1) a) In this experiment, you should use 20% of the training set and 20% of the test set; i.e., call the dataset-loading functions by passing 0.2 as their parameters. First, perform the classification of the instances in the test set by comparing posterior probabilities, $Pr(y_i \mid Doc)$, according to Eq. (1), for both classes. Then, report (i) the accuracy of your model; (ii) its precision; (iii) its recall; and (iv) the confusion matrix resulting from this experiment.

```
Accuracy  is:  0.374631
Precision is:  0.367178
Recall    is:  0.348956
```

Naive Bayes without log-probabilities

	Predicted Positive	Predicted Negative
Actual Positive	886	1653
Actual Negative	1527	1019

b) Now repeat the same experiment above but classify the instances in the test set by comparing log-probabilities, $\log(Pr(y_i \mid Doc))$, according to Eq. (5), for both classes. Report the same quantities as before. Discuss whether classifying instances by computing log-probabilities, instead of probabilities, affects the model's performance. Assuming that this transformation does have an impact on performance, does it affect more strongly the model's accuracy, precision, or recall? Why do you think that is the case?

```
Accuracy  is:  0.565500
Precision is:  0.592678
Recall    is:  0.504338
```

Naive Bayes with log-probabilities

	Predicted Positive	Predicted Negative
Actual Positive	1279	1257
Actual Negative	879	1501

The computation of summing log-probabilities instead of the posterior probabilities increases the accuracy, precision, and recall scores of the Naive Bayes algorithm.
We observe that: Accuracy scores increased from ~0.375 to ~0.565; Precision scores increased from ~0.367 to 0.593; Recall scores increased from ~0.349 to ~0.504

I believe that the increase in performance is because the multiplication of posterior probabilities can result in underflow due to limited numerical precision. Taking the logarithm of probabilities and adding them transforms multiplication into addition, solving the issue by making it more numerically stable. In our case, we noticed small values of probabilities for a large number of features resulting in inaccurate results with many instances being 0. Our significant increase in the precision value signifies that the model performed better while classifying positive instances suggesting that the logarithm transformation reduced the impact of outliers and noise, resulting in a more robust estimation of probabilities.

2) In this experiment, you should use 20% of the training set and 20% of the test set; i.e., call the dataset-loading functions by passing 0.2 as their parameters. You should first report the confusion matrix, precision, recall, and accuracy of your classifier (when evaluated on the test set) when using α = 1. Now, vary the value of α from 0.0001 to 1000, by multiplying α with 10 each time. That is, try values of α equal to 0.0001, 0.001, 0.01, 0.1, 1.0, 100, and 1000. For each value, record the accuracy of the resulting model when evaluated on the test set. Then, create a plot of the model's accuracy on the test set (shown on the y-axis) as a function of the value of α (shown on the x-axis). The x-axis should represent α values and use a log scale. Analyze this graph and discuss why do you think the accuracy suffers when α is too high or too low.

Confusion Matrix using α = 1:

```
Accuracy  is:  0.826034
Precision is:  0.861673
Recall    is:  0.769704
```

Naive Bayes with Lapace Smoothing (alpha = 1)

	Predicted Positive	Predicted Negative
Actual Positive	1875	561
Actual Negative	301	2218

We observe an increase in performance when applying Laplace smoothing as we were able to handle the case where a feature has zero frequency in a particular class. This occurs whenever a word does not appear in any reviews for a particular sentiment class and the Naive Bayes classifier assigned a probability of zero to that feature for that class, making the posterior probability of that class also zero.

Confusion Matrix varying the value of α:

When α is 0.0001:

```
Accuracy  is:  0.728820
Precision is:  0.744681
Recall    is:  0.691254
```

When α is 0.001:

```
Accuracy  is:  0.752259
Precision is:  0.790200
```

```
Recall    is:  0.699251
```

When α is 0.01:

```
Accuracy  is:  0.780342
Precision is:  0.808409
Recall    is:  0.738029
```

When α is 0.1:

```
Accuracy  is:  0.803320
Precision is:  0.846386
Recall    is:  0.751624
```

When α is 1:

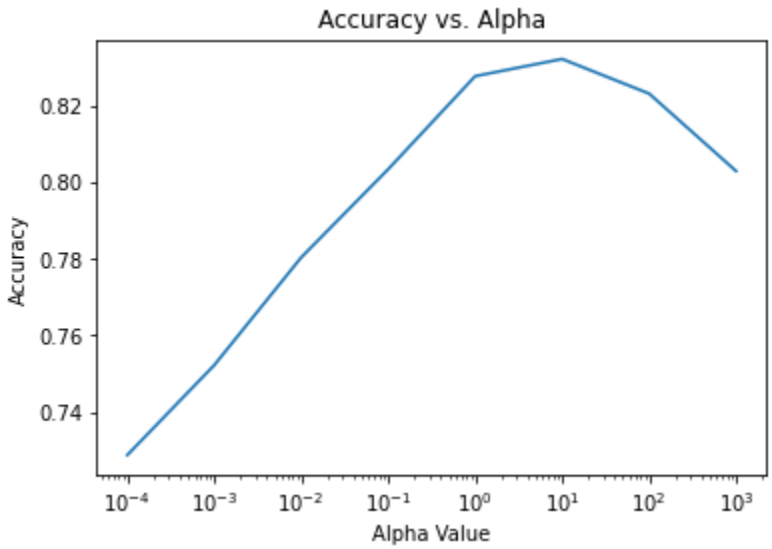
```
Accuracy  is:  0.827573
Precision is:  0.872536
Recall    is:  0.761809
```

When α is 100:

```
Accuracy  is:  0.832049
Precision is:  0.878285
Recall    is:  0.771567
```

When α is 1000:

```
Accuracy  is:  0.823034
Precision is:  0.888723
Recall    is:  0.731099
```



After analyzing the graph, it is observed that the performance of the Naive Bayes classifier is influenced by the choice of the value of α. This is the case because when the value of α is too small, the effect of Laplace smoothing is negligible, and the probabilities of the features are estimated based purely on their frequency in the training data. This can lead to overfitting, as the probabilities may not generalize well to new and unseen data. On the other hand, when α is too large, the normal probability's contribution is decreased, which can lead to underfitting.

