation:

Overall – Very similar if not the exact same to the case of rating dataset, we see something similar for the sentiment dataset. Judging from the word bank in count vectoriser, a lot of the (would have been) less useful words have been merged together to become something more useful. For example, we can see in the words where “wanting” became “want”. Because of this, all of the models, DT, MNB and BMB have performed better. In short, there are less less-valuable terms and more more-valuable terms. This is consistent for all classes as we can see that it has performed better in both the macro and micro results.

3iii) Explain any differences in the results between scenarios 1 (Ratings) and 2 (Sentiment)

Observations (Similarities & differences):

Overall - the sentiment results were much higher than the ratings in terms of micro and macro for all 3 models. The sentiment model was clearly superior for this filter.

DT, MNB, BNB – The MNB performed better than the BNB and DT model in all macro/micro results for both sentiment and rating dataset. However, the BNB performed better than the DT model. The MNB also had the most improvement with the Porter stem filter.

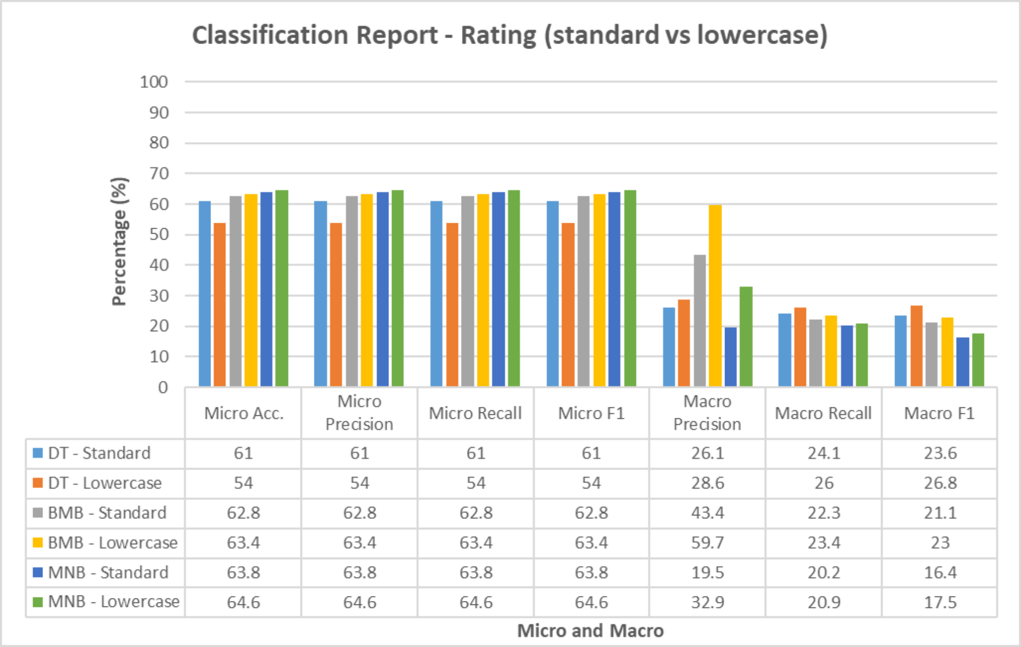
Explanation:

Overall – The sentiment dataset has fewer classes than the ratings dataset, since we have merged some classes from the ratings dataset together. This would generally produce better results.

DT, MNB, BNB – MNB uses a multinomial distribution for each of the features, whilst BNB uses discrete value. In this context of rating classification from text, MNB should generally out-perform BNB. Additionally, the BNB performed better than the DT model. This is likely due to the fact that decision trees generally overfit in text classification cases and that it would be quite hard for the model to predict text based classification with a terms that are not very clear in meaning (we can see this in the word bank where it has a lot of words that do not provide much meaning – especially random numbers).

