Homework 1

Naive Bayes, Decision Trees, MLE and MAP

CMU 10-701: Machine Learning (Spring 2021)

piazza.com/cmu/spring2021/10701
OUT: Monday, February 08 2021

DUE: February 17th 2021 11:59pm EST

START HERE: Instructions

- **Collaboration policy:** Collaboration on solving the homework is allowed, after you have thought about the problems on your own. It is also OK to get clarification (but not solutions) from books or online resources, again after you have thought about the problems on your own. There are two requirements: first, cite your collaborators fully and completely (e.g., "Jane explained to me what is asked in Question 2.1"). Second, write your solution *independently*: close the book and all of your notes, and send collaborators out of the room, so that the solution comes from you only. See the Academic Integrity Section on the course site for more information:
https://www.cs.cmu.edu/~aarti/Class/10701_Spring21/index.html
- **Extension Policy:** See the homework extension policy here:
https://www.cs.cmu.edu/~aarti/Class/10701_Spring21/index.html
- **Submitting your work:**
  - All portions of the assignments should be submitted to Gradescope (https://gradescope.com/).
  - **Programming:** We will autograde your Python code in Gradescope. After uploading your code, our grading scripts will autograde your assignment by running your program on a virtual machine (VM). We recommend debugging your implementation on your local machine (or the linux servers) and making sure your code is running correctly before any submission. Our autograder requires that you write your code using Python 3.6.9 and Numpy 1.17.0.
  - **Written questions:** You must type the answers in the provided .tex file. Handwritten solutions will not be accepted. Make sure to answer each question in the provided box. DO NOT change the size of the boxes, because it may mess up the autograder. Upon submission, make sure to label each question using the template provided by Gradescope. Please make sure to assign ALL pages corresponding to each question.

1 Course Policies [1pt]

1. [**1pt**] Have you read and understood the course policies?

   ● Yes

   ○ No

2 Probability Review [15pts]

A group of travellers find themselves lost in a cave. They come upon 3 tunnels A, B, C. Both tunnels A and B are closed loops that do not lead to an exit and in fact lead right back to the entrance of the 3 tunnels. Tunnel C is the tunnel which leads to the exit. If they go through tunnel A, then it takes 2 days to go through the tunnel. If they go through tunnel B, then it takes 1 day to go through the tunnel. If they go through tunnel C, then they immediately leave the cave. Suppose the travellers choose tunnels A, B and C with constant probability 0.3, 0.5, 0.2 every time. (For the following questions please round your answer to 4 digits.)

1. [**6 pts**] Suppose we record the travellers' choices as a sequence (e.g., $ABBA \dots C$). What is the probability that the pattern AAB appears in the sequence before any BAA appears?
   Note: You should also count cases where AAB appears in the sequence and BAA does not.

   Let E be the event that pattern AAB appears in the sequence before any BAA appears.
   The total probability that E happens is the sum of probabilities that E happens given that we first choose tunnel A, B, or C.
   $P(E) = P(E|A)P(A) + P(E|B)P(B) + P(E|C)P(C)$
   If we first choose tunnel B, then there is no way that AAB will appear before BAA.
   If we first choose tunnel C, then the sequence ends immediately.
   Therefore, $P(E) = P(E|A)P(A)$
   Following the same logic:
   $P(E|A) = P(E|AA)P(A) + P(E|AB)P(B) + P(E|AC)P(C)$
   If we choose B or C after we first choose A, then E will never happen.
   Therefore, $P(E|A) = P(E|AA)P(A)$
   $P(E|AA) = P(E|AAA)P(A) + P(E|AAB)P(B) + P(E|AAC)P(C)$
   We can see that $P(E|AA) = P(E|AAA)$ (choosing AAA will end up with the same state as choosing AA)
   so $P(E|AA) = P(E|AA) * P(A) + 1 * P(B)$
   $P(E|AA) = \frac{5}{7}$
   Therefore, $P(E) = \frac{5}{7} * 0.3 * 0.3 = 0.0643$

2. [**4 pts**] What is the expected number of days that the travellers will be lost in the cave?

