CS464
Introduction to Machine Learning
Homework 1

Kaan Tek
21901946

Question 1: The Online Shopping Case

Question 1.1:

We need to calculate the probability that the product is P and got positive feedback or is M and got positive feedback or is U and got positive feedback:

$$\%45 * \%95 + \%60 * \%30 + \%10 * \%25 = \%63.25 = P(F_P)$$

Question 1.2:

Using Bayes Theorem, we can say:

$$P(P|F_P) = \frac{P(F_P|P) * P(P)}{P(F_P)}$$

We know the terms in the numerator as they are given in the question definition, and we found the term in the denominator in Question 1.1 which is $P(F_P)$. Plugging in these values:

$$\frac{\%95 * \%45}{\%63.25} = \%67.5889$$

Question 1.3:

Using Bayes Theorem, we can say:

$$P(P|F_N) = \frac{P(F_N|P) * P(P)}{P(F_N)}$$

The first term in the numerator is simply $1 - P(F_P|P)$, and the second term is known from the question definition. And the term in the denominator $P(F_N)$ is equal to $1 - P(F_P) = \%36.75$. Plugging in these values:

$$\frac{\%5 * \%45}{\%36.75} = \%6.1224$$

Question 2: Spam Email Detection

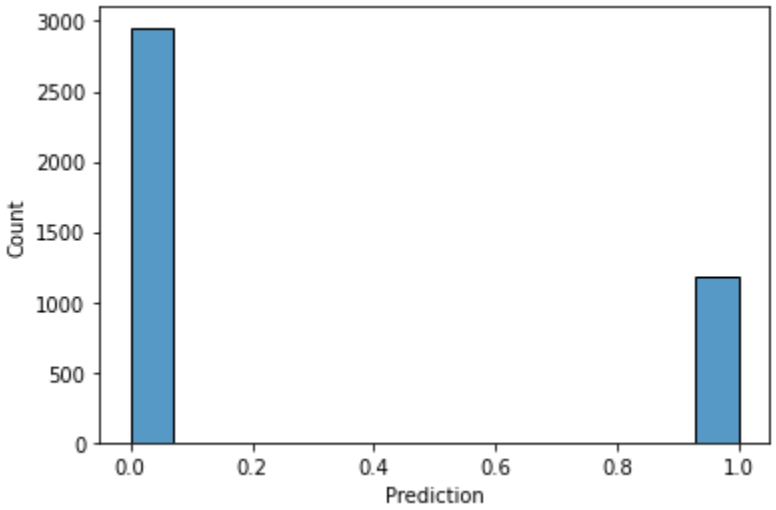
Question 2.1:

Question 2.1.1:

As we draw the histplot of the distribution of classes in *y_train.csv*, we see that out of 4137 total data instances, 2954 of these are labeled as non-spam, and 1183 are labeled as spam. Therefore, the percentage of the spam e-mails is

$$1183/4137 * 100 \approx \%28.5956$$

```
0    2954
1    1183
Name: Prediction, dtype: int64
```



Question 2.1.2:

The result obtained from Question 2.1 indicated that our training set is skewed towards non-spam mails, and this affects our model. Since our model is trained with fewer data that are labeled as “spam”, it may not be efficient to identify a new “spam” data. Because our model is trained to favor the “non-spam”

class, and may not have completely learnt some patterns and features that are specific to “spam” emails.

Question 2.1.3:

As mentioned above, the skew towards non-spam emails can affect the accuracy. Our model will be more successful with a test dataset that includes less spam emails and more non-spam emails. But as the number of spam emails increase in a test dataset, our accuracy will keep decreasing, because it will make more false predictions as it is fed with more spam emails, which it is not good at estimating.

Question 2.2: Multinomial Naive Bayes

Our *calc_accuracy* method yields the following results:

- Accuracy: 0.9584541062801932 = % 95.84541062801932
- Number of wrong predictions: 43
- Number of True Positives: 289
- Number of False Positives: 15
- Number of True Negatives: 703
- Number of False Negatives: 28

Therefore, our confusion matrix can be derived as:

TP 289	FP 15
FN 28	TN 703

Question 2.3: Multinomial Naive Bayes with Smoothing

Our *calc_accuracy* method yields the following results:

- Accuracy: 0.9478260869565217 = % 94.78260869565217
- Number of wrong predictions: 54
- Number of True Positives: 300
- Number of False Positives: 37
- Number of True Negatives: 681
- Number of False Negatives: 17

Therefore, our confusion matrix can be derived as:

TP 300	FP 37
FN 17	TN 681

As can be seen from the above result, using smoothing slightly decreased our accuracy. However one important observation is our False Negative value decreased. False Negative indicates that our model predicted a spam email as a safe email. It may achieved this success by decreasing the effects of words that appear rarely by adding a prior value.This is an important metric, as it is the main purpose of this model: identifying spam emails. Therefore, using smoothing was a great choice.

Question 2.4: Bernoulli Naive Bayes

Our *calc_accuracy* method yields the following results:

- Accuracy: 0.9207729468599034 = % 92.07729468599034

- Number of wrong predictions: 82
- Number of True Positives: 247
- Number of False Positives: 12
- Number of True Negatives: 706
- Number of False Negatives: 70

Therefore, our confusion matrix can be derived as:

TP 247	FP 12
FN 70	TN 706

Question 2.5:

The very first observation could be, Multinomial models could be better choices than the Bernoulli model, as the Bernoulli model performs worse considering its lower accuracy. It also has a higher False Negative value, meaning that it is not as good as others in detecting a spam email. Between the normal Multinomial model and Smoothed Multinomial mode, even though the first one has a very slightly better accuracy, the Smoothed one could be a better choice for a real-world. It has a lower False Negative value, meaning that it is better at identifying a spam email correctly, which is the main purpose of the model. Its False Positive value is higher, meaning it can identify some safe emails as spam, but this should not be a huge issue as people tend to check their spam emails regularly. And I would say it is better to have safe emails identified as spam than spam emails identified as safe. Therefore, I think the Smoothed Multinomial model is better.

It can be said that accuracy is not the best metric for this experiment, as we chose the model with second-best accuracy as the best one. Accuracy does not take into account if more important classes are predicted better than less

important ones. Some other metrics such as F-Score could be a better option for this experiment.

Use of external libraries:

I used **pandas** primarily to read the csv files into a pandas DataFrame. This was useful as pandas' DataFrame provides some functions such as easily adding a column, copying a DataFrame, etc that was useful in my implementation. However, I benefit the most from the **NumPy** library. Most of the time I did my calculations by converting the DataFrame to a NumPy array. NumPy saved me from using lots of nested for loops and it significantly reduced the execution time, thanks to some functions it provides such as np.where(), np.count_nonzero(), np.transpose(), np.log(), np.sum(). I also used the **seaborn** library once, in order to draw a plot so that I could include it in my report.