Question 4

4i) Ratings – Standard vs Lower case



	Micro Acc.	Micro Precision	Micro Recall	Micro F1	Macro Precision	Macro Recall	Macro F1
DT - Standard	61	61	61	61	26.1	24.1	23.6
DT - Lowercase	54	54	54	54	28.6	26	26.8
BMB - Standard	62.8	62.8	62.8	62.8	43.4	22.3	21.1
BMB - Lowercase	63.4	63.4	63.4	63.4	59.7	23.4	23
MNB - Standard	63.8	63.8	63.8	63.8	19.5	20.2	16.4
MNB - Lowercase	64.6	64.6	64.6	64.6	32.9	20.9	17.5

Observations (Similarities & differences):

The lower-case filter had the following results on the three models:

DT: Worsened the micro, improved the macro

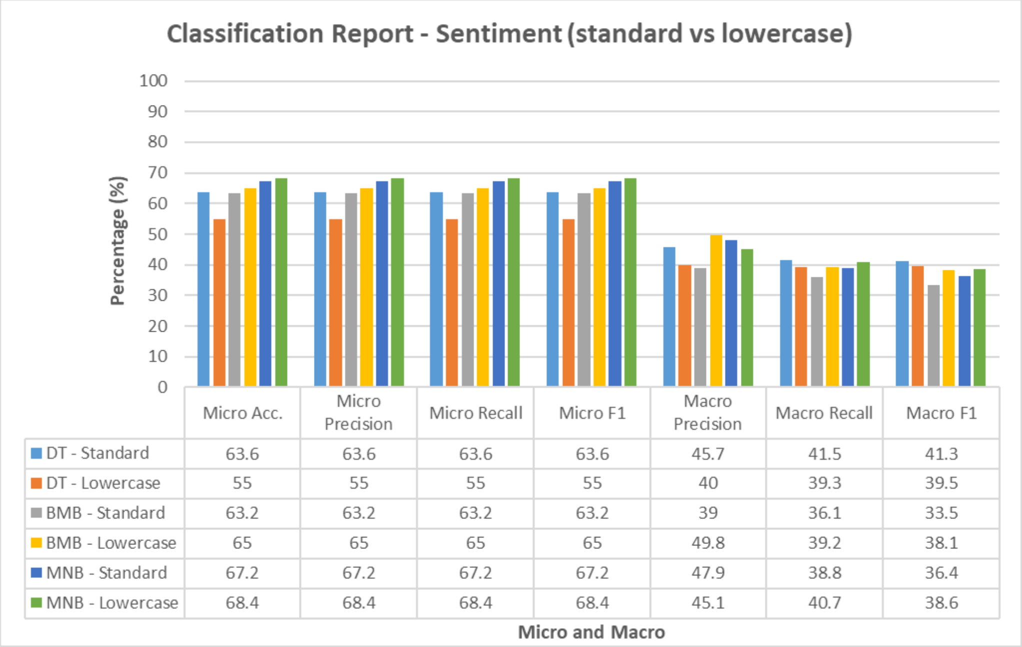
BMB: Improved the micro, improved the macro

MNB: Improved micro, improved macro

Explanation:

- Lower-case filter would generally help situations where there are capitalised words/terms that would have been potentially categorised as two terms. For example if there were two terms ‘excellent’ and ‘Excellent, without the lower-case filter this would be classified as two terms, but with it it becomes just ‘excellent’. It is hard to explain why exactly the DT unimproved in terms of performance with the lower-case filter in terms of micro results, however it has overall helped the classification according to the macro results. Perhaps, it may have had something to do with overfitting. The lowercasing helped the MNB and BNB, due to combining the terms (which would’ve been originally split into two terms in the model if it weren’t for lower-casing). This may be due to the fact that it is by nature. The MNB considers each word ‘as a feature’ and the probability of words occurring would be more affected by the number of useless/useful terms. Considering this, it is expected that the model would improve ever so slightly depending on the dataset. The BNB model is similar but coined by binary-values. Hence, we should also see similar improvements to the MNB.