$E[T] = 0.3 * (E[T] + 2) + 0.5 * (E[T] + 1) = 0.8 * E[T] + 1.1$
$0.2 * E[T] = 1.1$
$E[T] = 5.5$

3. **[5 pts]** What is the variance of days that the travellers will be lost in the cave? (Hint: To compute $Var(T)$ for a random variable T, compute $\mathrm{E}[T^2]$ first and apply the definition $Var(T) = E[T^2] - E[T]^2$.)

$E[T^2] = 0.3 * (E[T] + 2)^2 + 0.5 * (E[T] + 1)^2$
$= 0.3 * (5.5 + 2)^2 + 0.5(5.5 + 1)^2$
$= 16.875 + 21.125 = 38$

$Var(T) = E[T^2] - E[T]^2 = 38 - 5.5^2 = 7.75$

3 MLE and MAP [16 pts]

3.1 MLE [8 pts]

An exponential distribution with parameter λ has the probability density:

$$p(x) = \begin{cases} \lambda e^{-\lambda x} & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$$

- Given some i.i.d. data $\{x_i\}_{i=1}^n \sim \text{Exp}(\lambda)$, derive the maximum likelihood estimate (MLE) $\hat{\lambda}_{MLE}$.
- An estimator is unbiased if its expected value is equal to the true parameter it's estimating. Is this estimator biased?

$\hat{\lambda}_{MLE} = \text{argmax}_\lambda P_\lambda(x_1, x_2, ..., x_n)$
$= \text{argmax}_\lambda \prod_{i=1}^n P_\lambda(x_i)$
then $\hat{\lambda}_{MLE} = \text{argmax}_\lambda \prod_{i=1}^n \lambda e^{-\lambda x_i} = \text{argmax}_\lambda \lambda^n e^{(-\lambda)\sum_{i=1}^n x_i}$
$\hat{\lambda}_{MLE} = \text{argmax}_\lambda \ln(\lambda^n e^{(-\lambda)\sum_{i=1}^n x_i})$
$\hat{\lambda}_{MLE} = \text{argmax}_\lambda n\ln(\lambda) - \lambda \sum_{i=1}^n x_i$
We can then set $\frac{\partial}{\partial \lambda}(n\ln(\lambda) - \lambda\sum_{i=1}^n x_i) = 0$
so, $\frac{n}{\hat{\lambda}_{MLE}} = \sum_{i=1}^n x_i$, and $\hat{\lambda}_{MLE} = \frac{n}{\sum_{i=1}^n x_i}$

I think this estimator is biased. According to the hint on Piazza Hw1 FAQ, $\sum_{i=1}^n x_i$ is distributed as Gamma(n, λ). Let u denote $\sum_{i=1}^n x_i$.
Therefore, we can calculate $E[\frac{1}{u}] = \int_0^\infty \frac{1}{u} \cdot \frac{\lambda^n}{\Gamma(n)} u^{n-1} e^{-\lambda u} du = \frac{\lambda}{n-1} \int_0^\infty \frac{\lambda^{n-1}}{\Gamma(n-1)} u^{n-2} e^{-\lambda u} du$
Since what's inside the integral is Gamma$(n-1, \lambda)$ distribution, and we know that the integral of of probability distribution function is 1, we know that $E[\frac{1}{u}] = \frac{\lambda}{n-1}$, so $E[\hat{\lambda}_{MLE}] = \frac{n\lambda}{n-1} \neq \lambda$, which means the estimator is biased.

3.2 MAP [8 pts]

A gamma distribution with parameters α, β has a density function:

$$p(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x},$$

where $\Gamma(t)$ is the gamma function (see https://en.wikipedia.org/wiki/Gamma_distribution).

Suppose we start with a prior distribution for some parameters θ, and observe some data $\{x_i\}_{i=1}^n$ with likelihood $P(\text{data} \mid \theta)$. If the posterior for θ has the same form as the prior, then we say that the given prior is conjugate for the given likelihood.