The peak in accuracy around α=10 suggests that this value of α provides a good balance between the effect of Laplace smoothing and the normal probabilities. This value of α ensures that the probabilities of the features are smoothed, but not excessively so, allowing for better generalization to new and unseen data.

3) Now you will investigate the impact of the training set size on the performance of the model. The classification of new instances, here, should be done by comparing the posterior log-probabilities, $\log(\Pr(y_i \mid \text{Doc}))$, according to Eq. (5), for both classes. You should use the value of α that resulted in the highest accuracy according to your experiments in the previous question. In this question, you should use 100% of the training set and 100% of the test set; i.e., call the dataset-loading functions by passing 1.0 as their parameters. Then, report (i) the accuracy of your model; (ii) its precision; (iii) its recall; and (iv) the confusion matrix resulting from this experiment.

Confusion Matrix using α = 10

```
Accuracy  is:  0.842560
Precision is:  0.876208
Recall    is:  0.797840
```

Naive Bayes on Data Set with alpha = 10

	Predicted Positive	Predicted Negative
Actual Positive	9973	2527
Actual Negative	1409	11091

4) Now repeat the experiment above but use only 50% of the training instances; that is, load the training set by calling load training set(0.5, 0.5). The entire test set should be used. Report the same quantities as in the previous question. Discuss whether using such a smaller training set had any impact on the performance of your learned model. Analyze the confusion matrices (of this question and the previous one) and discuss whether one particular class was more affected by changing the size of the training set.

Confusion Matrix using α = 10:

```
Accuracy  is:  0.843560
Precision is:  0.863787
Recall    is:  0.815760
```

Naive Bayes on Data Set with alpha = 10 and 50% of the training set

	Predicted Positive	Predicted Negative
Actual Positive	10197	2303
Actual Negative	1608	10892

Comparing the results from Q3 and Q4, we observe that the performance of the learned model is not significantly affected by reducing the size of the training set from 100% to 50%. The accuracy, precision, and recall scores are all similar between the two experiments and the model still performs well.

I believe this is the case because the training dataset still contains enough information to create reasonable probabilities that accurately reflect the actual test cases, despite not having access to the full training dataset.

5) In this application (i.e., accurately classifying movie reviews), would you say that it is more important to have high accuracy, high precision, or high recall? Justify your opinion.

In the application of classifying movie reviews, all three measures have their own importance. However, the most important measure would depend on the specific use case and the end goal of the algorithm. For instance, if the aim is to provide recommendations to users based on their movie preferences, then recall may be the most important measure. This is because it is essential to correctly identify all positive reviews (true positives) to ensure that users receive recommendations for movies they are likely to enjoy. On the other hand, if the goal is to filter out spam reviews or prevent false recommendations, then precision may be the most critical measure.
Additionally, accuracy is generally an essential measure to evaluate the overall performance of the model. It reflects the proportion of correctly classified instances out of all instances. However, in some cases, accuracy alone may not provide a complete picture of the model's performance, especially when the dataset is imbalanced.
In conclusion, in the application of classifying movie reviews, the choice of the most important measure would depend on the specific use case and the end goal of the application. All three measures (accuracy, precision, and recall) should be considered together to obtain a comprehensive evaluation of the model's performance.

6) Finally, you will study how the performance of the learned model is affected by training it using an unbalanced dataset (i.e., a dataset with significantly more examples of one of the classes). The classification of new instances, here, should be done by comparing the posterior log-probabilities, $\log(\Pr(y_i \mid \text{Doc}))$, according to Eq. (5), for both classes. You should use the value of α that resulted in the highest accuracy according to your experiments in the previous questions. You will now conduct an experiment where you use only 10% of the available positive training instances and that uses 50% of the available negative training instances. That is, use load training set(0.1, 0.5). The entire test set should be used. Show the confusion matrix of your trained model, as well as its accuracy, precision, and recall. Compare this model's performance to the performance (according to these same metrics) of the model trained in question Q.4—that is, a model that was trained under a balanced dataset. Discuss how training under an unbalanced dataset affected each of these performance metrics.

Confusion Matrix using α = 10:

```
Accuracy    :  0.500200
Precision   :  1.000000
Recall      :  0.000400
```

Naive Bayes on Data Set with alpha = 10

	Predicted Positive	Predicted Negative
Actual Positive	5	12495
Actual Negative	0	12500

Firstly, even though α=10 performed the best in the above experiments, it performed poorly in this case because in the case of a highly unbalanced dataset, like the one used in this experiment, the prior probability of the positive class becomes significantly lower than that of the negative class. As a result, the positive class features may become less informative and might even be ignored by the classifier if α is set too high. This can be indicated by the recall value of ~0.0004 when α=10, signifying that the model is missing a large number of positive instances.

This is different then the model trained in question Q.4, which was trained under a balanced dataset and achieved much better results. Training under an unbalanced dataset significantly affected all the performance metrics, with a decrease in accuracy and recall and an increase in precision. This is because the lack of positive training data made the model biased towards negative predictions and affected its ability to identify true positive instances.