4ii) Sentiment – Standard vs Lower case



	Micro Acc.	Micro Precision	Micro Recall	Micro F1	Macro Precision	Macro Recall	Macro F1
🟦 DT - Standard	63.6	63.6	63.6	63.6	45.7	41.5	41.3
🟧 DT - Lowercase	55	55	55	55	40	39.3	39.5
⬛ BMB - Standard	63.2	63.2	63.2	63.2	39	36.1	33.5
🟨 BMB - Lowercase	65	65	65	65	49.8	39.2	38.1
🟦 MNB - Standard	67.2	67.2	67.2	67.2	47.9	38.8	36.4
🟩 MNB - Lowercase	68.4	68.4	68.4	68.4	45.1	40.7	38.6

Observations (Similarities & differences):

The lower-case filter had the following results on the three models:

DT: Worsened the micro, worsened the macro

BMB: Improved the micro, improved the macro

MNB: Improved micro, improved macro recall/f1, worsened macro precision

Overall – It had a mixed performance depending on the model used.

Explanation:

- This is very similar to the question above, but here since there are 3 classes instead of 5, the impact of lower-case terms would have a greater effect. However in general, lower-case filter would generally help situations where there are capitalised words/terms that would have been potentially categorised as two terms. For example if there were two terms ‘excellent’ and ‘Excellent, without the lower-case filter this would be classified as two terms, but with it it becomes just ‘excellent’. It is hard to explain why exactly the DT unimproved in terms of performance with the lower-case filter in terms of micro results, however it has overall helped the classification according to the macro results. Perhaps, it may have had something to do with overfitting. The lowercasing helped the MNB and BNB, due to combining the terms (which would’ve been originally split into two terms in the model if it weren’t for lower-casing). This may be due to the fact that it is by nature. The MNB considers each word ‘as a feature’ and the probability of words occurring would be more affected by the number of useless/useful terms. Considering this, it is expected that the model would improve ever so slightly depending on the dataset. The BNB model is similar but coined by binary-values. Hence, we should also see similar improvements to the MNB.

4iii) Explain any differences in the results between scenarios 1 and 2.

Observations (Similarities & differences):

- Overall, sentiment dataset performed much better than the ratings dataset.

- Macro results were significantly higher with the sentiment dataset

Explanation:

- The sentiment dataset is again, outperforming the ratings dataset. Similar to the explanation in question 3, selectively merging classes generally would produce a more balanced dataset (we can see this from the class distribution graph). This generally translates to better predictions as we can see for the lower casing filter. This is why we see slightly better results in sentiment than in ratings.

DT, MNB, BNB – MNB uses a multinomial distribution for each of the features, whilst BNB uses discrete value. In this context of rating classification from text, MNB generally out-perform BNB. The results align with this. Also, the BNB performed better than the DT model. This is expected (with explanation similar to question 3), the DT can quite easily overfit in general, but also in text classification questions where the word bank isn't so clear in conveying the message of the sentence.

Question 5 – My Classifier

For our model, we will be basing our model on the **MNB model**. It is clearly the best model for this dataset out of DT and BNB as shown in questions 1-4. On top of this, we will be using the **sentiment** dataset, as it has shown to also give much better results than the rating dataset.

For this question, the order of these results are (micro: accuracy, precision, recall, F1, macro: precision, recall, F1)

For reference, the standard MNB model is this:

- Standard MNB model: [0.672, 0.672, 0.672, 0.672, 0.479, 0.388, 0.364]

From the first four questions, it has been determined that following parameters helped improve the model:

Sentiment, Without 1% criterion, Not top 1000 most frequent (all vocab), PorterStem, and Lowercase.

To start, I decided to apply these filters/techniques to the standard MNB model and this was the result:

- Standard my_model : [0.688, 0.688, 0.688, 0.688, 0.464, 0.419, 0.404]

From here, I looked at the word bank in count vectoriser, and saw that many of the words were not "very clean", with a lot of accidentally combined words, and many numbers, which I felt, weren't providing much value to the accuracy of the prediction. Therefore, I decided to change the parameters of the count vectoriser.