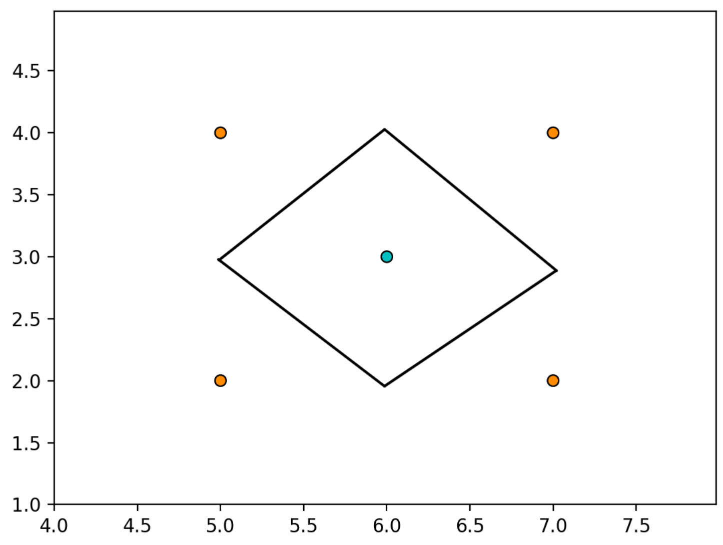
- Show that the Gamma distribution (that is $\lambda \sim \text{Gamma}(\alpha, \beta)$) is a conjugate prior of the Exp(λ) distribution. In other words, show that if the data points $x_i \sim \text{Exp}(\lambda)$ and $\lambda \sim \text{Gamma}(\alpha, \beta)$ then $P(\lambda|\text{data}) \sim \text{Gamma}(\alpha^*, \beta^*)$ for some values α^*, β^*.
- Derive the maximum a posteriori estimator (MAP) $\hat{\lambda}_{MAP}$ as a function of α, β. What happens as the number of data points n gets large?

As we have calculated in the last question,
$P(D|\lambda) = \lambda^n e^{(-\lambda)\sum_{i=1}^n x_i}$
and we also know $P(\lambda) = \frac{\beta^\alpha}{\Gamma(\alpha)}\lambda^{\alpha-1}e^{-\beta\lambda}$
We can then calculate $P(D|\lambda) * P(\lambda) = \frac{\beta^\alpha}{\Gamma(\alpha)}\lambda^{n+\alpha-1}e^{(-\lambda)((\sum_{i=1}^n x_i)+\beta)}$
Now we can tell that the posterior is also in the form of a gamma distribution.
($\frac{\beta^\alpha}{\Gamma(\alpha)}$ is a constant given α and β, so we don't need to worry about it)
so $P(\lambda|D) \sim \text{Gamma}(n+\alpha, (\sum_{i=1}^n x_i) + \beta)$

$\hat{\lambda}_{MAP} = \text{argmax}_\lambda\, \lambda^{n+\alpha-1}e^{(-\lambda)((\sum_{i=1}^n x_i)+\beta)}$
$= \text{argmax}_\lambda \ln(\lambda^{n+\alpha-1}e^{(-\lambda)((\sum_{i=1}^n x_i)+\beta)})$
$= \text{argmax}_\lambda (n+\alpha-1)\ln(\lambda) - \lambda((\sum_{i=1}^n x_i) + \beta)$
$\frac{\partial}{\partial\lambda}(n+\alpha-1)\ln(\lambda) - \lambda((\sum_{i=1}^n x_i) + \beta) = 0$
so $\frac{(n+\alpha-1)}{\lambda} = ((\sum_{i=1}^n x_i) + \beta)$, finally, $\hat{\lambda}_{MAP} = \frac{(n+\alpha-1)}{(\sum_{i=1}^n x_i)+\beta}$
When the number of data points get large, the effect of α and β on the posterior estimator will decrease.

4 K-Nearest Neighbors [15 Points]

1. **[3pt]** Consider K-NN using Euclidean distance on the following data set (each point belongs to one of two classes: orange or green).



   (a) **[2pt]** Draw the decision boundary for the above data set using the 1-NN classification rule.

   (b) **[1pt]** Would the decision boundary be different if another green point was added to the data set at $(6.1, 3.1)$? Why or why not?

The decision boundary will be different.
Take the point(7.05, 3) as an example.
Without adding the new point, the point (7.05, 3) is closer to the two orange points on the right.
However, when we have another green point at (6.1, 3.1), the point (7.05, 3) will be closer to the new green point, which means the decision boundary will change.

