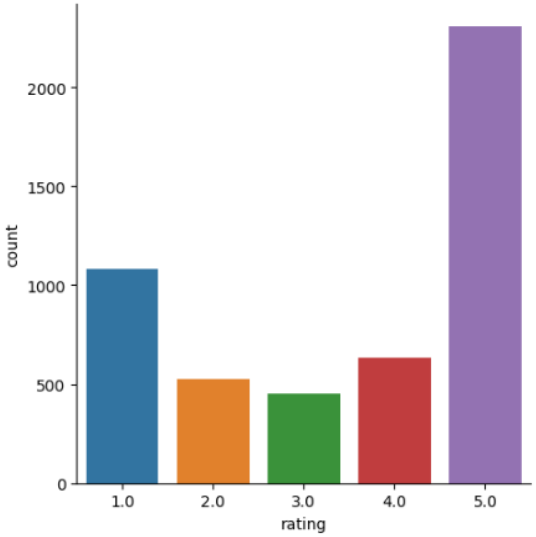
Preprocessing

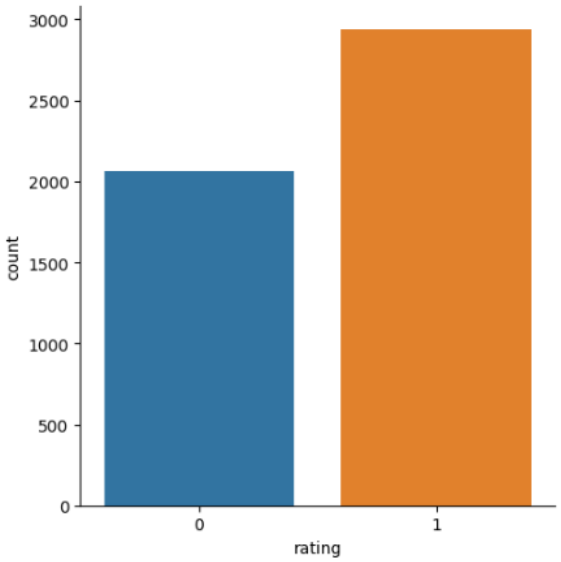
I used Kaggle's coffee maker dataset (https://www.kaggle.com/datasets/niramay/-coffeemakerclassification) for this assignment. I saved the csv file onto Amazon's S3 cloud storage under my account, so I can easily use NumPy's read_csv function and save the contents to a dataframe. I only used the 'rating' and 'review' columns, since I planned on implementing sentiment analysis. For pre-processing, I converted the words under 'review' to lowercase as well as removed punctuation.

I used Seaborn to visualize the count of each rating:



According to the graph, 5.0 ratings take up the majority. Interestingly enough, 1.0 rating is the next highest frequency, followed by 4.0, 2.0, then finally 3.0. Since I'm doing sentiment analysis I changed each rating to be either 0 or 1. If the initial rating is 3.0 or lower, the new rating is 0. Otherwise, it's changed to 1.

The distribution now looks like this:




```
df['rating'].value_counts()

5.0    2305
1.0    1084
4.0     631
2.0     526
3.0     454
Name: rating, dtype: int64
```

According to the graph and value counts function, there's a 1,221 difference in reviews between positive ratings and negative ratings.

Next, I used WordCloud to create a visualization of the most common words for positive reviews and negative reviews. Some of the common words in the positive WordCloud include "great", "bargain", and "wow." For the negatives, common words include "flimsy", "trash", and "plastic."

In this assignment, I wanted to compare if lemmatizing each word in the reviews would make a difference in the metric scores. I created a function that would implement lemmatization.

Naive Bayes

I compared Multinomial Naive Bayes (MNB), Complement Naive Bayes (CNB), and Bernoulli Naive Bayes (BNB) with lemmatized and non-lemmatized reviews.

Non-Lemmatized

The metric scores include accuracy, precision, recall, F1 score, and ROC AUC. MNB & CNB scores were on par with each other, with every score above 80%. However, for BNB, the metric scores were noticeably lower.

MNB

```
accuracy score:  86.4 %
precision score:  83.89458272327965 %
recall score:  95.65943238731218 %
f1 score:  89.39157566302653 %
None
ROC AUC:  94.4749978143123 %
None
```

CNB

```
accuracy score:  87.9 %
precision score:  87.93650793650794 %
recall score:  92.48747913188647 %
f1 score:  90.15459723352319 %
None
ROC AUC:  94.4749978143123 %
None
```

BNB

```
accuracy score:  77.7 %
precision score:  77.01149425287356 %
recall score:  89.48247078464107 %
f1 score:  82.77992277992277 %
None
ROC AUC:  89.12651593054093 %
None
```

Neural Networks

I used MLPClassifer from sklearn to create a neural network model. I had to change the alpha value a few times, including 0.01, 0.05, 0.09, and 0.2. Running the model took a few minutes to be completed. When alpha = 0.09, it had the highest metric scores.

0.01

```
accuracy score:  88.3 %
precision score:  90.16666666666666 %
recall score:  90.31719532554257 %
f1 score:  90.24186822351959 %
None
ROC AUC:  94.63070204288944 %
None
```

0.05

```
accuracy score:  88.7 %
precision score:  90.0990099009901 %
recall score:  91.15191986644408 %
f1 score:  90.62240663900415 %
None
ROC AUC:  94.56742117993831 %
None
```

0.09

```
accuracy score:  89.8 %
precision score:  91.48580968280467 %
recall score:  91.48580968280467 %
f1 score:  91.48580968280467 %
None
ROC AUC:  94.84594024121665 %
None
```

0.2

```
accuracy score:  89.7 %
precision score:  91.19601328903654 %
recall score:  91.65275459098497 %
f1 score:  91.42381348875938 %
None
ROC AUC:  94.92920453457342 %
None
```

Overall, these scores are similar to the best scores for Naive Bayes models (excluding BNB).

Logistic Regression

For logistic regression, the parameters I've chosen include C=2.5, n_jobs=4, solver='lbfgs', and random_state=17. The model predictions are also on par with the NN and NB models.

```
accuracy score:  89.3 %
precision score:  89.10569105691057 %
recall score:  93.19727891156462 %
f1 score:  91.10556940980882 %
None
ROC AUC:  95.57245228188361 %
None
```

Conclusion

In the end, I ran each model again but with lemmatized words. However, there wasn't a noticeable difference between the predictions for the lemmatized and non-lemmatized words. Sometimes, the metric scores were slightly lower.

Overall, the metric scores for each Naive Bayes model (except for Bernoulli), the Neural Network model, and the Logistic Regression model were high for every category. The NN model had the highest scores, followed by Logistic Regression, then Naive Bayes.