Then, I first started playing with numbers and special characters –

- Removing numbers: mnb [0.686, 0.686, 0.686, 0.686, 0.457, 0.42, 0.405]
- Removing special characters and numbers: mnb [0.692, 0.692, 0.692, 0.692, 0.466, 0.429, 0.417]
- Removing all special characters: mnb [0.686, 0.686, 0.686, 0.686, 0.465, 0.422, 0.41]

I found that removing the special characters and numbers improved the model than either of just the two.

Then, I tried changing the word parameters in the pre-processing:

Tried removing each individual symbol ($%/-), tried removing all numbers in pre-processing, tried removing all capital letters, tried removing lower-case letters: None of these could improve upon the results from the previous step.

Then I tried changing the number of letters required for a word:

Filtering out at least 4 letters: mnb_modified [0.684, 0.684, 0.684, 0.684, 0.526, 0.432, 0.429]

Filter out 3 letters minimum resulted in [0.698, 0.698, 0.698, 0.698, 0.596, 0.448, 0.45]

At least 5 letters: [0.684, 0.684, 0.684, 0.684, 0.505, 0.428, 0.424]

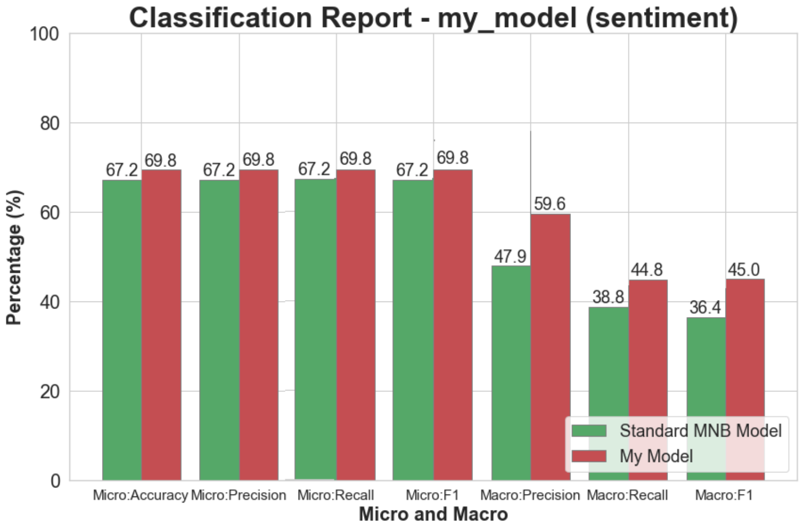
At least 1 letter: [0.694, 0.694, 0.694, 0.694, 0.577, 0.445, 0.447]

Having done all this, the best model that I could come up with was applying the following:

1. Sentiment dataset filter

2. Q1-4 parameters
3. Applying a minimum 3 letter words whilst pre-processing raw data
4. Removal of special characters and numbers as part of the CountVectoriser token parameter.

This is the visual representation of the before and after of my results.



Observations (Similarities & differences):

Overall – My model significantly outperforms the standard MNB model in terms of both micro and macro results

Results - 3 letter words, removing special characters and numbers helped the model, and all parameters from Q1-4.

Explanation:

Overall – It makes sense that it would improve the standard model since it was my goal when I changed each parameter. The fact that both micro and macro were outperforming implied that my model was consistently predicting better than the standard model for all classes.

Results - The minimum 3-letter words improved the results. This made sense because when I was looking through the raw data for words, there were many useless 2-letter words. Implementing this filtered those words out, which improved the prediction. Additionally, after looking at the word bank, I saw that there were many words with what seemed like not very useful terms like “0z” or “1/2”. After I removed these, I saw that it helped the result prediction as well. As mentioned before, having parameters from Q1-4 would help the model since we saw what each individual filter did.