2. **[6pt]** k-NN Black Box

   (a) **[3pt]** In a k-NN classification problem, assume that the distance measure is not explicitly specified to you. Instead, you are given a "black box" where you input a set of instances $P_1, P_2, \dots P_n$ and a new example Q, and the black box outputs the nearest neighbor of Q, say P_i, and its corresponding class label C_i. Is it possible to construct a k-NN classification algorithm (w.r.t the unknown distance metric) based on this black box alone? If so, how? If not, why not? You may use the black-box more than once on any subset of $P_1, P_2, \dots P_n$.

   Yes, it is possible.
   For a new example Q, we can first run the black-box to find the nearest neighbor,
   then we get rid of this nearest neighbor and run the black-box again to find the second nearest neighbor.
   We do this for k times and find the majority class label.

   (b) **[3pt]** If the black box returns the j nearest neighbors (and their corresponding class labels) instead of the single nearest neighbor (assume $j \neq k$) , is it possible to construct a k-NN classification algorithm based on the black box ? If so how, and if not why not? Consider the cases when $j < k$ and $j > k$. Justify your argument. Again, you may use the black-box more than once on any subset of $P_1, P_2, \dots P_n$.

   For a new example Q
   If $j < k$, then we can first run the black-box for one time.
   We will then have j nearest neighbors of Q and we can record their labels.
   Now we get rid of any one of the j nearest neighbors, and run the black-box again.
   The output will contain one neighbor that is not included before, which is the $j + 1$ nearest neighbors of Q.
   We can simply repeat this process, until we find the k nearest neighbors.
   Finally, we can find the majority class label.

   If $j > k$, there is no way to construct a k-NN classifier, since we cannot sort the j nearest neighbors returned by the black-box.

3. [**6 pts**] Figure 1 shows K-NN classification decision boundaries with various values of K and distance metrics. Match each plot from Figure 1 to one of the corresponding K-NN settings.
To obtain full credit, you do not need to provide any justification, but only provide the correct pairing.

Definition: Manhattan distance: $d_M(x, y) = \sum^{D} |\mathbf{x} - \mathbf{y}|$
where $\mathbf{x}, \mathbf{y}$ are both vectors of size D

(a) $K = 1$, Euclidean Distance $\boxed{\text{a}}$

(b) $K = 25$, Euclidean Distance $\boxed{\text{c}}$

(c) $K = 25$, Manhattan Distance $\boxed{\text{b}}$

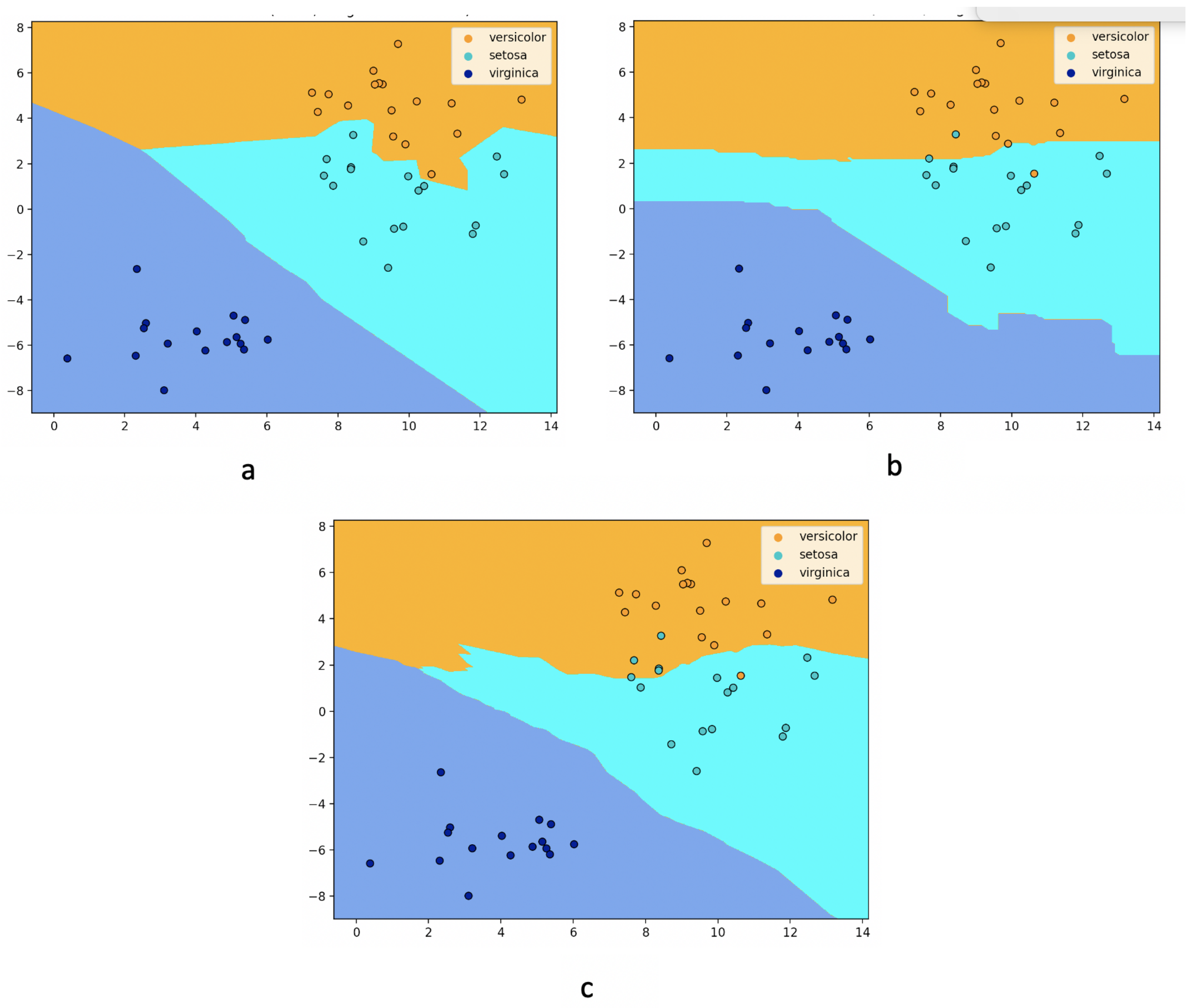


Figure 1: K-NN decision boundaries for various distance metrics and K values

5 Naive Bayes Programming [53 pts]

Programming Instructions

This part of the assignment will have you implement a Naive Bayes classifier. You will submit your completed `naive_bayes.py` file to Gradescope, where we will run your code against a suite of tests. Your grade will be automatically determined from the testing results. Since you get immediate feedback after submitting your code and you are allowed to submit as many different versions as you like (without any penalty), it easy for you to check your code as you go.

Our autograder requires that you write your code using Python 3.6.9 and Numpy 1.17.0. Otherwise, when running your program on Gradescope, it may produce a result different from the result produced on your local computer. **Please do not include print statements outside the provided functions, as this may crash the autograder.**

The file `hw2data.pkl` contains data regarding words used in articles from The Economist and articles from The Onion. This programming assignment is focused on identifying which words are characteristic of which articles. You can load the pickle file into Python using `pickle`. After loading the data, you will see that there are 5 variables: `Vocabulary`, `XTrain`, `yTrain`, `XTest`, and `yTest`.

- `Vocabulary` is a $V \times 1$ dimensional array that contains every word appearing in the documents. When we refer to the j^{th} word, we mean `Vocabulary[j,0]`.
- `XTrain` is a $n \times V$ dimensional matrix describing the n documents used for training your Naive Bayes classifier. The entry `XTrain[i,j]` is 1 if word j appears in the i^{th} training document and 0 otherwise.
- `yTrain` is a $n \times 1$ dimensional matrix containing the class labels for the training documents. `yTrain[i,0]` is 1 if the i^{th} document belongs to The Economist and 2 if it belongs to The Onion.
- Finally, `XTest` and `yTest` are the same as `XTrain` and `yTrain`, except instead of having n rows, they have m rows. This is the data you will test your classifier on and it should not be used for training.

Logspace Arithmetic

When working with very large or very small numbers (such as probabilities), it is useful to work in *logspace* to avoid numerical precision issues. In logspace, we keep track of the logs of numbers, instead of the numbers themselves. For example, if $p(x)$ and $p(y)$ are probability values, instead of storing $p(x)$ and $p(y)$ and computing $p(x) * p(y)$, we work in log space by storing $\log(p(x))$, $\log(p(y))$, and we can compute the log of the product, $\log(p(x) * p(y))$, by taking the sum in logspace: $\log(p(x) * p(y)) = \log(p(x)) + \log(p(y))$.

If we want the sum of two probabilities, it's a little trickier: if $l(x) = \log p(x)$ and $l(y) = \log p(y)$, then $\log(p(x) + p(y)) = \log(\exp(l(x)) + \exp(l(y)))$. If we compute this expression naively we risk overflow or underflow. A good workaround is to factor out $\exp(l(x))$ or $\exp(l(y))$, whichever is larger, before computing the sum.

Training Naive Bayes

1. [**8 Points**] Complete the function `D = NB_XGivenY(XTrain, yTrain, a=0.001, b=0.9)`. The output `D` is a $2 \times V$ matrix, where for any word index $w \in \{1, \ldots, V\}$ and class index $y \in \{1, 2\}$, the entry `D[y-1,w-1]` is the MAP estimate of $\theta_{yw} = P(X_w = 1|Y = y)$ with a Beta(1.001,1.9) prior distribution. Here we define $a = \alpha - 1$ and $b = \beta - 1$ where α, β are parameters of the Beta distribution. To help with numerical issues clip D to be in $[10^{-5}, 1 - 10^{-5}]$ before this function returns it.

2. [**8 Points**] Complete the function `p = NB_YPrior(yTrain)`. The output `p` is the MLE for $\rho = P(Y = 1)$.

3. [**8 Points**] Complete the function `yHat = NB_Classify(D, p, X)`. The input `X` is an $m \times V$ matrix containing m feature vectors (stored as its rows). The output `yHat` is a $m \times 1$ matrix of predicted class labels, where `yHat[i]` is the predicted label for the i^{th} row of `X`. So, the output vector should take the form $[[y_0],[y_1],...,[y_{m-1}]]$. [Hint: In this function, you will want to use Logspace Arithmetic to avoid numerical problems.]

4. [**2 Points**] Complete the function `e = NB_ClassificationAccuracy(yHat, yTruth)` which measures the average number of times yHat agrees with yTruth as a performance metric for the Naive Bayes classifier.

Questions

5. [**4 Points**] Train your classifier on the data contained in `XTrain` and `yTrain` by running

   ```
   D = NB_XGivenY(XTrain, yTrain)
   p = NB_YPrior(yTrain)
   ```

   Use the learned classifier to predict the labels for the article feature vectors in `XTrain` and `XTest` by running

   ```
   yHatTrain = NB_Classify(D, p, XTrain)
   yHatTest = NB_Classify(D, p, XTest)
   ```

   Use the function `NB_ClassificationAccuracy` to measure and report the training and testing accuracy by running

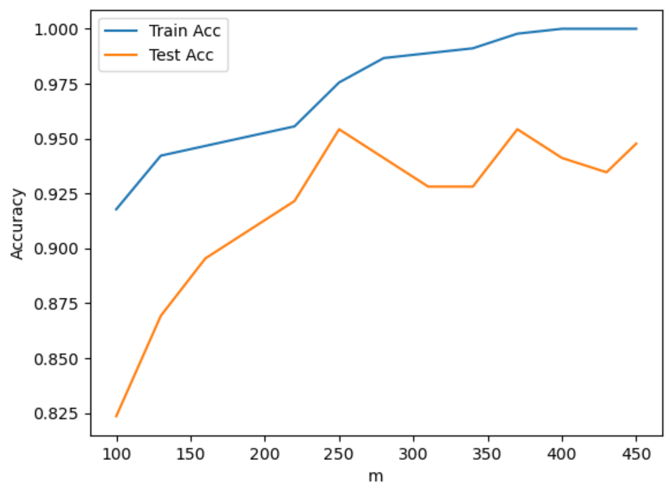
   ```
   trainAcc = NB_ClassificationAccuracy(yHatTrain, yTrain)
   testAcc = NB_ClassificationAccuracy(yHatTest, yTest)
   ```

   How do the train and test accuracies compare? Which is likely to be more representative of the performance of the trained classifier on a new collection of articles?

The train accuracy is 1.0, and the test accuracy is 0.9477.
The test accuracy is likely to be more representative of the performance of the trained classifier on a new collection of articles,
because the test data and the new collection of articles are both unseen data.

6. [**5 Points**] In this question we explore how the size of the training data set affects the test and train accuracy. For each value of m in $\{100, 130, 160, \ldots, 450\}$, train your Naive Bayes classifier on the first m training examples (that is, use the data given by `XTrain[0:m]` and `yTrain[0:m]`). Plot the training and testing accuracy for each such value of m. The x-axis of your plot should be m, the y-axis should be accuracy, and there should be one curve for training accuracy and one curve for testing accuracy.

   - Explain the general trend of both the curves.
   - What would you expect to happen to the test accuracy of the classifier if the Naive Bayes assumption is satisfied and we have infinite training data?



We can see that as m increases, train and test accuracy both increase. I think this is because with more data, our estimated parameters and their distributions will be more accurate.

I think if the Naive Bayes assumption is satisfied and we have infinite training data, the test accuracy can reach 1.0, because as the size of data increases, training data will become more and more representative of unseen data.

7. [**4 Points**] We will try to interpret the learned parameters. Train your classifier on the data contained in `XTrain` and `yTrain`. For each class label $y \in \{1, 2\}$, create the lists according to the following criteria (Note that some of the words may look a little strange because we have run them through a stemming algorithm that tries to make words with common roots look the same. For example, "stemming" and "stemmed" would both become "stem"):

   - Top five words that the model says are most likely to occur in a document from class y. That is, the top five words according to this metric:

   $$P(X_w = 1|Y = y)$$

   - Top five words w according to this metric:

   $$\frac{P(X_w = 1|Y = y)}{P(X_w = 1|Y \neq y)}.$$

   Which list of words is more informative about the class y? Briefly explain your reasoning.

Using the first metric:

Top five words when $Y = 1$: to, of, the, a, in

Top five words when $Y = 2$: said, and, that, in, of

Using the second metric:

Top five words when $Y = 1$: parliament, organis, favour, labour, reckon

Top five words when $Y = 2$: 4enlarg, 5enlarg, percent, realiz, center

I think the second list of words are more informative,
because we eliminate the words that appear a lot for both of the classes,
and this leaves us with words that appear frequently in only one class.

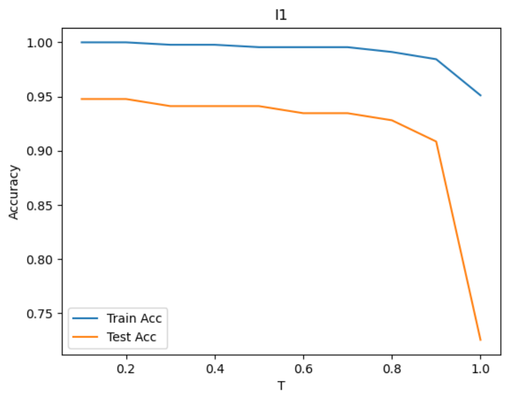
8. [**4 Points**] Having a metric to tell us the importance of a particular word in helping us classify text can be useful when wanting to shrink the dataset without affecting accuracy. Plot graphs of the training and testing accuracy of your model where you prune unimportant words from the feature set. You should compute the following importance values for every word:

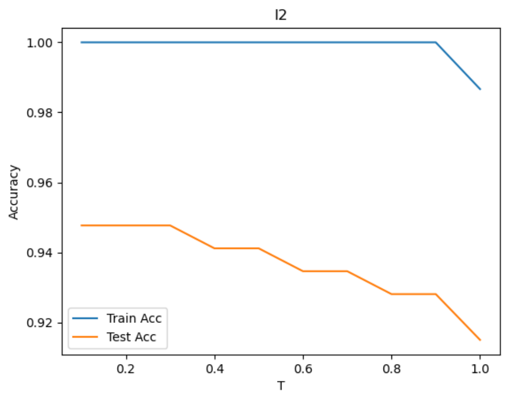
$$I_{0/1} = abs(1 - \frac{P(X_w = 1|Y = 1)}{P(X_w = 1|Y = 2)}).$$

$$I_{1/0} = abs(1 - \frac{P(X_w = 1|Y = 2)}{P(X_w = 1|Y = 1)}).$$

For a given threshold value T, you should keep words where the importance I is greater than the threshold, and discard words below the threshold (You will need to do this separately for both values of I). Plot a graph of the test and train accuracy over $T \in [0.1, 1.0]$ with step size 0.1 [Hint: It may be useful to pre-compute a mask and apply it to $P(X|Y)$]. Report the two graphs (one for each I) in your writeup.

- What is the reason for the graph's shape?
- How many vocab words are able to be remove before the accuracy is affected?

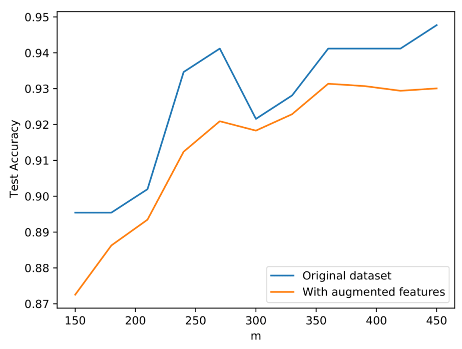




Initially, the accuracy won't decrease or only decrease slightly, because we only remove some of the words with low importance. As we remove more and more words with higher importance, the decrease in performance becomes more significant.
For $I_{0/1}$, the accuracy drops after $T = 0.2$ (accuracy unaffected at $T = 0.2$), where we have removed 7441 words.
For $I_{1/0}$, the accuracy drops after $T = 0.3$ (accuracy unaffected at $T = 0.3$), where we have removed 7663 words.

9. [**5 Points**] The `augmentFeatures` function is designed to augment each feature vector of length V with a subset of size v' of new features. These features actually happen to be exact copies of existing features — although the classifier doesn't know that, and will treat them the same as all the other features. The feature augmentation process is identical in the training and test datasets.

   We want to observe the effect of this augmentation on the accuracy of the Naive Bayes Classifier. The classifier is trained and tested separately both on the original dataset and the modified dataset with augmented feature vectors. For each value of m in $\{150, 180, \ldots, 450\}$, the Naive Bayes classifier is trained on the first m training examples and only the test accuracies are plotted separately for the original and augmented dataset. Generate the plot by running the `augmentFeatures` function and explain your observation. Does the augmentation of the feature vector violate any assumption for the Naive Bayes classifier? Why is the test accuracy of the trained Naive Bayes classifier affected by augmentation of the feature vector in this manner?



The test accuracy for our original dataset is higher than that of the modified dataset.
The augmentation of the feature vector violates the assumption that each feature should be independent given class. Having duplicated features means these features will affect the classification more than they should, leading to a decrease in accuracy.

10. **[5 Points]** Use the `generateData` function to generate a new dataset of size 1000 from your learned model, and then report the classification accuracy of your model (which was trained on the original training set) on this new data set. Explain how the new dataset is constructed using the generative properties of the Naive Bayes model.

The classification accuracy is 1.0

We first draw 1000 samples from a binomial distribution with probability of success ($Y = 1$) being prior p. For simplicity, let the number of class $Y = 1$ data be n, so the number of class $Y = 2$ data be $1000 - n$. Now we have $yData$.

We already have the matrix D of $P(X_w = 1|Y)$, so for each word, we can draw n samples from a binomial distribution with probability of success ($X_w = 1$) being $P(X_w = 1|Y = 1)$. Following the same logic, we can draw $1000 - n$ samples from a binomial distribution with probability of success ($X_w = 1$) being $P(X_w = 1|Y = 2)$. We can repeat this step for all the words, then combine them together to obtain $XData$.

Collaboration Questions Please answer the following:

1. Did you receive any help whatsoever from anyone in solving this assignment?
   Yes / **No**.
   - If you answered 'yes', give full details: ____________________
   - (e.g. "Jane Doe explained to me what is asked in Question 3.4")

2. Did you give any help whatsoever to anyone in solving this assignment?
   Yes / **No**.
   - If you answered 'yes', give full details: ____________________
   - (e.g. "I pointed Joe Smith to section 2.3 since he didn't know how to proceed with Question 2")

3. Did you find or come across code that implements any part of this assignment ?
   Yes / **No**. (See below policy on "found code")
   - If you answered 'yes', give full details: ____________________
   - (book & page, URL & location within the page, etc.).